

AURIX™

TC26x B-Step

32-Bit Single-Chip Microcontroller

User's Manual

V1.3 2014-12

Microcontrollers

Edition 2014-12

**Published by
Infineon Technologies AG
81726 Munich, Germany**

**© 2014 Infineon Technologies AG
All Rights Reserved.**

Legal Disclaimer

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics. With respect to any examples or hints given herein, any typical values stated herein and/or any information regarding the application of the device, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation, warranties of non-infringement of intellectual property rights of any third party.

Information

For further information on technology, delivery terms and conditions and prices, please contact the nearest Infineon Technologies Office (www.infineon.com).

Warnings

Due to technical requirements, components may contain dangerous substances. For information on the types in question, please contact the nearest Infineon Technologies Office.

Infineon Technologies components may be used in life-support devices or systems only with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

AURIX™

TC26x B-Step

32-Bit Single-Chip Microcontroller

User's Manual

V1.3 2014-12

Microcontrollers

TC26x B-Step User's Manual
Revision History: V1.3 2014-12
Significant changes since previous version: TC26x User's Manual V1.2

Chapter	Subjects (major changes since last revision)
Introduction	Removed redundant introduction sections.
FIRMWARE	<ul style="list-style-type: none"> Added description of the CRC calculation method below the table "Boot Mode Header (BMHD)"
SCU/CCU	<ul style="list-style-type: none"> Updated divider formula in registers CCUCON6,7,8. Updated table "CCU allowed Clock Ratios".
SCU/SCU	<ul style="list-style-type: none"> Corrected DTSCON.CAL width. Removed design related register reference from "OCDS Behavior of WDTs".
PORTS	<ul style="list-style-type: none"> Corrected description of LVDS pads in CMOS mode: removed sentence 'no open-drain mode available'
MTU	<ul style="list-style-type: none"> MCONTROL.Res4 field (bits 14:12) marked as "Must be written with 0x4" in to maintain redundancy. Sections "Reading a Single Memory Location" and "Writing a Single Memory Location" should always write MCONTROL with the MS nibble 0x4 instead of 0x0. Design related register reference "RDR" removed.

TC26x B-Step User's Manual

Revision History: V1.3 2014-12

SMU	<ul style="list-style-type: none"> • Corrected register long name of SMU_RTC to “Recovery Timer Configuration”. • DMA alarm ALM2[24] is now reserved. Alarm implemented by DMA SRI error detection alarms. • Improved description of alarm ALM3[27]. • SCU alarm ALM3[28] is now reserved. Feature not implemented. • Pre-alarms PreAlarm[129=DMA.(Safety FF Error Detection)] and PreAlarm[130=PMU.(Safety FF Error Detection)] removed because DMA and PMU do not implement Safety FF. • Added configuration hints to SMU_AG<n>CF<m> Registers: reserved alarms shall be configured as “No Action”. • Enhanced description of register SMU_RMEF and SMU_RMSTS. • Enhanced description of register SMU_PCTL with advice how to operate it in the application. • Changed section “SMU Integration Guidelines”. • Explained “Register Monitor” = “Safety Flip-Flop”. • Improved Figure “SMU State Machine”. • Improved section “FSP Fault State”.
-----	--

TC26x B-Step User's Manual

Revision History: V1.3 2014-12

PMU	<ul style="list-style-type: none"> Highlighted in the description of FCON and the wait cycle description that after System Resets and Power-On Resets the wait cycles are only sufficient for 100 MHz. Noted in the "Application Hints" that the recommended sequence for issuing "Verify Erased Logical Sector Range" is analog to "Erase". Changed recommendations in the "Application Hints" for ECC test patterns. Made the all-0/all-1 and address error patterns optional. Recommended to store 2 of the 4 pattern sets inverse to the other 2. Improved description of "Load Page" to make the offsets used for 32-bit transfers more explicit. Added to the description of "Write Page" the clear statement that each page shall be programmed only once. Noted that "Verify Erased Logical Sector Range" is not blocked on password containing UCBs when the protection is disabled. Removed confusing statement that unavailable DFlash ranges can be programmed to all-1. Noted explicitly that the UCBs 8 to 11 are read-only. Clarified that both "Write Page" and "Write Burst" can be used to program UCBs. Removed statement about restricted FAR capabilities. Removed note from "Erase Logical Sector Range" that warned from erasing more than 256 KByte because of increased suspend times. The suspend times are not increasing. Changed note for "Erase Logical Sector Range" that described that erase time can be much shorter than maximum times documented in the Data Sheet. The reason was removed as there are more dependencies. Documented "Test Pass Marker" in UCB_IFX. Removed term "Reset Class" and replaced by named resets. Stated explicitly that a failed HSM boot prevents device boot.
IR	<ul style="list-style-type: none"> Added clarification: a pending Service Request is cleared in the Service Request Node with the Acknowledge from the Service Provider. The Acknowledge does not mean that the Service Request is serviced at that time (e.g. disabled or not implemented DMA channel, Trap on CPU prevents execution of Service Request).
ASCLIN	<ul style="list-style-type: none"> Block "Auto Baud Rate Operation" added. Description of register field "BRD.MEASURED" improved.

TC26x B-Step User's Manual
Revision History: V1.3 2014-12

QSPI	<ul style="list-style-type: none"> Removed maximum baud rates from feature list as these depend on pad speeds and PCB implementation.
MSC	<ul style="list-style-type: none"> Description of "Clock Control when using the ABRA Block" improved. Equation of Ncycles for extended frames changed: added factor 32 to PPDE.
MultiCAN+	<ul style="list-style-type: none"> Fixed pad naming for all devices. CRC issue of CAN FD standard documented and recommended additional software CRC.
SENT	<ul style="list-style-type: none"> Updated Serial Message Frame Chapter.
GTM	<ul style="list-style-type: none"> Corrected error in BGA-416 TOUT numbering for P10.13 and P10.11. Connection shown in figures and register for SET5 and SET6 to MSC2 was corrected. Updated description of bit field OCS.SUS. Updated description of (A)TOM SOMP Mode. Clarified reset behavior of debug registers. Corrected error for P2.11; it is connected to TIM4_4 not to TIM4_7
VADC	<ul style="list-style-type: none"> Added hint how to use Pxx_PDISC for diagnostics in section "Signal Path Test Modes". Adjusted the position of accumulated results in section "Data Alignment".
DSADC	<ul style="list-style-type: none"> Changed in section "Recommended Settings" the documented full-scale values from $\pm 1900_D$ to $\pm 3800_D$ and their dependence on operating modes. Removed external modulator connections. Removed modulator combine mode. Separated rectifier and integrator, add data shift in figure DSADC structure overview.

Trademarks

TriCore[®] is a trademark of Infineon Technologies AG.

Copyrights

Portions Copyright © 2013 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc.

We Listen to Your Comments

Is there any information in this document that you feel is wrong, unclear or missing? Your feedback will help us to continuously improve the quality of this document. Please send your proposal (including a reference to this document) to:

mcdocu.comments@infineon.com



Trademarks of Infineon Technologies AG

AURIX[™], C166[™], CanPAK[™], CIPOS[™], CIPURSE[™], EconoPACK[™], CoolMOS[™], CoolSET[™], CORECONTROL[™], CROSSAVE[™], DAVE[™], DI-POL[™], EasyPIM[™], EconoBRIDGE[™], EconoDUAL[™], EconoPIM[™], EconoPACK[™], EiceDRIVER[™], eupec[™], FCOS[™], HITFET[™], HybridPACK[™], I²RF[™], ISOFACE[™], IsoPACK[™], MIPAQ[™], ModSTACK[™], my-d[™], NovalithIC[™], OptiMOS[™], ORIGA[™], POWERCODE[™], PRIMARION[™], PrimePACK[™], PrimeSTACK[™], PRO-SIL[™], PROFET[™], RASIC[™], ReverSave[™], SatRIC[™], SIEGET[™], SINDRION[™], SIPMOS[™], SmartLEWIS[™], SOLID FLASH[™], TEMPFET[™], thinQ![™], TRENCHSTOP[™], TriCore[™].

Other Trademarks

Advance Design System[™] (ADS) of Agilent Technologies, AMBA[™], ARM[™], MULTI-ICE[™], KEIL[™], PRIMECELL[™], REALVIEW[™], THUMB[™], μ Vision[™] of ARM Limited, UK. AUTOSAR[™] is licensed by AUTOSAR development partnership. Bluetooth[™] of Bluetooth SIG Inc. CAT-iq[™] of DECT Forum. COLOSSUS[™], FirstGPS[™] of Trimble Navigation Ltd. EMV[™] of EMVCo, LLC (Visa Holdings Inc.). EPCOS[™] of Epcos AG. FLEXGO[™] of Microsoft Corporation. FlexRay[™] is licensed by FlexRay Consortium. HYPERTERMINAL[™] of Hilgraeve Incorporated. IEC[™] of Commission Electrotechnique Internationale. IrDA[™] of Infrared Data Association Corporation. ISO[™] of INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. MATLAB[™] of MathWorks, Inc. MAXIM[™] of Maxim Integrated Products, Inc. MICROTEC[™], NUCLEUS[™] of Mentor Graphics Corporation. MIPI[™] of MIPI Alliance, Inc. MIPS[™] of MIPS Technologies, Inc., USA. muRata[™] of MURATA MANUFACTURING CO., MICROWAVE OFFICE[™] (MWO) of Applied Wave Research Inc., OmniVision[™] of OmniVision Technologies, Inc. Openwave[™] Openwave Systems Inc. RED HAT[™] Red Hat, Inc. RFMD[™] RF Micro Devices, Inc. SIRIUS[™] of Sirius Satellite Radio Inc. SOLARIS[™] of Sun Microsystems, Inc. SPANSION[™] of Spansion LLC Ltd. Symbian[™] of Symbian Software Limited. TAIYO YUDEN[™] of Taiyo Yuden Co. TEAKLITE[™] of

CEVA, Inc. TEKTRONIX™ of Tektronix Inc. TOKO™ of TOKO KABUSHIKI KAISHA TA. UNIX™ of X/Open Company Limited. VERILOG™, PALLADIUM™ of Cadence Design Systems, Inc. VLYNQ™ of Texas Instruments Incorporated. VXWORKS™, WIND RIVER™ of WIND RIVER SYSTEMS, INC. ZETEX™ of Diodes Zetex Limited.

Last Trademarks Update 2011-11-11

Table of Contents

1	TC26x B-Step Introduction	1-1
1.1	About this Document	1-1
1.1.1	Related Documentations	1-1
1.1.2	Text Conventions	1-1
1.1.3	Reserved, Undefined, and Unimplemented Terminology	1-3
1.1.4	Register Access Modes	1-3
1.1.5	Abbreviations and Acronyms	1-4
1.2	System Architecture of the TC26x B-Step	1-8
1.2.1	TC26x B-Step Block Diagram	1-9
1.3	On-Chip Debug Support (OCDS)	1-10
1.3.1	Feature List	1-11
1.3.2	Family Overview	1-12
1.3.3	Tool Interface Recommendations	1-13
1.3.4	Debug Access Server (DAS)	1-15
1.4	Emulation Device (ED)	1-16
1.4.1	Block Diagram	1-16
1.4.2	Feature List	1-18
1.4.3	Comparison to TC27xED	1-19
1.4.4	Trace Based Measurement	1-20
1.4.5	ED Design and Layout	1-21
1.4.6	Emulation System Components	1-21
1.4.6.1	Emulation Memory (EMEM)	1-21
1.4.6.2	Calibration Memory	1-22
1.4.6.3	Trace Memory	1-22
1.4.7	MCDS	1-23
1.4.8	DAP/JTAG based Tool Interface (IOC32)	1-24
1.4.9	Aurora GigaBit Trace (AGBT)	1-24
2	On-Chip System Buses and Bus Bridges	2-1
2.1	What is new	2-2
2.2	SRI Crossbar (XBar_SRI)	2-3
2.2.1	Introduction	2-3
2.2.1.1	XBar_SRI Features	2-5
2.2.2	SRI Transactions	2-5
2.2.3	SRI Op-Codes	2-6
2.2.4	SRI Error Conditions	2-7
2.2.5	SRI Transaction ID Error Conditions	2-7
2.2.6	Operational Overview	2-8
2.2.6.1	Functional Blocks	2-9
2.2.7	Functional Overview	2-12
2.2.7.1	Arbitration Block	2-12

Table of Contents

2.2.7.2	Default Slave	2-16
2.2.7.3	Register Access Protection	2-18
2.2.7.4	SRI ECC Error Handling	2-19
2.2.7.5	Error Tracking Capability	2-21
2.2.7.6	Debug Trigger Event Generation (OCDS Level 1)	2-22
2.2.7.7	Interrupt and Debug Events of the XBar_SRI Module	2-24
2.2.8	Implementation of the Cross Bar (XBar_SRI) in the TC26x B-Step	2-25
2.2.8.1	Mapping of SRI Master Modules to XBar_SRI Master Interfaces	2-26
2.2.8.2	Mapping of SRI Slave modules to XBar_SRI Slave Interfaces	2-27
2.2.8.3	TC26x B-Step SRI Master / Slave Interconnection Matrix	2-28
2.2.8.4	Connection Master-Slave in XBar_SRI	2-28
2.2.9	SRI Crossbar Registers	2-30
2.2.9.1	TC26x B-Step Control Registers	2-36
2.3	Shared Resource Interconnect to FPI Bus Interface (SFI Bridge)	2-71
2.3.1	Functional Overview	2-71
2.4	System Peripheral Bus	2-72
2.4.1	Overview	2-72
2.4.2	Bus Transaction Types	2-74
2.4.3	Reaction of a Busy Slave	2-74
2.4.4	Address Alignment Rules	2-75
2.4.5	FPI Bus Basic Operations	2-75
2.5	FPI Bus Control Unit (SBCU)	2-77
2.5.1	FPI Bus Arbitration	2-77
2.5.1.1	Arbitration on the System Peripheral Bus	2-77
2.5.1.2	Default Master	2-77
2.5.1.3	Arbitration Algorithms	2-77
2.5.1.4	Starvation Prevention	2-79
2.5.2	FPI Bus Error Handling	2-79
2.5.3	System Registers	2-81
2.5.3.1	Register Access Protection (ACCEN1/0)	2-82
2.5.3.2	Kernel Reset Registers (KRST1/0, KRSTCLR)	2-82
2.5.3.3	Clock Control Register (CLC)	2-82
2.5.3.4	OCDS Control and Status Register (OCS)	2-82
2.5.4	BCU Debug Support	2-83
2.5.4.1	Address Triggers	2-83
2.5.4.2	Signal Status Triggers	2-84
2.5.4.3	Grant Triggers	2-85
2.5.4.4	Combination of Triggers	2-86
2.5.4.5	BCU Breakpoint Generation Examples	2-86
2.5.5	System Bus Control Unit Registers	2-88
2.5.5.1	SBCU System Registers	2-91
2.5.5.2	SBCU Control Registers Descriptions	2-94
2.5.5.3	SBCU Error Registers Descriptions	2-97

Table of Contents

2.5.5.4	SBCU OCDS Registers Descriptions	2-102
2.6	On Chip Bus Master TAG Assignments	2-115
2.7	On Chip Bus Access Times	2-116
3	Memory Maps	3-1
3.1	How to Read the Address Maps	3-2
3.2	Contents of the Segments	3-4
3.3	Address Map of the On Chip Bus System	3-6
3.3.1	Segments 0 to 14	3-6
3.3.2	Segment 15	3-11
3.4	Memory Module Access Restrictions	3-18
3.5	Side Effects from Modules to CPU0 Data Scratch Pad SRAM (DSPR0)	3-19
4	TC26x B-Step BootROM Content	4-1
4.1	Startup Software	4-1
4.1.1	Events triggering SSW execution	4-1
4.1.1.1	Power-on	4-1
4.1.1.2	System reset	4-2
4.1.1.3	Application reset	4-2
4.1.2	Clock system during start-up	4-3
4.1.3	RAM overwrite during start-up	4-3
4.1.4	Boot Options Summary	4-3
4.1.5	Start-up mode selection	4-4
4.1.5.1	Hardware configuration	4-9
4.1.5.2	Configuration by Boot Mode Index (BMI)	4-9
4.1.6	Startup Software Main Flow	4-12
4.1.6.1	Basic Device Settings	4-12
4.1.6.2	Stand-by Controller preparation	4-13
4.1.6.3	RAMs Handling	4-13
4.1.6.4	Select and Prepare Startup Modes	4-13
4.1.6.5	Final Chip Settings	4-14
4.1.6.6	Ending the SSW and Starting the User Code	4-16
4.2	Bootstrap Loaders	4-17
4.2.1	ASC Bootstrap loader	4-17
4.2.2	CAN Bootstrap Loader	4-17
4.2.3	Summary of Bootstrap Loader Modes	4-19
4.3	Shutdown request handler	4-19
4.4	Power Supply Friendly Debug Monitor	4-20
4.4.1	PSFDM code with inverse exit condition	4-20
5	CPU Subsystem	5-1
5.1	Aurix Family CPU configurations	5-2
5.2	Central Processing Unit Features	5-4
5.3	TC1.6P Implementation Overview	5-6

Table of Contents

5.3.1	CPU Diagram	5-6
5.3.2	Instruction Fetch Unit	5-6
5.3.3	Execution Unit	5-7
5.3.4	General Purpose Register File	5-8
5.4	TC1.6E Implementation Overview	5-10
5.4.1	CPU Diagram	5-10
5.4.2	Instruction Fetch Unit	5-11
5.4.3	Execution Unit	5-12
5.4.4	General Purpose Register File	5-13
5.5	Summary of functional changes from TC1.3.1	5-14
5.6	CPU Implementation-Specific Features	5-15
5.6.1	Context Save Areas / Context Operations	5-15
5.6.2	Program Counter (PC) Register	5-15
5.6.3	Store Buffers	5-16
5.6.4	Interrupt System	5-17
5.6.5	Trap System	5-17
5.6.6	Memory Integrity Error Handling	5-19
5.6.6.1	Program Side Memories	5-19
5.6.6.2	Data Side Memories	5-20
5.6.6.3	Memory Initialisation	5-22
5.6.7	WAIT Instruction	5-22
5.6.8	Instruction Memory Range Limitations	5-22
5.6.9	Atomicity of Data Accesses	5-23
5.6.10	A11 usage	5-24
5.7	Memory Addressing	5-25
5.7.1	CSFR and SFR base Locations	5-25
5.7.2	Local and Global Addressing	5-25
5.7.3	Cache Memory Access	5-26
5.8	CPU Subsystem Registers	5-27
5.8.1	CPU Core Special Function Registers (CSFR)	5-28
5.8.1.1	Registers	5-28
5.8.2	CPU General Purpose Registers	5-40
5.8.3	CPU Memory Protection Registers	5-40
5.8.4	Temporal Protection Registers	5-40
5.8.5	FPU Registers	5-41
5.8.6	Memory Integrity Registers	5-42
5.8.6.1	Register Descriptions	5-43
5.8.7	CPU Core Debug and Performance Counter Registers	5-53
5.8.7.1	Counter Source Details	5-53
5.8.8	Summary of CSFR Reset Values and Access Modes	5-56
5.8.9	Summary of SFR Reset Values and Access modes	5-67
5.9	CPU Instruction Timing	5-71
5.9.1	Integer-Pipeline Instructions	5-72

Table of Contents

5.9.1.1	Simple Arithmetic Instruction Timings	5-72
5.9.1.2	Multiply Instruction Timings	5-76
5.9.1.3	Multiply Accumulate (MAC) Instruction Timing	5-77
5.9.1.4	Control Flow Instruction Timing TC1.6P	5-78
5.9.1.5	Control Flow Instruction Timing TC1.6E	5-79
5.9.2	Load-Store Pipeline Instructions	5-80
5.9.2.1	Address Arithmetic Timing	5-80
5.9.2.2	CSA Control Flow Instruction Timing	5-81
5.9.2.3	Load Instruction Timing	5-81
5.9.2.4	Store Instruction Timing	5-83
5.9.3	Floating Point Pipeline Timing	5-84
5.10	Program Memory Interface (PMI)	5-85
5.10.1	TC1.6P PMI Description	5-85
5.10.1.1	TC1.6P Scratchpad RAM	5-86
5.10.1.2	TC1.6P Program Cache	5-86
5.10.1.3	TC1.6P Program Line Buffer	5-88
5.10.1.4	TC1.6P CPU Slave Interface (CPS)	5-88
5.10.2	TC1.6E PMI Description	5-89
5.10.2.1	TC1.6E Program Scratchpad RAM	5-90
5.10.2.2	TC1.6E Program Cache	5-90
5.10.2.3	TC1.6E Program Line Buffer	5-92
5.10.2.4	TC1.6E CPU Slave Interface (CPS)	5-92
5.10.3	PMI Registers	5-93
5.10.3.1	PMI Register Descriptions	5-93
5.11	Data Memory Interface (DMI)	5-98
5.11.1	TC1.6P DMI Description	5-98
5.11.1.1	TC1.6P Data Scratchpad RAM (DSPR)	5-99
5.11.1.2	TC1.6P Data Cache (DCACHE)	5-99
5.11.2	TC1.6E DMI Description	5-101
5.11.2.1	TC1.6E Data Scratchpad RAM (DSPR)	5-102
5.11.2.2	TC1.6E Data Read Buffer (DRB)	5-102
5.11.3	DMI Trap Generation	5-103
5.11.4	DMI Registers	5-105
5.11.4.1	DMI Register Descriptions	5-105
5.12	Safety Features	5-111
5.12.1	SRI Data Master Address Phase Error Injection	5-111
5.12.2	SRI Data Master Write Phase Error Injection	5-111
5.12.3	SRI Data Slave Read Phase Error Injection	5-111
5.12.4	SRI Error Capture	5-112
5.12.5	SRI Safe Data Master tag	5-112
5.12.6	Safety Memory Protection	5-112
5.12.7	Safety Register Protection	5-113
5.12.8	Lock Step Implementation	5-113

Table of Contents

5.12.9	MBIST usage recommendations	5-114
5.12.10	Registers Implementing Safety Features	5-115
6	Lockstep Comparator Logic (LCL)	6-1
6.1	Feature List	6-1
6.2	Lockstep Control	6-1
6.3	Lockstep Monitoring	6-1
6.4	Lockstep Self Test	6-2
6.5	Lockstep Failure Signalling Test	6-4
6.6	Functional Redundancy	6-4
6.7	Revision History	6-5
7	System Control Units	7-1
7.1	Clocking and Clock Control Unit (CCU)	7-3
7.1.1	Clock Sources	7-3
7.1.1.1	Oscillator Circuit (OSC)	7-4
7.1.1.2	Back-up Clock	7-12
7.1.2	Clock Speed Upscaling	7-12
7.1.2.1	Phase-Locked Loop (PLL) Module	7-12
7.1.2.2	ERAY Phase-Locked Loop (PLL_ERAY) Module	7-28
7.1.3	Clock Distribution	7-41
7.1.3.1	Clock Control Unit	7-44
7.1.4	Individual Clock Generation	7-68
7.1.4.1	Clock Control Register CLC	7-68
7.1.5	Clock Monitors	7-72
7.1.5.1	Clock Monitor Registers	7-74
7.1.6	Clock Emergency Behavior	7-78
7.1.7	External Clock Output	7-79
7.1.7.1	Programmable Frequency Output for EXTCLK0	7-79
7.1.7.2	Programmable Frequency Output for EXTCLK1	7-82
7.1.7.3	Clock Output Control Register	7-84
7.1.8	Clock Generation Unit	7-88
7.1.8.1	Example Sequence	7-88
7.1.9	CCU Register Address	7-90
7.1.10	CCU Kernel Registers	7-90
7.2	Reset Control Unit (RCU)	7-93
7.2.1	Reset Operation	7-94
7.2.1.1	Overview	7-94
7.2.1.2	Reset Types	7-94
7.2.1.3	Reset Sources Overview	7-95
7.2.1.4	Warm and Cold Resets	7-95
7.2.1.5	EVR Resets and PORST	7-96
7.2.1.6	Module Reset Behavior	7-96
7.2.1.7	General Reset Operation	7-98

Table of Contents

7.2.1.8	Reset Generation	7-98
7.2.1.9	Shutdown and Reset Delay Timeout Counter (TOUTCNT)	7-99
7.2.1.10	Reset Triggers	7-100
7.2.1.11	Debug Reset Specific Behavior	7-101
7.2.1.12	Module Resets	7-101
7.2.1.13	Reset Controller Registers	7-103
7.2.2	External Reset Sources and Indications	7-113
7.2.2.1	External Service Requests (ESRx)	7-113
7.2.3	Boot Software Interface	7-125
7.2.3.1	Configuration done with Start-up	7-125
7.2.3.2	Start-up Configuration Options	7-125
7.2.3.3	Status Registers	7-126
7.2.4	NMI Trap Generation	7-129
7.2.4.1	Trap Control Registers	7-130
7.2.5	RCU Register Address	7-134
7.2.6	RCU Kernel Registers	7-134
7.3	Power Supply & Power Management Controller (PMC)	7-137
7.3.1	Power Supply and Control	7-138
7.3.1.1	Introduction	7-138
7.3.1.2	Supply Mode and Topology Selection	7-138
7.3.1.3	Linear Regulator Mode	7-143
7.3.1.4	Step-down Regulator Mode	7-146
7.3.1.5	External Supply Modes	7-149
7.3.1.6	Components and Layout	7-150
7.3.1.7	Voltage Monitoring	7-152
7.3.1.8	100 MHz EVR Clock Source	7-156
7.3.1.9	Sequence during Power-up and Power-down	7-157
7.3.1.10	EVR Control Registers	7-165
7.3.2	Power Management	7-193
7.3.2.1	Power Management Overview	7-193
7.3.2.2	Idle Mode	7-196
7.3.2.3	Sleep Mode	7-198
7.3.2.4	Standby Mode	7-199
7.3.2.5	Power Management Registers	7-205
7.3.3	Power Management Register Address	7-222
7.3.4	PMC Kernel Registers	7-222
7.4	System Control Unit (SCU)	7-227
7.4.1	External Request Unit (ERU)	7-228
7.4.1.1	Introduction	7-228
7.4.1.2	ERU Input Pin Connections	7-231
7.4.1.3	External Request Selector Unit (ERS)	7-232
7.4.1.4	Event Trigger Logic (ETL)	7-232
7.4.1.5	Connecting Matrix	7-234

Table of Contents

7.4.1.6	Output Gating Unit (OGU)	7-236
7.4.1.7	ERU Output Pin Connections	7-240
7.4.1.8	External Request Unit Registers	7-241
7.4.2	Lockstep CPU Configuration	7-251
7.4.2.1	Logic Monitor Control Registers	7-252
7.4.3	Die Temperature Measurement	7-257
7.4.3.1	Die Temperature Sensor Register	7-258
7.4.4	Watchdog Timers	7-263
7.4.4.1	Watchdog Timers Overview	7-263
7.4.4.2	Features of the Watchdog Timers	7-265
7.4.4.3	The Endinit Functions	7-265
7.4.4.4	Timer Operation	7-272
7.4.4.5	Watchdog Timer Registers	7-276
7.4.5	Emergency Stop Output Control	7-293
7.4.5.1	Port Triggered Emergency Stop	7-293
7.4.5.2	SMU Event Triggered Emergency Stop	7-294
7.4.5.3	Emergency Stop Register	7-295
7.4.6	LBIST Support	7-297
7.4.6.1	LBIST Control Register	7-297
7.4.7	Global Overlay Controls	7-301
7.4.7.1	Global Overlay Control	7-302
7.4.8	Miscellaneous System Control	7-307
7.4.8.1	System Control Register	7-307
7.4.8.2	Identification Registers	7-309
7.4.8.3	SCU Access Restriction Registers	7-314
7.4.9	SCU Register Address	7-316
7.4.10	SCU Kernel Registers	7-316
8	Memory Test Unit (MTU)	8-1
8.1	Memory Content Initialization	8-1
8.1.1	Non-Security Applications	8-1
8.1.2	Security Applications	8-2
8.2	Safety Notifications	8-2
8.3	Memory Test Unit (MTU) Kernel Registers	8-2
8.3.1	Descriptions	8-3
8.3.2	MTU Register Overview	8-13
8.4	Memory Controllers	8-15
8.4.1	Control and Status Interfaces	8-16
8.4.1.1	Direct CPU Interface	8-16
8.4.2	Registers	8-17
8.4.2.1	MBIST/ECC Registers	8-17
8.4.3	Safety Features	8-35
8.4.4	Operation Modes	8-38

Table of Contents

8.4.4.1	Starting a Memory Test Sequence (example)	8-38
8.4.4.2	Getting Detailed Memory Test Results	8-38
8.4.4.3	Dumping Fail Bitmap	8-38
8.4.4.4	Filling a Memory with Defined Contents	8-39
8.4.4.5	Reading a Single Memory Location	8-39
8.4.4.6	Writing to a Single Memory Location	8-40
8.4.5	Memory Controller Register Addresses	8-41
8.4.6	Memory Controller Register Overview	8-44
8.5	ECC Implementation	8-45
8.5.1	ECC	8-45
8.5.1.1	ECC Codes	8-45
8.5.2	Address Error Detection	8-60
8.6	Implementation Section	8-60
8.6.1	Memory Control Register Implementation	8-60
8.6.1.1	MEMTEST Implementation	8-61
8.6.1.2	MEMMAP Implementation	8-67
8.6.1.3	MEMSTAT Implementation	8-70
8.6.1.4	Memory Controller Instances	8-73
9	Safety Management Unit (SMU)	9-1
9.1	Introduction	9-1
9.2	Features	9-3
9.3	Functional Overview	9-5
9.4	Functional Description	9-6
9.4.1	Reset Types	9-6
9.4.2	Interfaces Overview	9-8
9.4.2.1	Interfaces to SCU	9-8
9.4.2.2	Interfaces to the Interrupt Router	9-8
9.4.2.3	Interface to the Ports (Error Pin)	9-8
9.4.2.4	Interface to the Safety Flip-Flop Safety Mechanism	9-12
9.4.3	SMU Integration Guidelines	9-13
9.4.4	Alarm Mapping	9-14
9.4.4.1	Pre-alarm Definition	9-14
9.4.4.2	Pre-alarm Signals	9-14
9.4.4.3	Non-compliant Alarms	9-22
9.4.4.4	Internal SMU Alarms	9-23
9.4.4.5	Alarm Signals	9-24
9.4.5	Alarm Handling	9-41
9.4.5.1	Alarm protocol	9-41
9.4.5.2	Alarm Configuration	9-41
9.4.5.3	Alarm operation	9-43
9.4.5.4	Alarm Status Registers	9-44
9.4.5.5	Alarm Debug Registers	9-44

Table of Contents

9.4.5.6	Port Emergency Stop	9-45
9.4.5.7	Recovery Timer	9-45
9.4.5.8	Watchdog Alarms	9-46
9.4.6	SMU Control Interface	9-48
9.4.7	SMU state machine	9-50
9.4.8	Fault Signaling Protocol (FSP)	9-52
9.4.8.1	Introduction	9-52
9.4.8.2	Bi-stable fault signaling protocol	9-54
9.4.8.3	Time switching protocol	9-55
9.4.8.4	FSP Fault State	9-56
9.4.8.5	FSP and SMU START State	9-57
9.4.9	OCDS Trigger Bus (OTGB) Interface	9-58
9.4.10	Register Properties	9-59
9.4.10.1	Register Write Protection	9-59
9.4.10.2	Safety Flip-Flops	9-60
9.5	SMU Module Registers	9-62
9.5.1	System Registers description	9-68
9.5.2	SMU Configuration Registers	9-79
9.5.3	SMU Alarm Configuration Registers	9-97
9.5.4	SMU Alarm Configuration Registers (Fault Signaling Protocol)	9-108
9.5.5	SMU Alarm Status Registers	9-115
9.5.6	SMU Alarm Debug Registers	9-116
9.5.7	SMU Special Safety Registers: Register Monitor	9-117
10	Program Memory Unit (PMU)	10-1
10.1	Generic Feature List	10-1
10.2	PMU Configuration of TC26x B-Step	10-2
10.2.1	Features of the BootROM	10-4
10.2.2	Features of the Program and Data Flash	10-4
10.2.2.1	Program Flash Features:	10-4
10.2.2.2	Data Flash Features	10-5
10.3	BootROM	10-6
10.4	Tuning Protection	10-6
10.5	Flash	10-7
10.5.1	Definition of Terms	10-7
10.5.2	Flash Structure	10-8
10.5.2.1	PFlash	10-8
10.5.2.2	DFlash of PMU0	10-10
10.5.3	Flash Read Access	10-11
10.5.3.1	Read Ports	10-12
10.5.3.2	DFlash, BootROM Read Port	10-13
10.5.3.3	Configuring Flash Wait Cycles	10-13
10.5.3.4	Requested DFlash Read	10-14

Table of Contents

10.5.4	Flash Operations	10-14
10.5.4.1	Modes of Operation	10-15
10.5.4.2	Command Sequences	10-15
10.5.4.3	Command Sequence Definitions	10-16
10.5.4.4	Operation Suspend and Resume	10-23
10.5.4.5	Programming Voltage Selection	10-24
10.5.5	Protection	10-24
10.5.5.1	Master Specific Access Control	10-25
10.5.5.2	Register Access Control	10-26
10.5.5.3	Effective Flash Read Protection	10-26
10.5.5.4	Effective Flash Write Protection	10-27
10.5.5.5	Configuring Protection in the UCB	10-28
10.5.5.6	System Wide Effects of Flash Protection	10-36
10.5.6	Data Integrity and Safety	10-37
10.5.6.1	SRI ECC (Safe Fetch Path)	10-37
10.5.6.2	Flash ECC	10-37
10.5.6.3	Margin Checks	10-40
10.5.6.4	PMU and Flash Register Supervision	10-40
10.5.7	Interrupts and Traps	10-41
10.5.8	Reset and Startup	10-42
10.5.9	Power Reduction	10-42
10.6	Signaling to the Safety Management Unit (SMU)	10-43
10.7	Register Set	10-44
10.7.1	PMU Registers	10-44
10.7.1.1	PMU Identification	10-45
10.7.2	Flash Registers	10-47
10.7.2.1	Master Specific Access Control	10-50
10.7.2.2	Flash Identification Register	10-52
10.7.2.3	Flash Status	10-52
10.7.2.4	Flash Configuration Control	10-59
10.7.2.5	Flash Protection	10-63
10.7.2.6	Protection Configuration	10-67
10.7.2.7	Flash Read Buffer Configuration	10-79
10.7.2.8	Requested Read Interface	10-80
10.7.2.9	Flash ECC Access	10-82
10.7.2.10	Margin Check Control	10-85
10.7.2.11	Corrected Bits Address Buffer (CBAB)	10-88
10.7.2.12	Uncorrectable Bits Address Buffer (UBAB)	10-91
10.7.2.13	Direct Flash Communication	10-93
10.8	Application Hints	10-96
10.8.1	Changes with Respect to Auto Families Auto-NG/F/S/Max	10-96
10.8.2	Performing Flash Operations	10-97
10.8.3	EEPROM Emulation With DFlash	10-99

Table of Contents

10.8.3.1	Robust EEPROM Emulation	10-100
10.8.4	Handling Errors	10-101
10.8.4.1	Handling Errors During Operation	10-101
10.8.4.2	Handling Errors During Startup	10-105
10.8.5	Resets During Flash Operation	10-105
10.8.5.1	General Advice	10-106
10.8.5.2	Advice for EEPROM Emulation	10-106
10.8.6	ECC	10-107
10.8.7	Startup Tests of ECC Logic	10-108
10.8.7.1	Testing ECC Alarms and Error Flags	10-108
10.8.7.2	Testing the “ECC Monitor”	10-109
10.8.7.3	Testing the SMU Alarm of the “ECC Monitor”	10-110
10.8.7.4	Testing the “EDC Comparator”	10-110
10.8.7.5	General Advice for Startup Tests	10-110
11	Local Memory Unit (LMU)	11-1
11.1	Feature List	11-1
11.2	Local Memory (LMU SRAM)	11-1
11.3	Emulation Memory (EMEM)	11-1
11.3.1	EMEM Memory Read Buffers	11-2
11.3.2	Access to Emulation Device Register Space	11-3
11.4	Error Detection and Signalling	11-3
11.4.1	EMEM Read Error	11-3
11.4.2	EMEM Write Error	11-3
11.4.3	SRI write access data phase error	11-4
11.4.4	SRI access address phase error	11-4
11.5	Online Data Acquisition (OLDA) and its Overlay	11-4
11.6	Clock Control	11-4
11.7	LMU Register Protection	11-5
11.8	LMU Registers	11-6
12	Data Access Overlay (OVC)	12-1
12.1	Data Access Redirection	12-2
12.2	Target Memories	12-4
12.2.1	Online Data Acquisition (OLDA) Space	12-4
12.3	Overlay Memories	12-4
12.3.1	Local Memory	12-4
12.3.2	Emulation Memory	12-5
12.3.3	DSPR & PSPR Memory	12-5
12.4	Global Overlay Control	12-5
12.4.1	Global Overlay Control Synchronisation	12-6
12.5	Overlay Configuration Change	12-7
12.6	Access Protection, Attributes, Concurrent Matches	12-7
12.7	Overlay Control Registers	12-8

Table of Contents

12.7.1	Block control registers	12-9
12.8	Global overlay control registers	12-15
13	8-Bit Standby Controller (TC2x_SCR)	13-1
13.1	Features	13-2
13.2	XC800 Core	13-3
13.2.1	SFRs of the CPU	13-3
13.2.1.1	Stack Pointer (SP, 81 _H)	13-3
13.2.1.2	Data Pointer (DPTR, 82-3 _H)	13-3
13.2.1.3	Accumulator (ACC, E0 _H)	13-3
13.2.1.4	B Register (F0 _H)	13-3
13.2.1.5	Program Status Word (PSW, D0 _H)	13-3
13.2.1.6	Extended Operation Register (EO, A4 _H)	13-5
13.2.1.7	Power Control Register (PCON, 87 _H)	13-6
13.2.1.8	Interrupt Registers	13-6
13.2.2	SFRs of The Core Peripherals	13-7
13.2.2.1	Timer Registers	13-7
13.2.2.2	UART Registers	13-7
13.2.3	Instruction Timing	13-8
13.3	Memory Organization	13-15
13.3.1	Program Memory	13-16
13.3.2	Data Memory	13-16
13.3.2.1	Internal Data Memory	13-16
13.3.2.2	External Data Memory	13-17
13.3.3	Special Function Registers	13-18
13.3.3.1	Address Extension by Mapping	13-18
13.3.3.2	System Control Register 0	13-20
13.3.3.3	Address Extension by Paging	13-21
13.3.3.4	Page Register	13-23
13.3.3.5	Bit-Addressing	13-24
13.3.3.6	Bit Protection Scheme	13-25
13.3.4	Memory Control Unit	13-27
13.3.4.1	Active Memory Map Select	13-28
13.3.4.2	Program and XRAM Data Bus Control	13-28
13.3.4.3	Memory Protection Unit	13-29
13.3.4.4	Boot ROM Protection	13-29
13.3.5	Arbiter	13-30
13.3.5.1	SPI Slave Interface	13-30
13.3.5.2	Arbiter State Machine	13-32
13.4	TC2x_SCR Firmware	13-35
13.4.1	Overview	13-35
13.4.1.1	BootROM location	13-35
13.4.1.2	Boot Mode Option	13-35

Feature Discontinued

Table of Contents

13.4.2	OCDS Mode	13-37
13.4.2.1	Features	13-37
13.4.2.2	OCDS Basic Understanding	13-37
13.4.2.3	OCDS Monitor Firmware	13-38
13.4.2.4	Monitor processed commands	13-44
13.4.2.5	Important information for Debugger	13-50
13.5	System Control Unit	13-59
13.5.1	Clock System and Control	13-59
13.5.1.1	CCU Register	13-61
13.5.2	Reset Control	13-63
13.5.2.1	Types of Reset	13-63
13.5.2.2	Module Reset Behavior	13-66
13.5.2.3	Reset Register Description	13-67
13.5.3	Interrupt Management	13-70
13.5.3.1	Overview	13-70
13.5.3.2	Interrupt Node Assignment	13-72
13.5.3.3	NMI Event Flags Handling	13-73
13.5.3.4	Interrupt from/to CPUx	13-73
13.5.3.5	Interrupt Related Registers	13-74
13.5.4	Boot and Startup	13-76
13.5.4.1	Boot ROM Operating Mode	13-76
13.5.5	Power Management	13-79
13.5.5.1	Overview	13-79
13.5.5.2	Functional Description	13-79
13.5.5.3	Exit Standby Mode via TOS_SCR	13-80
13.5.5.4	Register Description	13-80
13.5.6	Flexible Peripheral Management	13-84
13.5.6.1	Peripheral Management Registers	13-85
13.5.7	XRAM Addressing Modes	13-85
13.5.7.1	Access to XRAM Using the DPTR (16-bit addressing Mode) ...	13-85
13.5.7.2	Access to XRAM Using the Register R0/R1 (8-bit addressing Mode) ...	13-86
13.5.8	Module Suspend Control	13-87
13.5.9	JTAG ID	13-87
13.5.10	General Pin Control	13-88
13.5.10.1	Input Pin Function Selection	13-88
13.5.11	Miscellaneous Control	13-95
13.5.11.1	Bit Protection Register	13-95
13.5.11.2	System Control Registers	13-96
13.5.11.3	Standby Control Register	13-97
13.5.11.4	Spare Register	13-97
13.5.12	Register Mapping	13-99
13.6	Watchdog Timer	13-102

Table of Contents

13.6.1	Overview	13-102
13.6.2	System Information	13-103
13.6.2.1	Reset effects	13-103
13.6.2.2	Clocking Configuration	13-103
13.6.2.3	Interrupt Events and Assignment	13-103
13.6.2.4	Module Suspend Control	13-103
13.6.3	Functional Description	13-104
13.6.4	Registers Description	13-107
13.6.4.1	Watchdog Timer Registers	13-107
13.7	Interrupt System	13-111
13.7.1	Interrupt Sources	13-111
13.7.1.1	Interrupt Source and Vector	13-118
13.7.1.2	Interrupt Source and Priority	13-119
13.7.2	Interrupt Structure	13-120
13.7.2.1	Interrupt Structure 1	13-121
13.7.2.2	Interrupt Structure 2	13-121
13.7.3	Interrupt Handling	13-122
13.7.4	Interrupt Response Time	13-123
13.7.5	Registers Description	13-126
13.7.5.1	Interrupt Node Enable Registers	13-127
13.7.5.2	External Interrupt Control Registers	13-130
13.7.5.3	Interrupt Flag Registers	13-136
13.7.5.4	Interrupt Priority Registers	13-141
13.7.6	Interrupt Flag Overview	13-143
13.8	Debug System	13-147
13.8.1	Overview	13-147
13.8.1.1	Features	13-147
13.8.1.2	Components of the Debug System	13-147
13.8.2	Product Specific Information	13-148
13.8.2.1	Pinning	13-148
13.8.2.2	Clocking Configuration	13-148
13.8.2.3	Interrupt Events and Assignment	13-148
13.8.2.4	Debug Suspend Control	13-149
13.8.2.5	JTAG ID	13-150
13.8.3	Functional Overview of the Debug System	13-150
13.8.3.1	Recognizing Debug-events	13-150
13.8.3.2	Activating the Monitor Program	13-151
13.8.3.3	Debug Suspend Control	13-152
13.8.3.4	Running the Monitor	13-152
13.8.3.5	Returning to the User Program	13-152
13.8.3.6	Single Step Execution	13-152
13.8.4	JTAG Module	13-153
13.8.4.1	JTAG ID Register	13-154

Table of Contents

13.8.5	Interface Block	13-154
13.8.6	Program Execution Control	13-156
13.8.6.1	Entering Monitor Mode	13-157
13.8.6.2	Exiting Monitor Mode	13-158
13.8.6.3	Single Step Execution	13-159
13.8.7	Breakpoint Generation Module	13-159
13.8.7.1	Generating Hardware Breakpoints	13-160
13.8.7.2	Processing External Breaks	13-164
13.8.7.3	Processing Software Breakpoints	13-165
13.8.8	Reactions on Breakpoints without Monitor Entry	13-165
13.8.8.1	Triggering a NMI request	13-165
13.8.9	Important Notes about Break Event Processing	13-165
13.8.9.1	Calculating the Return Address	13-166
13.8.9.2	Specific Behavior while XC800 is in Debug Mode	13-168
13.8.10	NMI Request and Control by OCDS	13-169
13.8.10.1	NMI request from OCDS	13-169
13.8.10.2	OCDS NMI Service Routine Handling	13-170
13.8.10.3	General NMI control by OCDS	13-170
13.8.11	NMI-mode priority over Debug mode	13-171
13.8.12	Registers Description	13-172
13.8.12.1	Control and Status Registers	13-174
13.8.12.2	Data Transfer Registers	13-180
13.8.12.3	Hardware Breakpoint Registers	13-181
13.8.12.4	Monitor Work Registers	13-184
13.9	General Purpose I/O Ports and Peripheral I/O Lines (Ports)	13-187
13.9.1	Basic Port Operation	13-187
13.9.2	Description Scheme for the Port IO Functions	13-191
13.9.3	Port Register Overview	13-193
13.9.3.1	Port Input/Output Control Registers	13-196
13.9.3.2	Pad Driver Mode Register	13-206
13.9.3.3	Port Output Register	13-210
13.9.3.4	Port Output Modification Register	13-211
13.9.3.5	Port Output Modification Set Register	13-213
13.9.3.6	Port Output Modification Set Registers x	13-213
13.9.3.7	Port Output Modification Clear Register	13-216
13.9.3.8	Port Output Modification Clear Registers x	13-217
13.9.3.9	Port Input Register	13-219
13.9.4	T2X_SCR functions in Port 00	13-220
13.9.4.1	T2X_SCR Port 00 Configuration	13-220
13.9.4.2	Port 00 Function Table	13-220
13.9.4.3	Port 00 Registers	13-225
13.9.5	T2X_SCR functions in Port 02	13-227
13.9.5.1	T2X_SCR Port 02 Configuration	13-227

Table of Contents

13.9.5.2	Port 02 Function Table	13-227
13.9.5.3	Port 02 Registers	13-234
13.10	Timer 0 and Timer 1	13-236
13.10.1	Overview	13-236
13.10.2	System Information	13-236
13.10.2.1	Pinning	13-236
13.10.2.2	Clocking Configuration	13-237
13.10.2.3	Interrupt Events and Assignment	13-238
13.10.3	Basic Timer Operations	13-238
13.10.4	Timer Modes	13-239
13.10.4.1	Mode 0	13-240
13.10.4.2	Mode 1	13-241
13.10.4.3	Mode 2	13-242
13.10.4.4	Mode 3	13-243
13.10.5	Register Map	13-244
13.10.6	Register Description	13-245
13.11	Timer 2	13-250
13.11.1	Overview	13-250
13.11.2	System Information	13-250
13.11.2.1	Pinning	13-250
13.11.2.2	Clocking Configuration	13-252
13.11.2.3	Interrupt Events and Assignment	13-253
13.11.2.4	Module Suspend Control	13-253
13.11.3	Basic Timer Operations	13-253
13.11.4	Auto-Reload Mode	13-253
13.11.4.1	Up/Down Count Disabled	13-254
13.11.4.2	Up/Down Count Enabled	13-255
13.11.5	Capture Mode	13-257
13.11.6	Count Clock	13-258
13.11.7	External Interrupt Function	13-259
13.11.8	Register Map	13-260
13.11.9	Register Description	13-261
13.11.9.1	Mode Register	13-261
13.11.9.2	Control Register	13-263
13.11.9.3	Timer 2 Reload Capture Register	13-265
13.11.9.4	Timer 2 Control Register	13-266
13.12	T2CCU	13-267
13.12.1	Features	13-267
13.12.2	Requirements of Timer 2	13-268
13.12.3	Capture/Compare Timer of T2CCU	13-269
13.12.3.1	Synchronized Timer Starts	13-270
13.12.3.2	Cascading Timers For Flexible Count Rate	13-270
13.12.3.3	Dead-time with CCT Timer	13-271

Table of Contents

13.12.4	T2CCU Capture/Compare Channels	13-273
13.12.5	Capture/Compare Operation	13-275
13.12.6	Compare Function	13-276
13.12.6.1	Compare Mode 0	13-278
13.12.6.2	Compare Mode 1	13-280
13.12.6.3	Concurrent Compare Mode	13-281
13.12.6.4	Using Interrupts in Combination with the Compare Function ..	13-283
13.12.7	Capture Function	13-283
13.12.8	T2CCU Registers	13-285
13.12.8.1	Register Mapping	13-286
13.12.8.2	T2CCU Registers Description	13-288
13.12.9	Implementation Details of T2CCU	13-302
13.12.9.1	Interfaces of the T2CCU	13-302
13.12.9.2	Design Options	13-303
13.13	Real-Time Clock	13-305
13.13.1	Overview	13-305
13.13.2	System Information	13-305
13.13.2.1	Interrupt Events and Assignment	13-305
13.13.2.2	Module Suspend Control	13-305
13.13.3	Clock Source	13-306
13.13.4	Real Time Clock Operation	13-306
13.13.4.1	Periodic Wake-up Mode	13-306
13.13.5	Power Saving Mode Option	13-308
13.13.6	Real-Time Clock Registers	13-309
13.13.6.1	Register Mapping	13-309
13.13.6.2	Register Description	13-310
13.14	Universal Asynchronous Receiver/Transmitter	13-315
13.15	Overview	13-315
13.15.1	System Information	13-315
13.15.1.1	Pinning	13-315
13.15.1.2	Clocking Configuration	13-317
13.15.1.3	Interrupt Events and Assignment	13-317
13.15.2	UART Modes	13-318
13.15.2.1	Mode 0, 8-Bit Shift Register, Fixed Baud Rate	13-318
13.15.2.2	Mode 1, 4-Bit UART, Variable Baud Rate	13-319
13.15.2.3	Mode 2, 9-Bit UART, Fixed Baud Rate	13-321
13.15.2.4	Mode 3, 7-Bit UART, Variable Baud Rate	13-321
13.15.3	Multiprocessor Communication	13-323
13.15.4	Baud Rate Generation	13-324
13.15.4.1	Fixed Clock	13-324
13.15.4.2	UART Baud-rate Generator	13-324
13.15.5	LIN Support in UART	13-327
13.15.5.1	LIN Protocol	13-327

Table of Contents

13.15.5.2	LIN Header Transmission	13-329
13.15.5.3	Automatic Synchronization to the Host	13-329
13.15.5.4	Initialization of Break/Synch Field Detection Logic	13-330
13.15.5.5	Baud Rate Range Selection	13-330
13.15.5.6	LIN Baud Rate Detection	13-332
13.15.6	Register Description	13-333
13.15.6.1	UART Registers	13-334
13.15.6.2	Baud-rate Generator Control and Status Registers	13-336
13.15.6.3	Baud-rate Generator Timer/Reload Registers	13-338
13.16	High-Speed Synchronous Serial Interface	13-341
13.16.1	Overview	13-341
13.16.2	General Operation	13-342
13.16.2.1	Operating Mode Selection	13-343
13.16.2.2	Full-Duplex Operation	13-345
13.16.2.3	Half-Duplex Operation	13-348
13.16.2.4	Continuous Transfers	13-349
13.16.2.5	Baud Rate Generation	13-350
13.16.2.6	Error Detection Mechanisms	13-351
13.16.3	Interrupts	13-354
13.16.4	Register Description	13-355
13.16.4.1	Configuration Register	13-356
13.16.4.2	Baud Rate Timer Reload Register	13-361
13.16.4.3	Transmitter Buffer Register	13-361
13.16.4.4	Receiver Buffer Register	13-362
13.17	Wake-up CAN (WCAN) Filter	13-363
13.17.1	Wake-up CAN Filter Implementation	13-364
13.17.1.1	Overview	13-364
13.17.1.2	WCAN Filter Features	13-365
13.17.1.3	Introduction	13-366
13.17.1.4	WCAN Module Control Registers	13-367
13.17.1.5	Wake-up Frame Configuration and Detection	13-379
13.17.1.6	Oscillator Trimming	13-381
13.17.1.7	CAN Message Timeout	13-383
13.17.1.8	Error Counter	13-384
13.17.2	CAN Functional Description	13-385
13.17.2.1	CAN Node Control	13-385
13.17.2.2	WCAN Kernel Registers	13-390
13.17.2.3	Miscellaneous Register	13-403
13.17.3	WCAN Module Implementation	13-405
13.17.3.1	Wake-up CAN Module Registers	13-405
13.17.3.2	Port and I/O Line Control	13-407
13.17.4	Revision history	13-409
13.18	TC2x_SCR Implementation	13-415

Table of Contents

13.18.1	TC2x_SCR and QSPI Interface	13-415
14	General Purpose I/O Ports and Peripheral I/O Lines (Ports)	14-1
14.1	Basic Port Operation	14-1
14.2	Description Scheme for the Port IO Functions	14-5
14.3	Port Register Description	14-7
14.3.1	Module Identification Register	14-12
14.3.2	Port Input/Output Control Registers	14-13
14.3.3	Pad Driver Mode Register	14-24
14.3.4	LVDS Pad Control Register	14-28
14.3.5	Pin Function Decision Control Register	14-32
14.3.6	Pin Controller Select Register	14-33
14.3.7	Port Output Register	14-36
14.3.8	Port Output Modification Register	14-38
14.3.9	Port Output Modification Set Register	14-40
14.3.10	Port Output Modification Set Registers x	14-41
14.3.11	Port Output Modification Clear Register	14-45
14.3.12	Port Output Modification Clear Registers x	14-46
14.3.13	Emergency Stop Register	14-50
14.3.14	Port Input Register	14-51
14.3.15	Access Protection Registers	14-53
14.4	Port 00	14-56
14.4.1	Port 00 Configuration	14-56
14.4.2	Port 00 Function Table	14-56
14.4.3	Port 00 Registers	14-67
14.4.3.1	Port 00 Output Register	14-68
14.4.3.2	Port 00 Output Modification Register	14-68
14.4.3.3	Port 00 Output Modification Set Register	14-68
14.4.3.4	Port 00 Output Modification Set Register 12	14-68
14.4.3.5	Port 00 Output Modification Clear Register	14-68
14.4.3.6	Port 00 Output Modification Clear Register 12	14-68
14.4.3.7	Port 00 Input/Output Control Register 12	14-69
14.4.3.8	Port 00 Input Register	14-70
14.4.3.9	Port 00 Pad Driver Mode 1 Register	14-71
14.4.3.10	Port 00 Emergency Stop Register	14-71
14.4.3.11	Port 00 Pin Controller Select Register	14-72
14.5	Port 02	14-74
14.5.1	Port 02 Configuration	14-74
14.5.2	Port 02 Function Table	14-74
14.5.3	Port 02 Registers	14-82
14.5.3.1	Port 02 Output Register	14-83
14.5.3.2	Port 02 Output Modification Register	14-83
14.5.3.3	Port 02 Output Modification Set Register	14-83

Table of Contents

14.5.3.4	Port 02 Output Modification Set Register 8	14-83
14.5.3.5	Port 02 Output Modification Clear Register	14-83
14.5.3.6	Port 02 Output Modification Clear Register 8	14-84
14.5.3.7	Port 02 Input/Output Control Register 8	14-85
14.5.3.8	Port 02 Input Register	14-86
14.5.3.9	P02 Pad Driver Mode 1 Register	14-87
14.5.3.10	Port 02 Emergency Stop Register	14-87
14.5.3.11	Port 02 Pin Controller Select Register	14-88
14.6	Port 10	14-89
14.6.1	Port 10 Configuration	14-89
14.6.2	Port 10 Function Table	14-89
14.6.3	Port 10 Registers	14-94
14.6.3.1	Port 10 Output Register	14-94
14.6.3.2	Port 10 Output Modification Register	14-95
14.6.3.3	Port 10 Output Modification Set Register	14-95
14.6.3.4	Port 10 Output Modification Set Register 8	14-95
14.6.3.5	Port 10 Output Modification Clear Register	14-95
14.6.3.6	Port 10 Output Modification Clear Register 8	14-95
14.6.3.7	Port 10 Input/Output Control Register 8	14-96
14.6.3.8	Port 10 Input Register	14-97
14.6.3.9	Port 10 Pad Driver Mode 1 Register	14-98
14.6.3.10	Port 10 Emergency Stop Register	14-98
14.7	Port 11	14-99
14.7.1	Port 11 Configuration	14-99
14.7.2	Port 11 Function Table	14-99
14.7.3	Port 11 Registers	14-104
14.7.3.1	Port 11 Output Register	14-105
14.7.3.2	Port 11 Output Modification Register	14-105
14.7.3.3	Port 11 Output Modification Set Register	14-106
14.7.3.4	Port 11 Output Modification Set Register 0	14-106
14.7.3.5	Port 11 Output Modification Set Register 4	14-106
14.7.3.6	Port 11 Output Modification Set Register 8	14-106
14.7.3.7	Port 11 Output Modification Set Register 12	14-106
14.7.3.8	Port 11 Output Modification Clear Register	14-106
14.7.3.9	Port 11 Output Modification Clear Register 0	14-106
14.7.3.10	Port 11 Output Modification Clear Register 4	14-107
14.7.3.11	Port 11 Output Modification Clear Register 8	14-107
14.7.3.12	Port 11 Output Modification Clear Register 12	14-107
14.7.3.13	Port 11 Input/Output Control Register 0	14-108
14.7.3.14	Port 11 Input/Output Control Register 4	14-110
14.7.3.15	Port 11 Input/Output Control Register 8	14-112
14.7.3.16	Port 11 Input/Output Control Register 12	14-114
14.7.3.17	Port 11 Input Register	14-115

Table of Contents

14.7.3.18	Port 11 Pad Driver Mode 0 Register	14-116
14.7.3.19	Port 11 Pad Driver Mode 1 Register	14-117
14.7.3.20	Port 11 Emergency Stop Register	14-117
14.7.3.21	Port 11 Pin Controller Select Register	14-118
14.8	Port 13	14-120
14.8.1	Port 13 Configuration	14-120
14.8.2	Port 13 Function Table	14-121
14.8.3	Port 13 Registers	14-123
14.8.3.1	Port 13 Output Register	14-124
14.8.3.2	Port 13 Output Modification Register	14-124
14.8.3.3	Port 13 Output Modification Set Register	14-124
14.8.3.4	Port 13 Output Modification Clear Register	14-124
14.8.3.5	Port 13 Input Register	14-124
14.8.3.6	Port 13 Pad Driver Mode 0 Register	14-125
14.8.3.7	Port 13 LVDS Pad Control Register	14-127
14.8.3.8	Port 13 Emergency Stop Register	14-128
14.9	Port 14	14-129
14.9.1	Port 14 Configuration	14-129
14.9.2	Port 14 Function Table	14-129
14.9.3	Port 14 Registers	14-135
14.9.3.1	Port 14 Output Register	14-136
14.9.3.2	Port 14 Output Modification Register	14-136
14.9.3.3	Port 14 Output Modification Set Register	14-136
14.9.3.4	Port 14 Output Modification Set Register 8	14-136
14.9.3.5	Port 14 Output Modification Clear Register	14-136
14.9.3.6	Port 14 Output Modification Clear Register 8	14-136
14.9.3.7	Port 14 Input/Output Control Register 8	14-137
14.9.3.8	Port 14 Input Register	14-138
14.9.3.9	Port 14 Pad Driver Mode 1 Register	14-139
14.9.3.10	Port 14 Emergency Stop Register	14-140
14.10	Port 15	14-141
14.10.1	Port 15 Configuration	14-141
14.10.2	Port 15 Function Table	14-141
14.10.3	Port 15 Registers	14-146
14.10.3.1	Port 15 Output Register	14-147
14.10.3.2	Port 15 Output Modification Register	14-147
14.10.3.3	Port 15 Output Modification Set Register	14-147
14.10.3.4	Port 15 Output Modification Set Register 8	14-147
14.10.3.5	Port 15 Output Modification Clear Register	14-147
14.10.3.6	Port 15 Output Modification Clear Register 8	14-147
14.10.3.7	Port 15 Input/Output Control Register 8	14-148
14.10.3.8	Port 15 Input Register	14-149
14.10.3.9	Port 15 Pad Driver Mode 1 Register	14-150

Table of Contents

14.10.3.10	Port 15 Emergency Stop Register	14-150
14.11	Port 20	14-151
14.11.1	Port 20 Configuration	14-151
14.11.2	Port 20 Function Table	14-151
14.11.3	Port 20 Registers	14-158
14.11.3.1	Port 20 Output Register	14-159
14.11.3.2	Port 20 Output Modification Register	14-159
14.11.3.3	Port 20 Output Modification Set Register	14-159
14.11.3.4	Port 20 Output Modification Set Register 0	14-159
14.11.3.5	Port 20 Output Modification Set Register 4	14-160
14.11.3.6	Port 20 Output Modification Set Register 12	14-160
14.11.3.7	Port 20 Output Modification Clear Register	14-160
14.11.3.8	Port 20 Output Modification Clear Register 0	14-160
14.11.3.9	Port 20 Output Modification Clear Register 4	14-160
14.11.3.10	Port 20 Output Modification Clear Register 12	14-160
14.11.3.11	Port 20 Input/Output Control Register 0	14-161
14.11.3.12	Port 20 Input/Output Control Register 4	14-163
14.11.3.13	Port 20 Input/Output Control Register 12	14-165
14.11.3.14	Port 20 Input Register	14-166
14.11.3.15	Port 20 Pad Driver Mode 0 Register	14-167
14.11.3.16	Port 20 Pad Driver Mode 1 Register	14-168
14.11.3.17	Port 20 Emergency Stop Register	14-169
14.12	Port 21	14-170
14.12.1	Port 21 Configuration	14-170
14.12.2	Port 21 Function Table	14-171
14.12.3	Port 21 Registers	14-176
14.12.3.1	Port 21 Output Register	14-177
14.12.3.2	Port 21 Output Modification Register	14-177
14.12.3.3	Port 21 Output Modification Set Register	14-177
14.12.3.4	Port 21 Output Modification Clear Register	14-177
14.12.3.5	Port 21 Input Register	14-177
14.12.3.6	P21 LVDS Pad Control Register	14-178
14.12.3.7	Port 21 Emergency Stop Register	14-181
14.13	Port 22	14-182
14.13.1	Port 22 Configuration	14-182
14.13.2	Port 22 Function Table	14-183
14.13.3	Port 22 Registers	14-185
14.13.3.1	Port 22 Output Register	14-186
14.13.3.2	Port 22 Output Modification Register	14-186
14.13.3.3	Port 22 Output Modification Set Register	14-186
14.13.3.4	Port 22 Output Modification Clear Register	14-186
14.13.3.5	Port 22 Input Register	14-186
14.13.3.6	Port 22 Pad Driver Mode 0 Register	14-187

Table of Contents

14.13.3.7	Port 22 LVDS Pad Control Register	14-189
14.13.3.8	Port 22 Emergency Stop Register	14-190
14.14	Port 23	14-191
14.14.1	Port 23 Configuration	14-191
14.14.2	Port 23 Function Table	14-191
14.14.3	Port 23 Registers	14-195
14.14.3.1	Port 23 Output Register	14-196
14.14.3.2	Port 23 Input/Output Control Register 4	14-197
14.14.3.3	Port 23 Pad Driver Mode 0 Register	14-199
14.14.3.4	Port 23 Output Modification Register	14-200
14.14.3.5	Port 23 Output Modification Set Register	14-200
14.14.3.6	Port 23 Output Modification Set Register 4	14-200
14.14.3.7	Port 23 Output Modification Clear Register	14-200
14.14.3.8	Port 23 Output Modification Clear Register 4	14-200
14.14.3.9	Port 23 Input Register	14-200
14.14.3.10	Port 23 Emergency Stop Register	14-200
14.15	Port 32	14-201
14.15.1	Port 32 Configuration	14-201
14.15.2	Port 32 Function Table	14-201
14.15.3	Port 32 Registers	14-205
14.15.3.1	Port 32 Output Register	14-206
14.15.3.2	Port 32 Output Modification Register	14-206
14.15.3.3	Port 32 Input/Output Control Register 0	14-207
14.15.3.4	Port 32 Input/Output Control Register 4	14-209
14.15.3.5	Port 32 Pad Driver Mode 0 Register	14-211
14.15.3.6	Port 32 Output Modification Set Register	14-212
14.15.3.7	Port 32 Output Modification Set Register 0	14-212
14.15.3.8	Port 32 Output Modification Set Register 4	14-212
14.15.3.9	Port 32 Output Modification Clear Register	14-212
14.15.3.10	Port 32 Output Modification Clear Register 0	14-212
14.15.3.11	Port 32 Output Modification Clear Register 4	14-212
14.15.3.12	Port 32 Input Register	14-212
14.15.3.13	Port 32 Emergency Stop Register	14-213
14.16	Port 33	14-214
14.16.1	Port 33 Configuration	14-214
14.16.2	Port 33 Function Table	14-214
14.16.3	Port 33 Registers	14-223
14.16.3.1	Port 33 Emergency Stop Register	14-224
14.17	Port 40	14-225
14.17.1	Port 40 Configuration	14-225
14.17.2	Port 40 Function Table	14-226
14.17.3	Port 40 Registers	14-228
14.17.4	Port 40 Input/Output Control Registers	14-229

Table of Contents

14.17.5	Port 40 Pin Function Decision Control Register	14-231
14.17.6	Port 40 Pin Controller Select Register	14-233
15	Direct Memory Access (DMA)	15-1
15.1	What is new	15-1
15.2	Features	15-2
15.3	Block Diagram	15-4
15.4	Functional Description	15-6
15.4.1	Definition of Terms	15-6
15.4.2	DMA Principles	15-7
15.4.3	DMA Channel Functionality	15-8
15.4.3.1	Shadowed Source or Destination Address	15-8
15.4.3.2	DMA Channel Request Control	15-13
15.4.3.3	DMA Channel Operation Modes	15-14
15.4.3.4	DMA Service Requests	15-19
15.4.3.5	Channel Reset Operation	15-20
15.4.3.6	Channel Halt Operation	15-21
15.4.3.7	Transfer Count and Move Count	15-23
15.4.3.8	Circular Buffer	15-25
15.4.3.9	Address Counter	15-26
15.4.3.10	Flow Control	15-26
15.4.3.11	Double Buffering Operation	15-29
15.4.3.12	Linked Lists	15-37
15.4.3.13	DMA Linked List	15-38
15.4.3.14	Accumulated Linked List	15-40
15.4.3.15	Safe Linked List	15-40
15.4.3.16	Conditional Linked List	15-42
15.4.4	Transaction Control Engine	15-45
15.4.4.1	Error Conditions	15-47
15.4.5	Bus Switch, Bus Switch Priorities	15-48
15.4.6	DMA Module Priorities on On Chip Busses	15-51
15.4.6.1	On Chip Bus Access Rights, RMW support	15-51
15.4.6.2	On Chip Bus Master Interfaces	15-51
15.4.7	Pattern Detection	15-53
15.4.7.1	Pattern Compare Logic	15-53
15.4.7.2	Pattern Detection for 8-bit Data Width	15-54
15.4.7.3	Pattern Detection for 16-bit Data Width	15-56
15.4.7.4	Pattern Detection for 32-bit Data Width	15-59
15.4.8	DMA Configuration Interface	15-61
15.4.8.1	DMARAM Channel Control and Status Word	15-61
15.4.8.2	DMA Active Channel Write Back	15-61
15.4.8.3	DMA Active Channel Shadow Control	15-62
15.4.8.4	DMARAM Write Back During Linked List Execution	15-63

Table of Contents

15.4.9	Interrupt Service Requests	15-64
15.4.9.1	Channel Transfer Interrupt Service Request	15-66
15.4.9.2	Channel Pattern Detection Interrupt Service Request	15-66
15.4.9.3	Channel Wrap Buffer Interrupt Service Request	15-68
15.4.9.4	Transaction Request Lost Interrupt Service Request	15-69
15.4.9.5	Source and Destination Error Interrupt Service Requests	15-70
15.4.9.6	DMA Linked List Error Interrupt Service Request	15-71
15.5	Power Modes	15-72
15.5.1	Sleep Mode	15-72
15.6	Functional Safety Features	15-72
15.6.1	Access Protection	15-72
15.6.2	Data Integrity	15-73
15.6.2.1	DMARAM	15-73
15.6.2.2	DMA SRI Read and Write Data	15-74
15.7	Debug Features	15-74
15.7.1	Channel Suspend Mode	15-74
15.7.2	OCDS Trigger Bus (OTGB) Interface	15-75
15.7.3	MCDS Trace Interface	15-76
15.8	Register Description	15-77
15.8.1	DMA General Module Control Registers	15-86
15.8.2	DMA Access Protection Registers	15-90
15.8.3	DMA Sub-block Error Registers	15-92
15.8.4	DMA Sub-block Move Engine Registers	15-99
15.8.5	DMA Move Engine Active Channel Registers	15-108
15.8.6	DMA OCDS Registers	15-130
15.8.7	DMA Pattern Detection Registers	15-134
15.8.8	DMA Flow Control Registers	15-136
15.8.9	DMA Channel Hardware Resource Registers	15-137
15.8.10	DMA Channel Suspend Registers	15-139
15.8.11	DMA Transaction State Registers	15-141
15.8.12	DMA Transaction Control Set	15-144
15.9	Use Cases	15-156
16	Flexible CRC Engine (FCE)	16-1
16.1	Related documentation	16-2
16.2	Features	16-3
16.3	Operational overview	16-4
16.4	FCE Functional Description	16-5
16.4.1	Overview	16-5
16.4.2	CRC Operation	16-7
16.4.3	Register protection and monitoring methods	16-9
16.4.4	FCE interrupts	16-10
16.5	Interfaces of the FCE Module	16-12

Table of Contents

16.6	FCE Module Registers	16-13
16.6.1	System Registers description	16-17
16.6.2	CRC Kernel Control/Status Registers	16-24
16.7	Programming Guide	16-37
16.8	Properties of CRC code	16-40
17	Interrupt Router (IR)	17-1
17.1	Overview	17-1
17.2	Features	17-2
17.3	Service Request Nodes (SRN)	17-3
17.3.1	Service Request Control Registers	17-3
17.3.1.1	General Service Request Control Register Format	17-3
17.3.1.2	Changing the SRN configuration	17-7
17.3.1.3	Protection of the SRC Registers	17-7
17.3.1.4	Request Set and Clear Bits (SETR, CLRR)	17-8
17.3.1.5	Enable Bit (SRE)	17-8
17.3.1.6	Service Request Flag (SRR)	17-8
17.3.1.7	Type-Of-Service Control (TOS)	17-9
17.3.1.8	Service Request Priority Number (SRPN)	17-9
17.3.1.9	ECC Encoding (ECC)	17-10
17.3.1.10	Interrupt Trigger Overflow Bit (IOV)	17-11
17.3.1.11	Interrupt Trigger Overflow Clear Bit (IOVCLR)	17-11
17.3.1.12	SW Sticky Bit (SWS)	17-11
17.3.1.13	SW Sticky Clear Bit (SWSCLR)	17-11
17.4	Interrupt Control Unit (ICU)	17-12
17.4.1	ICU Control Registers	17-12
17.4.1.1	Latest Winning Service Request Register (LWSR)	17-13
17.4.1.2	Last Acknowledged Service Request Register (LASR)	17-15
17.4.1.3	Error Capture Register (ECR)	17-16
17.5	General Purpose Service Requests, Service Request Broadcast	17-17
17.5.1	General Purpose Service Requests (GPSRxy)	17-18
17.5.2	Service Request Broadcast Registers (SRBx)	17-18
17.6	System Registers	17-19
17.6.1	Register Access Protection (ACCEN1/0)	17-19
17.6.2	Kernel Reset Registers (KRST1/0, KRSTCLR)	17-20
17.6.3	Clock Control Register (CLC)	17-20
17.6.4	OCDS Control and Status Register (OCS)	17-20
17.7	Arbitration Process	17-21
17.7.1	Number of Clock Cycles per Arbitration Process	17-22
17.7.2	Service Request Acknowledge	17-23
17.7.3	Handling of detected ECC Errors	17-23
17.8	Usage of the TC26x B-Step Interrupt System	17-25
17.8.1	CPU to ICU Interface	17-25

Table of Contents

17.8.2	DMA to ICU Interface	17-25
17.8.3	Software-Initiated Interrupts	17-25
17.8.4	External Interrupts	17-26
17.9	Use Case Examples	17-26
17.9.1	Use Case Example Interrupt Handler	17-27
17.10	Module Implementation	17-29
17.10.1	Characteristics of TC26x B-Step Interrupt Router Module	17-29
17.10.2	Mapping of TC26x B-Step Module Service Request Triggers to SRNs	17-29
17.10.2.1	Mapping of Service Request Control Registers	17-30
17.10.2.2	Interrupts related to the Debug Reset	17-31
17.10.2.3	Timing characteristics of Service Request Trigger Signals	17-31
17.11	Interrupt Router System and Module Registers	17-33
17.11.1	System and ICU Control Registers	17-37
17.12	OTGM Registers	17-44
17.12.1	Status and Control	17-44
17.12.2	IRQ MUX Control	17-45
17.12.3	Interrupt System Trace	17-48
17.12.4	MCDS Interface	17-49
17.13	Interrupt Router SRC Registers	17-51
18	System Timer (STM)	18-1
18.1	Overview	18-1
18.2	Operation	18-1
18.2.1	Compare Register Operation	18-3
18.2.2	Compare Match Interrupt Control	18-4
18.2.3	Using Multiple STMs	18-5
18.2.4	STM as Reset Trigger	18-5
18.3	STM Registers	18-5
18.3.1	Clock Control Register	18-8
18.3.2	Timer/Capture Registers	18-10
18.3.3	Compare Registers	18-13
18.3.4	Interrupt Registers	18-17
18.3.5	Interface Registers	18-20
19	Asynchronous/Synchronous Interface (ASCLIN)	19-1
19.1	Feature List	19-2
19.2	Overview	19-4
19.3	External Signals	19-5
19.4	User Interface	19-6
19.4.1	TxFIFO Overview	19-6
19.4.2	Using the TxFIFO	19-7
19.4.2.1	Standard ASC Mode	19-7
19.4.2.2	High Speed ASC Mode	19-9

Table of Contents

19.4.2.3	LIN Mode	19-11
19.4.2.4	SPI Mode	19-11
19.4.3	RxFIFO Overview	19-13
19.4.4	Using the RxFIFO	19-15
19.4.4.1	Standard ASC Mode	19-15
19.4.4.2	High Speed ASC Mode	19-17
19.4.4.3	LIN Mode	19-17
19.4.4.4	SPI Mode	19-19
19.4.5	RTS / CTS Handshaking	19-19
19.5	Clock System	19-20
19.5.1	Baud Rate Generation	19-20
19.5.2	Bit Timing Properties	19-21
19.6	Data Frame Configuration	19-24
19.7	Miscellaneous Configuration	19-24
19.8	Synchronous Mode	19-25
19.8.1	Baud Rate and Clock Generation	19-25
19.8.2	Data Frame Configuration	19-26
19.8.3	Slave Selects Configuration	19-26
19.8.4	Miscellaneous Configuration	19-26
19.9	LIN Support	19-27
19.9.1	LIN Watchdog	19-29
19.9.1.1	LIN Break, Wake, Stuck Handling	19-30
19.9.1.2	LIN Header and Response Timers	19-32
19.9.2	LIN Master Sequences	19-34
19.9.3	LIN Slave Sequences	19-38
19.9.4	Using the ENI and HO Bits	19-39
19.9.5	LIN Error Recovery	19-40
19.9.6	LIN Sleep and LIN Wake-Up	19-41
19.10	Auto Baud Rate Detection	19-42
19.11	Collision Detection	19-44
19.12	LIN Protocol Control	19-45
19.13	Interrupts	19-47
19.14	Digital Glitch Filter	19-50
19.15	Suspend, Sleep and Power-Off Behavior	19-51
19.15.1	OCDS Suspend	19-51
19.15.2	Sleep Mode	19-51
19.15.3	Disable Request (Power-Off)	19-52
19.16	Reset Behavior	19-52
19.17	Use Case Example ASC Interface	19-53
19.18	Kernel Registers	19-56
19.18.1	Kernel Registers	19-60
19.19	Implementation	19-106
19.19.1	BPI_FPI Module Registers	19-106

Table of Contents

19.19.1.1	System Registers	19-106
19.20	On-Chip Connections	19-114
19.21	ASC at CAN Support	19-120
20	Queued Synchronous Peripheral Interface (QSPI)	20-1
20.1	Feature List	20-1
20.2	Overview	20-3
20.2.1	External Signals	20-3
20.2.2	Operating Modes	20-4
20.2.3	Queue Support Overview	20-6
20.2.4	Architecture Overview	20-7
20.2.5	Three Wire Connection	20-8
20.3	Abstract Overview	20-9
20.4	Frequency Domains	20-10
20.5	Master Mode State Machine	20-11
20.5.1	Phases of one Communication Cycle	20-11
20.5.2	Configuration Extensions	20-17
20.5.3	Details of the Baud Rate and Phase Duration Control	20-18
20.5.4	Calculation of the Baud Rates and the Delays	20-20
20.5.5	State Diagram of Standard Communication Cycle	20-21
20.5.6	Expect Phase	20-24
20.5.7	External Slave Select Expansion	20-25
20.6	User Interface	20-26
20.6.1	Transmit and Receive FIFOs	20-27
20.6.1.1	Short Data Mode	20-30
20.6.1.2	Long Data Mode	20-32
20.6.1.3	Continuous Data Mode	20-35
20.6.1.4	Single Configuration - Multiple Frames Behavior	20-38
20.6.1.5	Big Endian Data Format	20-38
20.6.2	Loop-Back Mode	20-40
20.7	Interrupts	20-42
20.7.1	Slave Mode SLSI Interrupt	20-43
20.7.2	Interrupt Flags Behavior	20-44
20.7.3	TXFIFO Interrupt Generation	20-45
20.7.4	RXFIFO Interrupt Generation	20-49
20.7.5	DMA Transfer Example	20-52
20.8	Slave Mode	20-53
20.8.1	Shift Clock Phase and Polarity in Slave Mode	20-54
20.8.2	Shift Clock Monitoring	20-55
20.8.2.1	Baud Rate Error Detection	20-56
20.8.2.2	Spike Detection	20-56
20.8.2.3	Shift Clock Monitor Flags	20-57
20.8.3	Parity	20-58

Table of Contents

20.9	Kernel Registers	20-59
20.9.1	Kernel Registers	20-62
20.10	Operation Modes	20-94
20.10.1	OCDS Suspend	20-97
20.10.2	Sleep Mode	20-99
20.10.3	Disabling the QSPI	20-100
20.11	Reset Behavior	20-101
20.12	QSPI Module Implementation	20-102
20.12.1	Module Identification Registers	20-102
20.12.2	Interfaces of the QSPI Modules	20-102
20.12.3	On-Chip Connections	20-103
20.12.4	QSPI Related External Registers	20-104
20.12.4.1	BPI_FPI Module Registers	20-105
20.12.4.2	Interrupt Control Registers	20-113
20.13	High Speed Input Capture	20-114
20.13.1	HSIC Registers	20-117
20.13.1.1	Register Description	20-117
20.13.2	HSIC Implementation	20-118
20.13.2.1	Connection to the pins	20-119
21	High Speed Serial Link (HSSL)	21-1
21.1	Lower communication layers module (HSCT)	21-4
21.2	HSSL Protocol Definition	21-6
21.2.1	List of Abbreviations, Acronyms, and Term Definitions	21-6
21.2.2	Frame Types	21-6
21.2.2.1	Frame and Payload Lengths	21-8
21.2.2.2	Data Types	21-9
21.2.2.3	Cyclic Redundancy Check Field - CRC	21-10
21.2.2.4	Header Structure	21-11
21.2.3	Single and Block Transfers	21-13
21.2.4	Streaming Interface	21-13
21.2.5	Sliding Window Protocol	21-13
21.2.6	Error Management	21-15
21.2.7	Shift Direction	21-17
21.3	HSSL Implementation	21-18
21.4	Overview	21-18
21.4.1	HSSL Module Operation	21-19
21.4.1.1	Frame Transmission Priorisation	21-20
21.4.1.2	Received Frame Management and Command Execution	21-21
21.4.2	HSSL Channel Architecture	21-24
21.4.3	Acknowledge Responses	21-26
21.4.4	Cross Dependencies Between the Frame Types	21-27
21.4.5	Command Timeout	21-29

Table of Contents

21.4.5.1	Command Timeout Operation	21-29
21.4.6	Stream Timeout	21-31
21.4.6.1	Stream Timeout Operation	21-33
21.4.7	Data FIFOs of the Streaming Channel 2	21-35
21.5	Modes of Operation	21-36
21.6	Interrupts	21-39
21.7	Operating a Command Channel	21-40
21.7.1	Initiating a Single Write Command	21-40
21.7.2	Initiating a Single Read Command	21-40
21.7.3	Initiating a Single Trigger Command	21-40
21.7.4	DMA Operated Command Queues	21-40
21.7.5	Receiver Error Handling	21-41
21.7.5.1	Timeout Error	21-41
21.7.5.2	Transaction Tag Error	21-42
21.7.6	Global Error	21-42
21.8	Memory Block Transfer Modes of the Stream Channel	21-42
21.9	HSSL Reset	21-45
21.10	OCDS SRI / SPB Master Suspend	21-45
21.11	OCDS Trigger Sets	21-46
21.12	Access Protection	21-48
21.13	Kernel Registers	21-52
21.13.1	Global Registers	21-54
21.13.2	Channel.Flags Registers	21-58
21.13.3	Channel.Initiator Registers	21-76
21.13.4	Channel.Target Registers	21-80
21.13.5	Initiator Stream Registers	21-84
21.13.6	Target Stream Registers	21-86
21.13.7	Access Protection Registers	21-88
21.14	Module Implementation	21-91
21.14.1	BPI_SPB Module Registers	21-91
21.14.1.1	System Registers	21-91
21.15	High Speed Communication Tunnel (HSCT)	21-99
21.15.1	Overview	21-99
21.15.1.1	Features	21-100
21.15.2	Functional Description	21-100
21.15.2.1	Introduction	21-100
21.15.2.2	Physical Layer	21-101
21.15.2.3	Electrical Characteristics Based on LVDS for Reduced Link trace length	21-105
21.15.2.4	Protocol Layer	21-116
21.15.3	Use Cases	21-128
21.15.3.1	MC to ASIC	21-128
21.15.3.2	MC to FPGA	21-128

Table of Contents

21.15.3.3	MC to MC	21-128
21.15.4	HSCT Register Description	21-128
21.15.4.1	Registers Definition	21-129
21.15.5	Suspend, Sleep and Power-Off Behavior	21-165
21.15.5.1	OCDS Suspend	21-165
21.15.5.2	Sleep Mode	21-166
21.15.5.3	Disable Request (Power-Off)	21-166
21.15.6	References	21-166
22	Micro Second Channel (MSC)	22-1
22.1	MSC Kernel Description	22-3
22.1.1	Overview	22-3
22.1.2	Downstream Channel	22-5
22.1.2.1	Frame Formats and Definitions	22-6
22.1.2.2	Shift Register Operation	22-13
22.1.2.3	External Signal Injection	22-16
22.1.2.4	Transmission Modes	22-17
22.1.2.5	Downstream Counter and Enable Signals	22-22
22.1.2.6	Baud Rate	22-23
22.1.2.7	Abort of Frames	22-23
22.1.3	Upstream Channel	22-24
22.1.3.1	Data Frames	22-25
22.1.3.2	Parity Checking	22-25
22.1.3.3	Data Reception	22-26
22.1.3.4	Baud Rate	22-28
22.1.3.5	Spike Filter	22-29
22.1.3.6	Upstream Timer	22-30
22.1.4	I/O Control	22-31
22.1.4.1	Downstream Channel Output Control	22-31
22.1.4.2	Upstream Channel	22-34
22.1.5	MSC Interrupts	22-35
22.1.5.1	Data Frame Interrupt	22-36
22.1.5.2	Command Frame Interrupt	22-36
22.1.5.3	Time Frame Finished Interrupt	22-37
22.1.5.4	Receive Data Interrupt	22-38
22.1.5.5	Interrupt Request Compressor	22-39
22.2	ABRA (Asynchronous Baud Rate Adjustment Block)	22-40
22.2.1	Overview	22-40
22.2.2	Timings Issues	22-42
22.2.3	Adjusting the Passive Phase of a Frame	22-42
22.2.4	Jitter of the Downstream Frames	22-44
22.2.4.1	Jitter in Active Phase Clock Mode	22-44
22.2.4.2	Jitter in Continuous Clock Mode	22-45

Table of Contents

22.2.5	Interrupt Position with the ABRA Block	22-46
22.2.6	Configuring the ABRA block	22-47
22.2.7	Implementation Issues	22-48
22.2.8	ABRA Disable, Sleep and Suspend Behavior	22-49
22.2.8.1	Disable and Sleep Behavior	22-49
22.2.8.2	OCDS Suspend Behavior	22-49
22.3	MSC Kernel Registers	22-50
22.3.1	Module Identification Register	22-53
22.3.2	Status and Control Registers	22-54
22.3.3	Data Registers	22-81
22.3.4	Extension Registers	22-85
22.3.5	Asynchronous Block Registers	22-88
22.4	MSC Module Implementation	22-92
22.4.1	Module Identification Registers	22-92
22.4.2	BPI_FPI Module Registers (Single Kernel Configuration)	22-93
22.4.2.1	System Registers	22-93
22.4.3	Interface Connections of the MSC Module	22-101
22.4.4	MSC Module-Related External Registers	22-102
22.4.5	Clock Control	22-103
22.4.5.1	Clock Control Without the ABRA Block	22-104
22.4.5.2	Clock Control when using the ABRA Block	22-106
22.4.5.3	Fractional Divider Register	22-108
22.4.6	Port Control	22-109
22.4.7	On-Chip Connections	22-110
22.4.7.1	EMGSTOPMSC Signal (from SCU)	22-110
22.4.7.2	Interrupt Service Requests	22-110
22.4.7.3	Connections to Ports / Pins	22-110
22.4.7.4	GTM Connections	22-111
22.5	Use Case Example Micro Second Channel (MSC)	22-112
23	Controller Area Network Controller (MultiCAN+)	23-1
23.1	CAN Basics	23-2
23.1.1	Addressing and Bus Arbitration	23-2
23.1.2	CAN Frame Formats	23-3
23.1.3	CAN Frame Types	23-3
23.1.3.1	Data Frames	23-3
23.1.3.2	Remote Frames	23-7
23.1.3.3	Error Frames	23-7
23.1.3.4	Overload Frame	23-8
23.1.4	The Nominal Bit Time	23-9
23.1.4.1	CAN FD bit timing	23-9
23.1.5	Error Detection and Error Handling	23-10
23.2	Overview	23-12

Table of Contents

23.2.1	Features List	23-12
23.3	CAN Flexible Data-Rate (CAN FD)	23-15
23.3.1	Transmitter Delay Compensation	23-15
23.4	MultiCAN+ Kernel Functional Description	23-17
23.4.1	Module Structure	23-17
23.4.2	Clock Control	23-19
23.4.3	Port Input Control	23-24
23.4.4	OCDS Suspend	23-24
23.4.5	OCDS Trigger Bus (OTGB) Interface	23-25
23.4.6	CAN Node Control	23-27
23.4.6.1	Bit Timing Unit	23-28
23.4.6.2	Bitstream Processor	23-30
23.4.6.3	Error Handling Unit	23-30
23.4.6.4	CAN Frame Counter	23-31
23.4.6.5	Node Timing Functions	23-32
23.4.6.6	CAN Node Interrupts	23-34
23.4.7	Message Object List Structure	23-36
23.4.7.1	Basics	23-36
23.4.7.2	List of Unallocated Elements	23-37
23.4.7.3	Connection to the CAN Nodes	23-37
23.4.7.4	List Command Panel	23-38
23.4.8	CAN Node Analyzer Mode	23-40
23.4.8.1	Analyzer Mode	23-40
23.4.8.2	Loop-Back Mode	23-41
23.4.8.3	Bit Timing Analysis	23-42
23.4.9	Message Acceptance Filtering	23-43
23.4.9.1	Receive Acceptance Filtering	23-43
23.4.9.2	Transmit Acceptance Filtering	23-45
23.4.10	Message Postprocessing	23-46
23.4.10.1	Message Object Interrupts	23-46
23.4.10.2	Pending Messages	23-48
23.4.11	Message Object Data Handling	23-50
23.4.11.1	Frame Reception	23-50
23.4.11.2	Frame Transmission	23-53
23.4.12	Message Object Functionality	23-56
23.4.12.1	Standard Message Object	23-56
23.4.12.2	Single Data Transfer Mode	23-56
23.4.12.3	Single Transmit Trial	23-56
23.4.12.4	Message Object Format (Classical CAN & CAN FD)	23-57
23.4.12.5	Message Object FIFO Structure	23-57
23.4.12.6	Receive FIFO	23-60
23.4.12.7	Transmit FIFO	23-60
23.4.12.8	Gateway Mode	23-61

Table of Contents

23.4.12.9	Foreign Remote Requests	23-64
23.4.12.10	CAN FD - 64 byte Messages	23-64
23.4.13	Measurement for Oscillator Calibration	23-65
23.5	Use Case Example MultiCAN+	23-67
23.6	MultiCAN+ Kernel Registers	23-71
23.6.1	Global Module Registers	23-77
23.6.2	CAN Node Registers	23-94
23.6.3	Message Object Registers	23-123
23.7	MultiCAN+ Module Implementation	23-149
23.7.1	Interfaces of the MultiCAN+ Module	23-149
23.7.2	MultiCAN+ Module External Registers	23-150
23.7.2.1	System Registers	23-152
23.7.3	MultiCAN+ Clock Interconnects With SCU	23-160
23.7.4	Module Clock Generation	23-162
23.7.4.1	Clock Selection	23-162
23.7.4.2	Fractional Divider	23-162
23.7.5	Port and I/O Line Control	23-166
23.7.5.1	Input/Output Function Selection in Ports	23-166
23.7.5.2	Node Receive Input Selection	23-166
23.7.5.3	CAN Transmit Trigger Inputs	23-167
23.7.5.4	Connections to Interrupt Router Inputs	23-167
23.7.5.5	Connections to General Timer Module (GTM) Inputs	23-168
23.7.6	Interrupt Control	23-170
23.7.7	MultiCAN+ Module Register Address Map	23-172
23.8	MultiCAN+ Soft Configuration	23-173
23.9	Revision history	23-174
24	Single Edge Nibble Transmission (SENT)	24-1
24.1	SENT Kernel Description	24-1
24.1.1	Overview	24-2
24.1.2	General Operation	24-4
24.1.2.1	Definitions	24-5
24.1.3	Standard SENT Operation	24-6
24.1.3.1	Frame Formats and Definitions	24-7
24.1.4	SPC Operation	24-13
24.1.4.1	Synchronous Transmission	24-13
24.1.4.2	Range Selection	24-13
24.1.4.3	ID Selection	24-14
24.1.4.4	Bidirectional Transmit Mode	24-15
24.1.4.5	SPC Timing	24-15
24.1.4.6	Abort of Frames	24-16
24.1.5	Baud Rate Generation	24-17
24.1.6	Error Detection Capabilities	24-19

Table of Contents

24.1.7	Digital Glitch Filter	24-20
24.1.8	Interrupts	24-21
24.1.9	Trigger Outputs	24-21
24.2	SENT Kernel Registers	24-22
24.2.1	Module Control	24-26
24.2.2	Channel Baud Rate Registers	24-30
24.2.3	Receiver Control and Status Registers	24-32
24.2.4	Input and Output Control	24-42
24.2.5	Receive Data Registers	24-46
24.2.6	SPC Control	24-50
24.2.7	Interrupt Control Registers	24-53
24.3	SENT Module Implementation	24-70
24.3.1	Interface Connections of the SENT Module	24-70
24.3.1.1	On-Chip Connections	24-71
24.3.1.2	Interrupt and DMA Controller Service Requests	24-71
24.3.2	SENT Module-Related External Registers	24-73
24.3.2.1	Port Control	24-73
24.3.3	BPI_FPI Module Registers	24-77
24.4	Revision History	24-84
25	FlexRay™ Protocol Controller (E-Ray)	25-1
25.1	E-Ray Kernel Description	25-1
25.2	Overview	25-2
25.3	Definitions	25-3
25.4	Block Diagram	25-3
25.5	Programmer's Model	25-6
25.5.1	Register Map	25-6
25.5.2	E-Ray Kernel Registers	25-8
25.5.2.1	Customer Registers	25-15
25.5.2.2	Special Registers	25-22
25.5.2.3	Service Request Registers	25-32
25.5.2.4	Communication Controller Control Registers	25-79
25.5.2.5	Communication Controller Status Registers	25-106
25.5.2.6	Message Buffer Control Registers	25-129
25.5.2.7	Message Buffer Status Registers	25-136
25.5.2.8	Identification Registers	25-156
25.5.2.9	Input Buffer	25-158
25.5.2.10	Output Buffer	25-169
25.6	Functional Description	25-187
25.6.1	Communication Cycle	25-187
25.6.1.1	Static Segment	25-187
25.6.1.2	Dynamic Segment	25-188
25.6.1.3	Symbol Window	25-188

Table of Contents

25.6.1.4	Network Idle Time (NIT)	25-188
25.6.1.5	Configuration of Network Idle Time (NIT) Start and Offset Correction Start.	25-188
25.6.2	Communication Modes	25-190
25.6.3	Clock Synchronization	25-190
25.6.3.1	Global Time	25-190
25.6.3.2	Local Time	25-190
25.6.3.3	Synchronization Process	25-191
25.6.3.4	External Clock Synchronization	25-192
25.6.4	Error Handling	25-193
25.6.4.1	Clock Correction Failed Counter	25-193
25.6.4.2	Passive to Active Counter	25-194
25.6.4.3	HALT Command	25-194
25.6.4.4	FREEZE Command	25-194
25.6.5	Communication Controller States	25-196
25.6.5.1	Communication Controller State Diagram	25-196
25.6.5.2	DEFAULT_CONFIG State	25-198
25.6.5.3	MONITOR_MODE	25-199
25.6.5.4	READY State	25-200
25.6.5.5	WAKEUP State	25-200
25.6.5.6	STARTUP State	25-205
25.6.5.7	Startup Time-outs	25-208
25.6.5.8	Path of leading Coldstart Node (initiating coldstart)	25-209
25.6.5.9	NORMAL_ACTIVE State	25-211
25.6.5.10	NORMAL_PASSIVE State	25-211
25.6.5.11	HALT State	25-212
25.6.6	Network Management	25-213
25.6.7	Filtering and Masking	25-213
25.6.7.1	Frame ID Filtering	25-214
25.6.7.2	Channel ID Filtering	25-214
25.6.7.3	Cycle Counter Filtering	25-215
25.6.7.4	FIFO Filtering	25-216
25.6.8	Transmit Process	25-217
25.6.8.1	Static Segment	25-217
25.6.8.2	Dynamic Segment	25-217
25.6.8.3	Transmit Buffers	25-217
25.6.8.4	Frame Transmission	25-218
25.6.8.5	NULL Frame Transmission	25-219
25.6.9	Receive Process	25-220
25.6.9.1	Frame Reception	25-220
25.6.9.2	NULL Frame reception	25-221
25.6.10	FIFO Function	25-221
25.6.10.1	Description	25-221

Table of Contents

25.6.10.2	Configuration of the FIFO	25-222
25.6.10.3	Access to the FIFO	25-223
25.6.11	Message Handling	25-223
25.6.11.1	Host access to Message RAM	25-223
25.6.11.2	Data Transfers between IBF / OBF and Message RAM	25-228
25.6.11.3	Minimum f_{CLC_ERAY}	25-234
25.6.11.4	FlexRay™ Protocol Controller access to Message RAM	25-238
25.6.12	Message RAM	25-239
25.6.12.1	Header Partition	25-241
25.6.12.2	Data Partition	25-244
25.6.12.3	ECC Check	25-245
25.6.13	Host Handling of Errors	25-248
25.6.13.1	Self-Healing	25-248
25.6.13.2	CLEAR_RAM Command	25-248
25.6.13.3	Temporary Unlocking of Header Section	25-248
25.7	Module Service Request	25-250
25.8	Restrictions	25-253
25.8.1	Message Buffers with the same Frame ID	25-253
25.8.2	Data Transfers between IBF / OBF and Message RAM	25-253
25.9	E-Ray Module Implementation	25-254
25.9.1	Interconnections of the E-Ray Module	25-254
25.9.2	Port Control and Connections	25-255
25.9.2.1	Input/Output Function Selection	25-255
25.9.3	On-Chip Connections	25-258
25.9.3.1	E-Ray Connections with IR	25-258
25.9.3.2	E-Ray Connections with SMU	25-258
25.9.3.3	E-Ray Connections with the External Request Unit of SCU	25-258
25.9.3.4	E-Ray Connections to GTM	25-259
25.9.3.5	E-Ray Connections with the External Clock Output of SCU	25-259
25.9.4	OCDS Trigger Bus (OTGB) Interface	25-259
25.9.5	OTGB E-Ray Registers	25-262
25.9.5.1	OCDS Trigger Bus (OTGB)	25-262
25.9.6	BPI_FPI Module Registers	25-263
25.9.7	Interrupt Registers	25-271
25.10	Revision History	25-280
26	Generic Timer Module (GTM)	26-1
26.1	Introduction	26-1
26.1.1	Overview	26-1
26.1.2	Document Structure	26-2
26.2	GTM Architecture	26-4
26.2.1	Overview	26-4
26.2.1.1	GTM Architecture Block Diagram	26-4

Table of Contents

26.2.1.2	ARU Data Word Description	26-5
26.2.1.3	GTM signal multiplex	26-7
26.2.2	GTM Interfaces	26-9
26.2.2.1	GTM Generic Bus Interface (AEI)	26-10
26.2.2.2	GTM Multi-master and multi-tasking support	26-12
26.2.3	ARU Routing Concept	26-12
26.2.3.1	Principle of data routing using ARU	26-14
26.2.3.2	ARU Round Trip Time	26-16
26.2.3.3	ARU Blocking Mechanism	26-16
26.2.4	GTM Clock and Time Base Management (CTBM)	26-17
26.2.4.1	GTM Clock and time base management architecture	26-17
26.2.5	GTM Interrupt Concept	26-19
26.2.5.1	Level interrupt mode	26-21
26.2.5.2	Pulse interrupt mode	26-24
26.2.5.3	Pulse-notify interrupt mode	26-25
26.2.5.4	Single-pulse interrupt mode	26-27
26.2.5.5	GTM Interrupt concentration method	26-29
26.2.6	GTM Software Debugger Support	26-29
26.2.6.1	Register behaviour in case of Software Debugger accesses	26-29
26.2.7	GTM Programming conventions	26-30
26.2.8	GTM TOP-Level Configuration Registers Overview	26-30
26.2.9	GTM TOP-Level Configuration Registers Description	26-31
26.2.9.1	Register GTM_REV	26-31
26.2.9.2	Register GTM_RST	26-32
26.2.9.3	Register GTM_CTRL	26-33
26.2.9.4	Register GTM_AEI_ADDR_XPT	26-34
26.2.9.5	Register GTM_IRQ_NOTIFY	26-35
26.2.9.6	Register GTM_IRQ_EN	26-36
26.2.9.7	Register GTM_IRQ_FORCINT	26-37
26.2.9.8	Register GTM_IRQ_MODE	26-39
26.2.9.9	Register GTM_BRIDGE_MODE	26-40
26.2.9.10	Register GTM_BRIDGE_PTR1	26-42
26.2.9.11	Register GTM_BRIDGE_PTR2	26-43
26.2.9.12	Register GTM_EIRQ_EN	26-44
26.2.9.13	Register GTM_TIMi_AUX_IN_SRC (i= 0...n)	26-46
26.3	Advanced Routing Unit (ARU)	26-47
26.3.1	Overview	26-47
26.3.2	Special Data Sources	26-47
26.3.3	ARU Access via AEI	26-47
26.3.3.1	Default ARU Access	26-47
26.3.3.2	Debug Access	26-49
26.3.4	ARU Interrupt Signals	26-49
26.3.5	ARU Configuration Registers Overview	26-49

Table of Contents

26.3.6	ARU Configuration Registers Description	26-50
26.3.6.1	Register ARU_ACCESS	26-50
26.3.6.2	Register ARU_DATA_H	26-53
26.3.6.3	Register ARU_DATA_L	26-54
26.3.6.4	Register ARU_DBG_ACCESS0	26-55
26.3.6.5	Register ARU_DBG_DATA0_H	26-55
26.3.6.6	Register ARU_DBG_DATA0_L	26-56
26.3.6.7	Register ARU_DBG_ACCESS1	26-56
26.3.6.8	Register ARU_DBG_DATA1_H	26-57
26.3.6.9	Register ARU_DBG_DATA1_L	26-58
26.3.6.10	Register ARU_IRQ_NOTIFY	26-59
26.3.6.11	Register ARU_IRQ_EN	26-60
26.3.6.12	Register ARU_IRQ_FORCINT	26-61
26.3.6.13	Register ARU_IRQ_MODE	26-63
26.4	Broadcast Module (BRC)	26-64
26.4.1	Overview	26-64
26.4.2	BRC Configuration	26-64
26.4.3	BRC Interrupt Signals	26-66
26.4.4	BRC Configuration Registers Overview	26-66
26.4.5	BRC Configuration Registers Description	26-67
26.4.5.1	Register BRC_SRCx_ADDR (x=0-11)	26-67
26.4.5.2	Register BRC_SRCx_DEST (x:0...11)	26-68
26.4.5.3	Register BRC_IRQ_NOTIFY	26-70
26.4.5.4	Register BRC_IRQ_EN	26-72
26.4.5.5	Register BRC_IRQ_FORCINT	26-73
26.4.5.6	Register BRC_IRQ_MODE	26-74
26.4.5.7	Register BRC_RST	26-75
26.4.5.8	Register BRC_EIRQ_EN	26-76
26.5	First In First Out Module (FIFO)	26-77
26.5.1	Overview	26-77
26.5.2	Operation Modes	26-78
26.5.2.1	Normal Operation Mode	26-78
26.5.2.2	Ring Buffer Operation Mode	26-78
26.5.3	FIFO Interrupt Signals	26-78
26.5.4	FIFO _i Configuration Registers Overview	26-79
26.5.5	FIFO _i Configuration Registers Description	26-81
26.5.5.1	Register FIFO _i _CHx_CTRL (x:0...7)	26-81
26.5.5.2	Register FIFO _i _CHx_END_ADDR (x:0...7)	26-82
26.5.5.3	Register FIFO _i _CHx_START_ADDR (x:0...7)	26-83
26.5.5.4	Register FIFO _i _CHx_UPPER_WM (x:0...7)	26-83
26.5.5.5	Register FIFO _i _CHx_LOWER_WM (x:0...7)	26-84
26.5.5.6	Register FIFO _i _CHx_STATUS (x:0...7)	26-85
26.5.5.7	Register FIFO _i _CHx_FILL_LEVEL (x:0...7)	26-86

Table of Contents

26.5.5.8	Register FIFOi_CHx_WR_PTR (x:0...7)	26-87
26.5.5.9	Register FIFOi_CHx_RD_PTR (x:0...7)	26-87
26.5.5.10	Register FIFOi_CHx_IRQ_NOTIFY (x:0...7)	26-88
26.5.5.11	Register FIFOi_CHx_IRQ_EN (x:0...7)	26-89
26.5.5.12	Register FIFOi_CHx_IRQ_FORCINT	26-90
26.5.5.13	Register FIFOi_CHx_IRQ_MODE	26-91
26.5.5.14	Register FIFOi_CHx_EIRQ_EN (x:0...7)	26-93
26.6	AEI to FIFO Data Interface (AFD)	26-94
26.6.1	Overview	26-94
26.6.2	AFD Register overview	26-94
26.6.3	AFD Register description	26-95
26.6.3.1	Register AFD0_CHx_BUF_ACC (x:0...7)	26-95
26.7	FIFO to ARU Unit (F2A)	26-96
26.7.1	Overview	26-96
26.7.2	Transfer modes	26-96
26.7.2.1	Data transfer of both ARU words between ARU and FIFO	26-97
26.7.3	F2A Configuration Registers Overview	26-98
26.7.4	F2A Configuration Registers description	26-98
26.7.4.1	Register F2Ai_ENABLE	26-98
26.7.4.2	Register F2Ai_CHx_ARU_RD_FIFO (x: 0...7)	26-99
26.7.4.3	Register F2Ai_CHx_STR_CFG (x: 0...7)	26-100
26.8	Clock Management Unit (CMU)	26-101
26.8.1	Overview	26-101
26.8.1.1	CMU Block Diagram	26-102
26.8.2	Global Clock Divider	26-103
26.8.3	Configurable Clock Generation Subunit (CFGU)	26-103
26.8.4	Wave Form of Generated Clock Signal CMU_CLK[x]	26-104
26.8.5	Fixed Clock Generation (FXU)	26-105
26.8.6	External Generation Unit (EGU)	26-105
26.8.7	CMU Configuration Registers Overview	26-106
26.8.8	CMU Configuration Register Description	26-108
26.8.8.1	Register CMU_CLK_EN	26-108
26.8.8.2	Register CMU_GCLK_NUM	26-109
26.8.8.3	Register CMU_GCLK_DEN	26-111
26.8.8.4	Register CMU_CLK_x_CTRL (x:0...5)	26-112
26.8.8.5	Register CMU_CLK_6_CTRL	26-113
26.8.8.6	Register CMU_CLK_7_CTRL	26-114
26.8.8.7	Register CMU_ECLK_z_NUM (z:0...2)	26-115
26.8.8.8	Register CMU_ECLK_z_DEN (z:0...2)	26-116
26.8.8.9	Register CMU_FXCLK_CTRL	26-117
26.9	Time Base Unit (TBU)	26-118
26.9.1	Overview	26-118
26.9.1.1	TBU Block Diagram	26-119

Table of Contents

26.9.2	TBU Time Base Channels	26-120
26.9.2.1	TBU Channel Modes	26-120
26.9.3	TBU Configuration Registers Overview	26-120
26.9.4	TBU Registers description	26-121
26.9.4.1	Register TBU_CHEN	26-121
26.9.4.2	Register TBU_CH0_CTRL	26-122
26.9.4.3	Register TBU_CH0_BASE	26-123
26.9.4.4	Register TBU_CHy_CTRL (y:1, 2)	26-124
26.9.4.5	Register TBU_CHy_BASE (y:1,2)	26-125
26.10	Timer Input Module (TIM)	26-126
26.10.1	Overview	26-126
26.10.1.1	TIM Block Diagram	26-126
26.10.1.2	Input source selection INPUTSRCx	26-127
26.10.1.3	External capture source selection EXTCAPSRCx	26-128
26.10.2	TIM Filter Functionality (FLT)	26-129
26.10.2.1	Overview	26-129
26.10.2.2	TIM Filter Modes	26-131
26.10.2.3	TIM Filter reconfiguration	26-137
26.10.3	Timeout Detection Unit (TDU)	26-137
26.10.3.1	Architecture of the TDU Subunit	26-138
26.10.4	TIM Channel Architecture	26-140
26.10.4.1	Overview	26-140
26.10.4.2	TIM Channel Modes	26-142
26.10.5	MAP Submodule Interface	26-152
26.10.6	TIM Interrupt Signals	26-153
26.10.7	TIM Configuration Registers Overview	26-154
26.10.8	TIM Configuration Registers Description	26-156
26.10.8.1	Register TIMi_CHx_CTRL (x:0...7) (i: 12)	26-156
26.10.8.2	Register TIM0_CHx_CTRL (x:0...7)	26-161
26.10.8.3	Register TIMi_CHx_FLT_RE (i:0...2)(x:0...7)	26-166
26.10.8.4	Register TIMi_CHx_FLT_FE (i:0...2)(x:0...7)	26-167
26.10.8.5	Register TIMi_CHx_TDU (i:0...2)(x:0...7)	26-168
26.10.8.6	Register TIMi_CHx_GPR0 (i:0...2)(x:0...7)	26-169
26.10.8.7	Register TIMi_CHx_GPR1 (i:0...2)(x:0...7)	26-170
26.10.8.8	Register TIMi_CHx_CNT (i:0...2)(x:0...7)	26-171
26.10.8.9	Register TIMi_CHx_CNTS (i:0...2)(x:0...7)	26-172
26.10.8.10	Register TIMi_CHx_IRQ_NOTIFY (i:0...2)(x:0...7)	26-173
26.10.8.11	Register TIMi_CHx_IRQ_EN (i:0...2)(x:0...7)	26-174
26.10.8.12	Register TIMi_CHx_IRQ_FORCINT (i:0...2)(x:0...7)	26-175
26.10.8.13	Register TIMi_CHx_IRQ_MODE (i:0...2)(x:0...7)	26-177
26.10.8.14	Register TIMi_RST (i:0...2)	26-178
26.10.8.15	Register TIMi_IN_SRC (i:0...2)	26-179
26.10.8.16	Register TIMi_CHx_EIRQ_EN (i:0...2)(x:0...7)	26-182

Table of Contents

26.10.8.17	Register TIMi_CHx_TDUV (i:0...2)(x:0...7)	26-184
26.10.8.18	Register TIMi_CHx_TDUC (i:0...2)(x:0...7)	26-185
26.10.8.19	Register TIMi_CHx_ECNT (i:0...2)(x:0...7)	26-186
26.10.8.20	Register TIMi_CHx_ECTRL (i:0...2)(x:0...7)	26-187
26.11	Timer Output Module (TOM)	26-188
26.11.1	Overview	26-188
26.11.1.1	TOM Block diagram	26-189
26.11.2	TOM Global Channel Control (TGC0, TGC1)	26-190
26.11.2.1	Overview	26-190
26.11.2.2	TGC Subunit	26-190
26.11.3	TOM Channel (TOM_CH[x])	26-193
26.11.3.1	TOM Channel 0...7 architecture	26-194
26.11.3.2	TOM Channel 8...14 architecture	26-195
26.11.3.3	TOM Channel 15 architecture for PCM generation	26-196
26.11.3.4	Duty cycle, period and selected counter clock frequency update mechanisms 26-197	
26.11.3.5	TOM continuous mode	26-200
26.11.3.6	TOM One shot mode	26-201
26.11.3.7	Pulse count modulation	26-203
26.11.4	TOM BLDC Support	26-205
26.11.5	TOM Gated Counter Mode	26-205
26.11.6	TOM Interrupt signals	26-205
26.11.7	TOM Configuration Register overview	26-206
26.11.8	TOM Configuration Registers Description	26-208
26.11.8.1	Register TOMi_TGC0_GLB_CTRL	26-208
26.11.8.2	Register TOMi_TGC0_ENDIS_CTRL	26-211
26.11.8.3	Register TOMi_TGC0_ENDIS_STAT	26-213
26.11.8.4	Register TOMi_TGC0_ACT_TB	26-215
26.11.8.5	Register TOMi_TGC0_OUTEN_CTRL	26-216
26.11.8.6	Register TOMi_TGC0_OUTEN_STAT	26-218
26.11.8.7	Register TOMi_TGC0_FUPD_CTRL	26-220
26.11.8.8	Register TOMi_TGC0_INT_TRIG	26-223
26.11.8.9	Register TOMi_CHx_CTRL (x:0...14)	26-224
26.11.8.10	Register TOMi_CH15_CTRL	26-228
26.11.8.11	Register TOMi_CHx_CN0 (x:0...15)	26-231
26.11.8.12	Register TOMi_CHx_CM0 (x:0...15)	26-231
26.11.8.13	Register TOMi_CHx_SR0 (x:0...15)	26-233
26.11.8.14	Register TOMi_CHx_CM1 (x:0...15)	26-233
26.11.8.15	Register TOMi_CHx_SR1 (x:0...15)	26-235
26.11.8.16	Register TOMi_CHx_STAT (x:0...15)	26-235
26.11.8.17	Register TOMi_CHx_IRQ_NOTIFY (x:0...15)	26-236
26.11.8.18	Register TOMi_CHx_IRQ_EN (x:0...15)	26-237
26.11.8.19	Register TOMi_CHx_IRQ_FORCINT (x:0...15)	26-238

Table of Contents

26.11.8.20	Register TOMi_CHx_IRQ_MODE (x:0...15)	26-240
26.12	ARU-connected Timer Output Module (ATOM)	26-241
26.12.1	Overview	26-241
26.12.1.1	ATOM Block diagram	26-241
26.12.1.2	ATOM Global control (AGC)	26-243
26.12.1.3	ATOM Channel mode overview	26-243
26.12.2	ATOM Channel architecture	26-244
26.12.2.1	ATOM Channel architecture	26-244
26.12.2.2	ARU Communication Interface	26-245
26.12.3	ATOM Channel modes	26-247
26.12.3.1	ATOM Signal Output Mode Immediate (SOMI)	26-247
26.12.3.2	ATOM Signal Output Mode Compare (SOMC)	26-250
26.12.3.3	ATOM Signal Output Mode PWM (SOMP)	26-273
26.12.3.4	ATOM Signal Output Mode Serial (SOMS)	26-282
26.12.4	ATOM Interrupt signals	26-290
26.12.5	ATOM Register overview	26-290
26.12.6	ATOM Register description	26-292
26.12.6.1	Register ATOMi_AGC_GLB_CTRL	26-292
26.12.6.2	Register ATOMi_CHx_CTRL (x: 0...7)	26-294
26.12.6.3	Register ATOMi_CHx_STAT (x: 0...7)	26-300
26.12.6.4	Register ATOMi_CHx_RDADDR (x: 0...7)	26-302
26.12.6.5	Register ATOMi_CHx_CN0 (x: 0...7)	26-303
26.12.6.6	Register ATOMi_CHx_CM0 (x: 0...7)	26-304
26.12.6.7	Register ATOMi_CHx_SR0 (x: 0...7)	26-305
26.12.6.8	Register ATOMi_CHx_CM1 (x: 0...7)	26-306
26.12.6.9	Register ATOMi_CHx_SR1 (x: 0...7)	26-307
26.12.6.10	Register ATOMi_CHx_IRQ_NOTIFY (x:0...7)	26-308
26.12.6.11	Register ATOMi_CHx_IRQ_EN (x:0...7)	26-309
26.12.6.12	Register ATOMi_CHx_IRQ_FORCINT (x:0...7)	26-310
26.12.6.13	Register ATOMi_CHx_IRQ_MODE (x:0...7)	26-311
26.13	Multi Channel Sequencer (MCS)	26-312
26.13.1	Overview	26-312
26.13.1.1	Architecture	26-312
26.13.1.2	Scheduling	26-315
26.13.2	Instruction Set	26-317
26.13.2.1	Instruction Format	26-318
26.13.2.2	Data Transfer Instructions	26-319
26.13.2.3	ARU Instructions	26-324
26.13.2.4	Arithmetic Logic Instructions	26-329
26.13.2.5	Test Instructions	26-335
26.13.2.6	Control Flow Instructions	26-337
26.13.2.7	Other Instructions	26-339
26.13.3	MCS Internal Registers	26-341

Table of Contents

26.13.3.1	MCS Internal Registers Overview	26-341
26.13.3.2	General purpose register Rx (x:0...7)	26-342
26.13.3.3	Register STA	26-343
26.13.3.4	Register ACB	26-347
26.13.3.5	Register CTRG	26-348
26.13.3.6	Register STRG	26-350
26.13.3.7	Register TBU_TS0	26-352
26.13.3.8	Register TBU_TS1	26-353
26.13.3.9	Register TBU_TS2	26-353
26.13.3.10	Register MHB	26-355
26.13.4	MCS Configuration Registers	26-355
26.13.4.1	MCS Configuration Registers Overview	26-355
26.13.4.2	Register MCSi_CHx_CTRL (x:07)	26-357
26.13.4.3	Register MCSi_CHx_PC (x:0...7)	26-360
26.13.4.4	Register MCSi_CHx_Ry (x:0...7, y:0...7)	26-361
26.13.4.5	Register MCSi_CHx_ACB (x:0...7)	26-365
26.13.4.6	Register MCSi_CHx_IRQ_NOTIFY (x:0...7)	26-366
26.13.4.7	Register MCSi_CHx_IRQ_EN (x:0...7)	26-367
26.13.4.8	Register MCSi_CHx_IRQ_FORCINT (x:0...7)	26-368
26.13.4.9	Register MCSi_CHx_IRQ_MODE (x:0...7)	26-370
26.13.4.10	Register MCSi_CHx_EIRQ_EN (x:0...7)	26-372
26.13.4.11	Register MCSi_CTRL	26-373
26.13.4.12	Register MCSi_CTRG	26-375
26.13.4.13	Register MCSi_STRG	26-377
26.13.4.14	Register MCSi_RST	26-380
26.13.4.15	Register MCSi_ERR	26-384
26.14	Memory Configuration (MCFG)	26-386
26.14.1	Overview	26-386
26.14.1.1	Memory Layout Configurations (MSC_RAM1_EN_ADDR_MSB=0)	26-387
26.14.1.2	Memory layout Parameters (MSC_RAM1_EN_ADDR_MSB=0)	26-388
26.14.1.3	Memory Layout Configurations (MSC_RAM1_EN_ADDR_MSB=1)	26-389
26.14.1.4	Memory Layout Parameters (MSC_RAM1_EN_ADDR_MSB=1)	26-390
26.14.2	MCFG Configuration Registers	26-390
26.14.2.1	Register MCFG_CTRL	26-391
26.15	TIM0 Input Mapping Module (MAP)	26-392
26.15.1	Overview	26-392
26.15.1.1	MAP Submodule architecture	26-392
26.15.2	TIM Signal Preprocessing (TSPP)	26-393
26.15.2.1	TIM Signal Preprocessing (TSPP) subunit architecture	26-394
26.15.2.2	Bit Stream Combination	26-394
26.15.3	MAP Register overview	26-395

Table of Contents

26.15.4	MAP Register description	26-396
26.15.4.1	Register MAP_CTRL	26-396
26.16	Digital PLL Module (DPLL)	26-399
26.16.1	Overview	26-399
26.16.2	Requirements and demarcation	26-399
26.16.3	Input signal courses	26-401
26.16.4	Block and interface description	26-401
26.16.4.1	DPLL Block diagram	26-402
26.16.4.2	Interface description of DPLL	26-403
26.16.5	DPLL Architecture	26-408
26.16.5.1	Purpose of the module	26-408
26.16.5.2	Explanation of the prediction methodology	26-408
26.16.5.3	Clock topology	26-408
26.16.5.4	Clock generation	26-408
26.16.5.5	Typical frequencies	26-409
26.16.5.6	Time stamps and systematic corrections	26-409
26.16.5.7	DPLL Architecture overview	26-410
26.16.5.8	DPLL Architecture description	26-412
26.16.5.9	Block diagrams of time stamp processing	26-414
26.16.5.10	Register and RAM address overview	26-415
26.16.6	Prediction of the current increment duration	26-422
26.16.6.1	The use of increments in the past	26-422
26.16.6.2	Increment prediction in Normal Mode forwards (DIR1=0)	26-422
26.16.6.3	Increment prediction in Emergency Mode forwards (DIR2=0) ..	26-427
26.16.6.4	Increment prediction in Normal Mode backwards (DIR1=1) ..	26-430
26.16.6.5	Increment prediction in Emergency Mode backwards (DIR2=1)	26-431
26.16.7	Calculations for actions	26-433
26.16.7.1	Action calculations for TRIGGER forwards	26-435
26.16.7.2	Action calculations for TRIGGER backwards	26-437
26.16.7.3	Action calculations STATE forwards	26-439
26.16.7.4	Action calculations for STATE backwards	26-440
26.16.7.5	Update of RAM in Normal and Emergency Mode	26-442
26.16.7.6	Time and position stamps for actions in Normal Mode	26-446
26.16.7.7	The use of the RAM	26-449
26.16.7.8	Time and position stamps for actions in Emergency Mode ..	26-449
26.16.8	Signal processing	26-452
26.16.8.1	Time stamp processing	26-452
26.16.8.2	Count and compare unit	26-452
26.16.8.3	Sub pulse generation for SMC=0	26-452
26.16.8.4	Sub pulse generation for SMC=1	26-457
26.16.8.5	Calculation of the Accurate Position Values	26-461
26.16.8.6	Scheduling of the Calculation	26-462
26.16.9	DPLL Interrupt signals	26-487

Table of Contents

26.16.9.1	DPLL Interrupt signals	26-487
26.16.10	DPLL Register overview	26-489
26.16.11	DPLL Register and Memory description	26-501
26.16.11.1	Register DPLL_CTRL_0	26-501
26.16.11.2	Register DPLL_CTRL_1	26-505
26.16.11.3	Register DPLL_CTRL_2	26-513
26.16.11.4	Register DPLL_CTRL_3	26-515
26.16.11.5	Register DPLL_CTRL_4	26-517
26.16.11.6	Register DPLL_ACT_STA	26-519
26.16.11.7	Register DPLL_OSW	26-521
26.16.11.8	Register DPLL_AOSV_2	26-523
26.16.11.9	Register DPLL_APT	26-524
26.16.11.10	Register DPLL_APS	26-527
26.16.11.11	Register DPLL_APT_2C	26-530
26.16.11.12	Register DPLL_APS_1C3	26-532
26.16.11.13	Register DPLL_NUTC	26-533
26.16.11.14	Register DPLL_NUSC	26-536
26.16.11.15	Register DPLL_NTI_CNT	26-539
26.16.11.16	Register DPLL_IRQ_NOTIFY	26-540
26.16.11.17	Register DPLL_IRQ_EN	26-543
26.16.11.18	Register DPLL_IRQ_FORCINT	26-547
26.16.11.19	Register DPLL_IRQ_MODE	26-549
26.16.11.20	Register DPLL_EIRQ_EN	26-550
26.16.11.21	Register DPLL_INC_CNT1	26-554
26.16.11.22	Register DPLL_INC_CNT2	26-555
26.16.11.23	Register DPLL_APT_SYNC	26-556
26.16.11.24	Register DPLL_APS_SYNC	26-559
26.16.11.25	Register DPLL_TBU_TS0_T	26-562
26.16.11.26	Register DPLL_TBU_TS0_S	26-562
26.16.11.27	Register DPLL_ADD_IN_LD1	26-564
26.16.11.28	Register DPLL_ADD_IN_LD2	26-565
26.16.11.29	Register DPLL_STATUS	26-566
26.16.11.30	Register DPLL_ID_PMTR_x	26-573
26.16.11.31	Register DPLL_CTRL_0_SHADOW_TRIGGER	26-574
26.16.11.32	Register DPLL_CTRL_0_SHADOW_STATE	26-576
26.16.11.33	Register DPLL_CTRL_1_SHADOW_TRIGGER	26-577
26.16.11.34	Register DPLL_CTRL_1_SHADOW_STATE	26-578
26.16.11.35	Register DPLL_RAM_INI	26-580
26.16.11.36	Memory DPLL_PSA[i]	26-581
26.16.11.37	Memory DPLL_DLA[i]	26-581
26.16.11.38	Memory DPLL_NA[i]	26-582
26.16.11.39	Memory DPLL_DTA[i]	26-584
26.16.11.40	Memory DPLL_TS_T	26-584

Table of Contents

26.16.11.41	Memory DPLL_TS_T_OLD	26-586
26.16.11.42	Memory DPLL_FTV_T	26-587
26.16.11.43	Memory DPLL_TS_S	26-588
26.16.11.44	Memory DPLL_TS_S_OLD	26-589
26.16.11.45	Memory DPLL_FTV_S	26-590
26.16.11.46	Memory DPLL_THMI	26-591
26.16.11.47	Memory DPLL_THMA	26-592
26.16.11.48	Memory DPLL_THVAL	26-592
26.16.11.49	Memory DPLL_TOV	26-594
26.16.11.50	Memory DPLL_TOV_S	26-595
26.16.11.51	Memory DPLL_ADD_IN_CAL1	26-596
26.16.11.52	Memory DPLL_ADD_IN_CAL2	26-596
26.16.11.53	Memory DPLL_MPVAL1	26-598
26.16.11.54	Memory DPLL_MPVAL2	26-599
26.16.11.55	Memory DPLL_NMB_T_TAR	26-600
26.16.11.56	Memory DPLL_NMB_T_TAR_OLD	26-601
26.16.11.57	Memory DPLL_NMB_S_TAR	26-602
26.16.11.58	Memory DPLL_NMB_S_TAR_OLD	26-603
26.16.11.59	Memory DPLL_RCDT_TX	26-604
26.16.11.60	Memory DPLL_RCDT_SX	26-604
26.16.11.61	Memory DPLL_RCDT_TX_NOM	26-606
26.16.11.62	Memory DPLL_RCDT_SX_NOM	26-606
26.16.11.63	Memory DPLL_RDT_T_ACT	26-608
26.16.11.64	Memory DPLL_RDT_S_ACT	26-608
26.16.11.65	Memory DPLL_DT_T_ACT	26-610
26.16.11.66	Memory DPLL_DT_S_ACT	26-610
26.16.11.67	Memory DPLL_EDT_T	26-612
26.16.11.68	Memory DPLL_MEDT_T	26-612
26.16.11.69	Memory DPLL_EDT_S	26-614
26.16.11.70	Memory DPLL_MEDT_S	26-614
26.16.11.71	Memory DPLL_CDT_TX	26-616
26.16.11.72	Memory DPLL_CDT_SX	26-616
26.16.11.73	Memory DPLL_CDT_TX_NOM	26-617
26.16.11.74	Memory DPLL_CDT_SX_NOM	26-617
26.16.11.75	Memory DPLL_TLR	26-618
26.16.11.76	Memory DPLL_SLR	26-618
26.16.11.77	Memory DPLL_PDT[i]	26-620
26.16.11.78	Memory DPLL_MLS1	26-621
26.16.11.79	Memory DPLL_MLS2	26-622
26.16.11.80	Memory DPLL_CNT_NUM1	26-623
26.16.11.81	Memory DPLL_CNT_NUM2	26-623
26.16.11.82	Memory DPLL_PVT	26-624
26.16.11.83	Memory DPLL_PSTC	26-625

Table of Contents

26.16.11.84	Memory DPLL_PSSC	26-626
26.16.11.85	Memory DPLL_PSTM	26-627
26.16.11.86	Memory DPLL_PSTM_OLD	26-628
26.16.11.87	Memory DPLL_PSSM	26-629
26.16.11.88	Memory DPLL_PSSM_OLD	26-630
26.16.11.89	Memory DPLL_NMB_T	26-631
26.16.11.90	Memory DPLL_NMB_S	26-631
26.16.11.91	Memory DPLL_RDT_Sx	26-632
26.16.11.92	Memory DPLL_TSF_S[i]	26-633
26.16.11.93	Memory DPLL_ADT_S[i]	26-633
26.16.11.94	Memory DPLL_DT_S[i]	26-634
26.16.11.95	Memory DPLL_TSAC[i]	26-635
26.16.11.96	Memory DPLL_PSAC[i]	26-636
26.16.11.97	Memory DPLL_ACBi	26-637
26.16.11.98	Memory DPLL_RDT_T[i]	26-639
26.16.11.99	Memory DPLL_TSF_T[i]	26-639
26.16.11.100	Memory DPLL_ADT_T[i]	26-641
26.16.11.101	Memory DPLL_DT_T[i]	26-642
26.16.12	Terms and Abbreviations	26-642
26.17	Sensor Pattern Evaluation (SPE)	26-643
26.17.1	Overview	26-643
26.17.1.1	SPE Submodule integration concept into GTM	26-643
26.17.1.2	SPE Sample input pattern for	26-644
26.17.2	SPE Submodule description	26-644
26.17.2.1	SPE to TOM Connections	26-645
26.17.2.2	SPE Submodule architecture	26-645
26.17.2.3	SPE[i]_IN_PAT register representation	26-648
26.17.2.4	SPE Revolution detection	26-649
26.17.3	SPE Interrupt signals	26-649
26.17.4	SPE Register overview	26-650
26.17.5	SPE Register description	26-650
26.17.5.1	Register SPEi_CTRL_STAT	26-650
26.17.5.2	Register SPEi_PAT	26-653
26.17.5.3	Register SPEi_OUT_PATx (x: 07)	26-655
26.17.5.4	Register SPEi_OUT_CTRL	26-656
26.17.5.5	Register SPEi_CNT	26-657
26.17.5.6	Register SPEi_CMP	26-658
26.17.5.7	Register SPEi_IRQ_NOTIFY	26-659
26.17.5.8	Register SPEi_IRQ_EN	26-660
26.17.5.9	Register SPEi_IRQ_FORCINT	26-661
26.17.5.10	Register SPEi_IRQ_MODE	26-662
26.17.5.11	Register SPEi_EIRQ_EN	26-663
26.18	Interrupt Concentrator Module (ICM)	26-664

Table of Contents

26.18.1	Overview	26-664
26.18.2	Bundling	26-664
26.18.2.1	GTM Infrastructure Interrupt Bundling	26-664
26.18.2.2	DPLL Interrupt Bundling	26-664
26.18.2.3	TIM Interrupt Bundling	26-665
26.18.2.4	MCS Interrupt Bundling	26-665
26.18.2.5	TOM and ATOM Interrupt Bundling	26-665
26.18.3	ICM Interrupt Signals	26-668
26.18.4	ICM Configuration Registers Overview	26-670
26.18.5	ICM Configuration Registers Description	26-671
26.18.5.1	Register ICM_IRQG_0	26-671
26.18.5.2	Register ICM_IRQG_1 (DPLL Interrupt Group)	26-674
26.18.5.3	Register ICM_IRQG_2	26-677
26.18.5.4	Register ICM_IRQG_4	26-680
26.18.5.5	Register ICM_IRQG_6	26-683
26.18.5.6	Register ICM_IRQG_9	26-686
26.18.5.7	Register ICM_IRQG_MEI (Module Error Interrupt)	26-689
26.18.5.8	Register ICM_IRQG_CEI0 (Channel Error Interrupt 0)	26-691
26.18.5.9	Register ICM_IRQG_CEI1 (Channel Error Interrupt 1)	26-693
26.18.5.10	Register ICM_IRQG_CEI3 (Channel Error Interrupt 3)	26-696
26.19	Output Compare Unit (CMP)	26-699
26.19.1	Overview	26-699
26.19.1.1	Architecture of the Compare Unit	26-700
26.19.2	Bitwise Compare Unit (BWC)	26-700
26.19.3	Configuration of the Compare Unit	26-701
26.19.4	Error Generator	26-701
26.19.5	CMP Interrupt Signal	26-701
26.19.6	CMP Configuration Registers Overview	26-702
26.19.7	CMP Configuration Registers Description	26-702
26.19.7.1	Register CMP_EN	26-702
26.19.7.2	Register CMP_IRQ_NOTIFY	26-705
26.19.7.3	Register CMP_IRQ_EN	26-708
26.19.7.4	Register CMP_IRQ_FORCINT	26-710
26.19.7.5	Register CMP_IRQ_MODE	26-713
26.19.7.6	Register CMP_EIRQ_EN	26-714
26.20	Monitor Unit (MON)	26-716
26.20.1	Overview	26-716
26.20.1.1	MON Block Diagram	26-716
26.20.1.2	Realization without Activity Checker of the clock signals	26-717
26.20.2	Clock Monitoring	26-718
26.20.3	CMP error Monitoring	26-718
26.20.4	Checking the Characteristics of Signals by MCS	26-718
26.20.5	Checking ARU Cycle Time	26-719

Table of Contents

26.20.6	MON Interrupt Signals	26-719
26.20.7	MON Registers Overview	26-719
26.20.8	MON Configuration Registers Description	26-721
26.20.8.1	Register MON_STATUS	26-721
26.20.8.2	Register MON_ACTIVITY_0	26-723
26.21	Appendix	26-727
26.21.1	ARU Write Address Overview	26-727
26.22	GTM Implementation	26-729
26.22.1	GTM Registers	26-729
26.22.2	Port Connections	26-741
26.22.2.1	Port to GTM Control Registers	26-756
26.22.2.2	GTM to Port Control Registers	26-787
26.22.3	MSC Connections	26-788
26.22.3.1	GTM to MSC Control Registers	26-791
26.22.4	DSADC Connections	26-811
26.22.5	ADC Connections	26-826
26.22.6	SENT Connections	26-828
26.22.7	CAN Connection	26-829
26.22.8	CCU6x Connections	26-832
26.22.9	PSI5 Connections	26-834
26.22.10	GTM Data Exchange Registers	26-837
26.22.11	SCU Connections	26-850
26.22.12	GTM Debug Interface	26-852
26.22.12.1	OCDS Trigger Bus (OTGB) Interface	26-852
26.22.12.2	GTM Debug Registers	26-858
26.23	Revision History	26-864
27	Capture/Compare Unit 6 (CCU6)	27-1
27.1	Introduction	27-1
27.1.1	Feature Set Overview	27-2
27.1.2	Block Diagram	27-3
27.1.3	CCU6 Kernel Registers	27-4
27.2	Operating Timer T12	27-8
27.2.1	T12 Overview	27-9
27.2.2	T12 Counting Scheme	27-11
27.2.2.1	Clock Selection	27-11
27.2.2.2	Edge-Aligned / Center-Aligned Mode	27-12
27.2.2.3	Single-Shot Mode	27-14
27.2.3	T12 Compare Mode	27-15
27.2.3.1	Compare Channels	27-15
27.2.3.2	Channel State Bits	27-16
27.2.3.3	Hysteresis-Like Control Mode	27-21
27.2.4	Compare Mode Output Path	27-22

Table of Contents

27.2.4.1	Dead-Time Generation	27-22
27.2.4.2	State Selection	27-24
27.2.4.3	Output Modulation and Level Selection	27-25
27.2.5	T12 Capture Modes	27-27
27.2.6	T12 Shadow Register Transfer	27-31
27.2.7	Timer T12 Operating Mode Selection	27-32
27.2.8	T12 related Registers	27-33
27.2.8.1	T12 Counter Register	27-33
27.2.8.2	Period Register	27-34
27.2.8.3	Capture/Compare Registers	27-35
27.2.8.4	Capture/Compare Shadow Registers	27-36
27.2.8.5	Dead-time Control Register	27-37
27.2.9	Capture/Compare Control Registers	27-39
27.2.9.1	Channel State Bits	27-39
27.2.9.2	T12 Mode Control Register	27-43
27.2.9.3	Timer Control Registers	27-44
27.3	Operating Timer T13	27-53
27.3.1	T13 Overview	27-53
27.3.2	T13 Counting Scheme	27-56
27.3.2.1	Clock Selection	27-56
27.3.2.2	T13 Counting	27-57
27.3.2.3	Single-Shot Mode	27-58
27.3.2.4	Synchronization to T12	27-59
27.3.3	T13 Compare Mode	27-61
27.3.4	Compare Mode Output Path	27-63
27.3.5	T13 Shadow Register Transfer	27-64
27.3.6	T13 related Registers	27-66
27.3.6.1	T13 Counter Register	27-66
27.3.6.2	Period Register	27-67
27.3.6.3	Compare Register	27-68
27.3.6.4	Compare Shadow Register	27-69
27.4	Synchronous Start Feature	27-70
27.5	Trap Handling	27-71
27.6	Multi-Channel Mode	27-73
27.7	Hall Sensor Mode	27-75
27.7.1	Hall Pattern Evaluation	27-76
27.7.2	Hall Pattern Compare Logic	27-78
27.7.3	Hall Mode Flags	27-79
27.7.4	Hall Mode for Brushless DC-Motor Control	27-81
27.8	Modulation Control Registers	27-83
27.8.1	Modulation Control	27-83
27.8.2	Trap Control Register	27-85
27.8.3	Passive State Level Register	27-88

Table of Contents

27.8.4	Multi-Channel Mode Registers	27-89
27.9	Interrupt Handling	27-96
27.9.1	Interrupt Structure	27-96
27.9.2	Interrupt Registers	27-98
27.9.2.1	Interrupt Status Register	27-98
27.9.2.2	Interrupt Status Set Register	27-101
27.9.2.3	Status Reset Register	27-103
27.9.2.4	Interrupt Enable Register	27-105
27.9.2.5	Interrupt Node Pointer Register	27-107
27.10	General Module Operation	27-110
27.10.1	Input Selection	27-110
27.10.2	Input Monitoring	27-111
27.10.3	OCDS Suspend	27-112
27.10.4	OCDS Trigger Bus (OTGB) Interface	27-114
27.10.5	General Registers	27-115
27.10.5.1	ID Register	27-115
27.10.5.2	Port Input Select Registers	27-116
27.10.5.3	Module Configuration Register	27-121
27.10.5.4	Input Monitoring Register	27-122
27.10.5.5	Lost Indicator Register	27-125
27.10.5.6	Kernel State Control Sensitivity Register	27-127
27.10.6	BPI Registers	27-128
27.10.6.1	System Registers	27-130
27.11	Implementation	27-138
27.11.1	Address Map	27-138
27.11.1.1	Module Registers	27-139
27.11.2	Module Output Select	27-142
27.11.3	Synchronous Start	27-143
27.11.4	Digital Connections	27-144
27.11.4.1	Connections of CCU60	27-144
27.11.4.2	Connections of CCU61	27-149
28	General Purpose Timer Unit (GPT12)	28-1
28.1	Timer Block GPT1	28-2
28.1.1	GPT1 Core Timer T3 Control	28-4
28.1.2	GPT1 Core Timer T3 Operating Modes	28-6
28.1.3	GPT1 Auxiliary Timers T2/T4 Control	28-13
28.1.4	GPT1 Auxiliary Timers T2/T4 Operating Modes	28-14
28.1.5	GPT1 Clock Signal Control	28-25
28.1.6	GPT1 Registers	28-28
28.1.6.1	GPT1 Timer Registers	28-28
28.1.6.2	GPT1 Timer Control Registers	28-30
28.2	Timer Block GPT2	28-40

Table of Contents

28.2.1	GPT2 Core Timer T6 Control	28-42
28.2.2	GPT2 Core Timer T6 Operating Modes	28-44
28.2.3	GPT2 Auxiliary Timer T5 Control	28-47
28.2.4	GPT2 Auxiliary Timer T5 Operating Modes	28-48
28.2.5	GPT2 Register CAPREL Operating Modes	28-53
28.2.6	GPT2 Clock Signal Control	28-59
28.2.7	GPT2 Registers	28-62
28.2.7.1	GPT2 Timer Registers	28-62
28.2.7.2	GPT2 Timer Control Registers	28-64
28.3	GPT12 Kernel Register Overview	28-71
28.4	General Module Operation	28-72
28.4.1	Input Selection	28-72
28.4.2	OCDS Suspend	28-72
28.4.3	Miscellaneous GPT12 Registers	28-73
28.4.3.1	Port Input Select Register	28-73
28.4.3.2	Identification Register	28-74
28.4.4	BPI Registers	28-76
28.4.4.1	System Registers	28-78
28.5	Implementation of the GPT12 Module	28-85
28.5.1	Address Map	28-85
28.5.2	Module Connections	28-85
29	Versatile Analog-to-Digital Converter (VADC)	29-1
29.1	Introduction and Basic Structure	29-4
29.2	Electrical Models	29-9
29.3	Configuration of General Functions	29-14
29.3.1	Module Identification	29-14
29.3.2	System Registers	29-15
29.3.3	General Clocking Scheme and Control	29-22
29.3.4	Register Access Control	29-27
29.4	Analog Module Activation and Control	29-30
29.4.1	Analog Converter Control	29-30
29.4.2	Alternate Reference Selection	29-31
29.4.3	Calibration	29-31
29.5	Conversion Request Generation	29-32
29.5.1	Queued Request Source Handling	29-34
29.5.2	Channel Scan Request Source Handling	29-50
29.5.3	Request Source Arbitration	29-66
29.5.3.1	Arbiter Operation and Configuration	29-67
29.5.3.2	Conversion Start Mode	29-72
29.6	Analog Input Channel Configuration	29-74
29.6.1	Channel Parameters	29-74
29.6.2	Alias Feature	29-80

Table of Contents

29.6.3	Conversion Modes	29-82
29.6.4	Compare with Standard Conversions (Limit Checking)	29-84
29.6.5	Utilizing Fast Compare Mode	29-86
29.6.6	Boundary Flag Control	29-87
29.6.7	Conversion Timing	29-93
29.7	Conversion Result Handling	29-94
29.7.1	Storage of Conversion Results	29-94
29.7.2	Data Alignment	29-106
29.7.3	Wait-for-Read Mode	29-107
29.7.4	Result FIFO Buffer	29-108
29.7.5	Result Event Generation	29-109
29.7.6	Data Modification	29-110
29.8	Synchronization of Conversions	29-117
29.8.1	Synchronized Conversions for Parallel Sampling	29-117
29.8.2	Equidistant Sampling	29-121
29.9	Safety Features	29-123
29.9.1	Broken Wire Detection	29-123
29.9.2	Signal Path Test Modes	29-124
29.9.3	Configuration of Test Functions	29-126
29.10	External Multiplexer Control	29-128
29.11	Service Request Generation	29-133
29.12	Implementation into the TC26x B-Step	29-147
29.12.1	Product-Specific Configuration	29-147
29.12.2	Summary of Registers and Locations	29-149
29.12.3	Analog Module Connections in the TC26x B-Step	29-154
29.12.4	Digital Module Connections in the TC26x B-Step	29-157
29.13	Use Case Example for VADC	29-165
30	Delta-Sigma Analog-to-Digital Converter (DSADC)	30-1
30.1	Introduction and Basic Structure	30-4
30.2	Configuration of General Functions	30-6
30.2.1	Module Identification	30-6
30.2.2	System Registers	30-7
30.2.3	Register Access Control	30-14
30.2.4	Global Configuration Registers	30-16
30.3	Input Channel Configuration	30-21
30.3.1	Modulator Clock Selection and Generation	30-24
30.3.2	Input Data Selection	30-25
30.3.3	On-Chip Modulator	30-27
30.3.4	Input Path Control	30-29
30.3.5	Common Mode Voltage	30-36
30.3.6	Common Mode Hold Voltage	30-36
30.3.7	Calibration Support	30-41

Table of Contents

30.4	Main Filter Chain	30-48
30.4.1	CIC Filter	30-48
30.4.2	FIR Filters	30-50
30.4.3	Offset Compensation	30-53
30.4.4	Integrator Stage	30-54
30.5	Auxiliary Filter	30-58
30.6	Filter Configuration and Control	30-61
30.6.1	Filter Configuration Options	30-61
30.6.2	Recommended Settings	30-66
30.6.3	Group Delay	30-69
30.7	Conversion Result Handling	30-70
30.8	Service Request Generation	30-72
30.9	Resolver Support	30-75
30.9.1	Carrier Signal Generation	30-75
30.9.2	Return Signal Synchronization	30-79
30.10	Time-Stamp Support	30-83
30.11	Implementation into the TC26x B-Step	30-85
30.11.1	Product-Specific Configuration	30-85
30.11.2	Summary of Registers and Locations	30-86
30.11.3	Analog Module Connections in the TC26x B-Step	30-89
30.11.4	Digital Module Connections in the TC26x B-Step	30-90
31	Inter-Integrated Circuit Module (I2C)	31-1
31.1	Overview	31-1
31.1.1	I2C-bus Overview	31-1
31.1.2	I2C Module Overview	31-2
31.1.3	References	31-3
31.2	I2C Module Functional Specification	31-4
31.2.1	I2C Protocol	31-4
31.2.2	Clock and Timing Control	31-11
31.2.2.1	Baudrate Generation	31-11
31.2.2.2	I2C Signal Timing Adjustment	31-14
31.2.3	I2C Kernel Control Logic	31-15
31.2.4	FIFO Operation	31-28
31.2.4.1	Data Transmission	31-29
31.2.4.2	Transmit Request Generation	31-30
31.2.4.3	Transmit Data Alignment	31-32
31.2.4.4	Data Reception	31-37
31.2.4.5	Receive Request Generation	31-37
31.2.4.6	Receive Data Alignment	31-40
31.2.4.7	Switching between Transmission and Reception	31-45
31.2.5	Service Request Block Operation	31-45
31.2.5.1	Overview of Service Requests	31-45

Table of Contents

31.2.5.2	Interrupt Service Request Structure	31-48
31.3	I2C Module Internal Registers	31-51
31.3.1	Global Module Control Registers	31-51
31.3.2	FIFO Registers	31-61
31.3.3	Basic Interrupt Registers	31-67
31.3.4	Error Interrupt Source Registers	31-73
31.3.5	Protocol Interrupt Source Registers	31-76
31.4	I2C Module Implementation	31-81
31.4.1	Interfaces of the I2C Module(s)	31-81
31.4.2	Module Clock Control	31-82
31.4.3	Bus Peripheral Interface Registers	31-84
31.4.4	I2C Module Registers Overview	31-85
31.4.5	Port and I/O Line Control	31-90
31.4.6	Interrupt Control	31-91
31.5	Module Integration	31-92
31.5.1	Integration Overview	31-92
31.5.2	BPI_SPB Module Registers	31-93
31.5.2.1	System Registers	31-93
32	Input Output Monitor (IOM)	32-1
32.1	Overview	32-1
32.2	Features	32-1
32.3	Interfaces	32-2
32.4	Kernel Description	32-3
32.5	Filter & Prescaler Channel Description	32-4
32.6	EXOR Combiner Description	32-12
32.7	Logic Analyzer Module (LAM) Description	32-13
32.8	Event Combiner Module (ECM) Description	32-15
32.9	IOM Registers	32-17
32.9.1	IOM Identification Register (IOM_ID)	32-20
32.9.2	Filter & Prescaler Cell (FPC) Registers	32-21
32.9.3	GTM Input Related Registers	32-25
32.9.4	Logic Analyzer Module (LAM) Registers	32-27
32.9.5	Event Combiner Module (ECM) Registers	32-32
32.9.6	System Registers	32-37
32.10	SoC Integration	32-44
32.11	Example Monitor/Safety Measures	32-47
32.11.1	Example 1 - Pulse or duty cycle too short	32-48
32.11.2	Example 2 - Pulse or duty cycle too long	32-49
32.11.3	Example 3 - Period too short	32-50
32.11.4	Example 4 - Period too long	32-51
32.11.5	Example 5 - Diagnosis of Command and Feedback - acceptable propagation window and/or signal consistency check	32-52

Table of Contents

32.11.6	Example 6 - Diagnosis of Set-up and Hold times	32-53
33	Peripheral Sensor Interface (PSI5)	33-1
33.1	PSI5 Kernel Description	33-1
33.1.1	Overview	33-2
33.2	General Operation	33-3
33.3	Definitions	33-4
33.4	PSI5 Operation	33-5
33.5	Frame Formats and Definitions	33-5
33.5.1	PSI5 V1.3 Frame	33-5
33.5.2	Extended PSI5 Frame (non standard)	33-6
33.5.3	Extended Serial Data Encoding ("Slow Channel")	33-7
33.5.4	Extended Serial Data Frame	33-7
33.6	Sync Pulses	33-8
33.6.1	Synchronous Transmission	33-8
33.6.2	ECU to Sensor Communication	33-9
33.7	Manchester Decoding	33-11
33.8	Bit Rate Generation	33-14
33.9	Digital Glitch Filter	33-17
33.10	Time Stamp Generation	33-18
33.11	Error Detection Capabilities	33-20
33.12	Interrupts	33-20
33.13	Trigger Outputs	33-21
33.14	PSI5 Kernel Registers	33-22
33.14.1	Module Control	33-28
33.14.2	Input and Output Control	33-49
33.14.3	Receiver Control Registers	33-52
33.14.4	Receive Data and Status Registers	33-60
33.14.5	Receive Data Memory	33-67
33.14.6	Sync Pulse Control	33-97
33.14.7	Interrupt Control Registers	33-115
33.15	PSI5 Module Implementation	33-141
33.15.1	Interface Connections of the PSI5 Module	33-141
33.15.1.1	On-Chip Connections	33-142
33.15.2	PSI5 Module-Related External Registers	33-143
33.15.2.1	Port Control	33-143
33.15.2.2	Timing Constraints	33-144
33.16	Revision History	33-146
34	Peripheral Sensor Interface with Serial PHY Connection (PSI5-S)	34-1
34.1	PSI5-S Description	34-1
34.1.1	Overview	34-2
34.2	Definitions	34-4
34.3	General Operation	34-4

Table of Contents

34.4	PSI5 ECU to Sensor Operation	34-5
34.5	Frame Formats and Definitions	34-6
34.5.1	Communication between PSI5-S and PHY via UART	34-6
34.5.1.1	“Packet Frames” received from PHY	34-6
34.5.1.2	PSI5-S UART Frames transmitted to PHY	34-10
34.5.2	Communication between PHY and Sensor (PSI5 Standard)	34-10
34.5.2.1	PSI5 Standard Frame Format	34-10
34.5.2.2	PSI5 Extended Frame Format	34-11
34.5.2.3	Sync Pulses	34-11
34.6	Clock Generation	34-14
34.6.1	Overview on Clocks in the System	34-15
34.7	Time Stamp Generation	34-17
34.8	Watch Dog Timers	34-18
34.9	Send Data	34-21
34.9.1	Channel Trigger	34-21
34.9.2	Sync Pulse Control	34-24
34.9.3	Send Data Preparation	34-24
34.10	Message Generation	34-26
34.11	DMA Support	34-27
34.11.1	Single DMA, 8 dedicated DMAs	34-27
34.11.2	Two daisy chained DMAs	34-28
34.11.3	Interrupts for DMA support	34-36
34.12	Error Detection Capabilities	34-37
34.13	Special use of Channel 0	34-38
34.14	ASC Kernel Description	34-38
34.14.1	Overview	34-39
34.14.2	General Operation	34-41
34.14.3	Asynchronous Operation	34-42
34.14.3.1	Asynchronous Data Frames	34-43
34.14.3.2	Asynchronous Transmission	34-45
34.14.3.3	Asynchronous Reception	34-45
34.14.3.4	RXD/TXD Data Path Selection in Asynchronous Modes	34-46
34.14.4	Synchronous Operation	34-47
34.14.4.1	Synchronous Transmission	34-48
34.14.4.2	Synchronous Reception	34-48
34.14.4.3	Synchronous Timing	34-49
34.14.5	Baud Rate Generation	34-50
34.14.5.1	Baud Rates in Asynchronous Mode	34-51
34.14.5.2	Baud Rates in Synchronous Mode	34-54
34.14.6	Hardware Error Detection Capabilities	34-55
34.14.7	Interrupts	34-55
34.15	Interrupts	34-56
34.16	Trigger Outputs	34-57

Table of Contents

34.17	PSI5-S Kernel Registers	34-58
34.17.1	Module Control	34-62
34.17.2	Input and Output Control	34-81
34.17.3	Receiver Control Registers	34-83
34.17.4	Receive Data and Status Registers	34-88
34.17.5	Sync Pulse Control	34-97
34.17.6	ASC Registers	34-104
34.17.7	Interrupt Control Registers	34-114
34.18	PSI5-S Module Implementation	34-139
34.18.1	Interface Connections of the PSI5-S Module	34-139
34.18.1.1	On-Chip Connections	34-140
34.18.2	PSI5-S Module-Related External Registers	34-141
34.18.2.1	Port Control	34-141
34.19	Revision History	34-143
35	Ethernet MAC (ETH)	35-1
35.1	Overview	35-1
35.1.1	General Module Description	35-2
35.1.2	System Overview	35-3
35.1.2.1	System-Level Block Diagram	35-3
35.1.2.2	Interfaces	35-3
35.1.2.3	Transmit and Receive FIFOs	35-3
35.1.3	Features List	35-5
35.1.3.1	GMAC Core Features	35-5
35.1.3.2	DMA Block Features	35-6
35.1.3.3	Transaction Layer (MTL) Features	35-6
35.1.3.4	Monitoring, Test, and Debugging Support Features	35-8
35.2	Architecture	35-9
35.2.1	Introduction	35-9
35.2.2	IEEE 1588-2002 Overview	35-10
35.2.2.1	Reference Timing Source	35-12
35.2.2.2	Transmit Path Functions	35-12
35.2.2.3	Receive Path Functions	35-13
35.2.2.4	Time Stamp Error Margin	35-13
35.2.2.5	Frequency Range of Reference Timing Clock	35-13
35.2.2.6	Advanced Time Stamp Feature Support	35-14
35.2.3	AHB Application Host Interface	35-24
35.2.4	DMA Controller	35-26
35.2.4.1	Initialization	35-27
35.2.4.2	Transmission	35-30
35.2.4.3	Reception	35-35
35.2.4.4	Interrupts	35-39
35.2.5	MAC Transaction Layer (MTL)	35-40

Table of Contents

35.2.5.1	Transmit Path	35-40
35.2.5.2	Receive Path	35-45
35.2.6	GMAC Core	35-48
35.2.6.1	Transmission	35-48
35.2.6.2	MAC Transmit Interface Protocol	35-52
35.2.6.3	Reception	35-52
35.2.6.4	System Time Register Module	35-60
35.2.7	MAC Management Counters	35-63
35.2.7.1	Address Assignments	35-63
35.2.7.2	MMC Register Description	35-72
35.2.8	Power Management Block	35-86
35.2.8.1	PMT Block Description	35-86
35.2.8.2	Remote Wake-Up Frame Detection	35-90
35.2.8.3	Magic Packet Detection	35-90
35.2.8.4	System Considerations During Power-Down	35-91
35.2.9	Station Management Agent	35-92
35.2.9.1	Functions	35-92
35.2.9.2	MII Management Write Operation	35-93
35.2.9.3	MII Management Read Operation	35-94
35.2.10	Reduced Media Independent Interface	35-95
35.2.10.1	Block Diagram	35-95
35.2.10.2	Block Overview	35-96
35.2.10.3	Transmit Bit Ordering	35-97
35.2.10.4	RMII Transmit Timing Diagrams	35-97
35.2.11	Interrupts From the GMAC Core	35-100
35.3	Register	35-102
35.3.1	Register Maps	35-103
35.3.1.1	Register Overview	35-103
35.3.1.2	DMA Register Map	35-103
35.3.1.3	GMAC Register Map	35-105
35.3.1.4	Ethernet MAC Additional Module Control Registers	35-111
35.3.2	Register Description	35-112
35.4	Descriptors	35-419
35.4.1	Normal Descriptor Formats	35-419
35.4.1.1	Receive Descriptor	35-420
35.4.1.2	Transmit Descriptor	35-427
35.4.1.3	Descriptor Format With IEEE 1588 Time Stamping Enabled ..	35-434
35.4.2	Alternate or Enhanced Descriptors	35-438
35.4.2.1	Transmit Descriptor	35-438
35.4.2.2	Receive Descriptor	35-445
35.5	Ethernet MAC Module Implementation	35-454
35.5.1	Interface Connections of the Ethernet MAC Module	35-454
35.5.1.1	On-Chip Connections	35-456

Table of Contents

35.5.1.2	Clocks	35-456
35.5.2	Ethernet MAC Module-Implementation Related Registers	35-458
35.5.2.1	Port Control	35-458
35.5.2.2	Clock Control	35-460
35.5.2.3	Additional Register	35-461
35.6	Revision History	35-470
36	Advanced Driver Assistance Subsystem (ADAS)	36-1
36.1	Camera and ADC Interface (CIF)	36-3
36.1.1	Introduction	36-3
36.1.1.1	Camera and ADC Interface Functional Overview	36-3
36.1.1.2	Camera and ADC Interface Block Diagram	36-4
36.1.2	Camera and ADC Interface Functional Specification	36-6
36.1.2.1	Target Applications	36-6
36.1.2.2	Functional Description	36-10
36.1.3	CIF Registers Description	36-28
36.1.3.1	CIF Registers Overview	36-28
36.1.3.2	CIF Control Registers	36-48
36.1.3.3	ISP Programming Registers	36-55
36.1.3.4	Linear Downscaler Programming Registers	36-82
36.1.3.5	Memory Interface Programming Registers	36-84
36.1.3.6	JPEG Encoder Programming Registers	36-125
36.1.3.7	Security Watchdog Programming Registers	36-160
36.1.3.8	ISP Image Stabilization Registers	36-168
36.1.3.9	Extra Path Programming Registers	36-180
36.1.3.10	Debug Path Programming Registers	36-221
36.1.4	CIF Module Implementation	36-227
36.1.4.1	BPI_SPB Module Registers	36-228
36.1.5	CIF Programming Hints	36-238
36.1.5.1	Configuration and Shadow Registers	36-238
36.1.5.2	General Setup for Operation	36-239
36.1.5.3	Start-Stop Programming	36-241
36.1.5.4	Abort of Processing	36-242
36.1.5.5	Interrupt Handling	36-242
36.1.5.6	Reset Handling	36-246
36.1.5.7	Programming Guide	36-246
36.1.5.8	Use Case Description	36-262
36.1.5.9	Power Management	36-265
36.1.5.10	Basics on Configuration Access	36-266
36.2	Extended Memory (EMEM)	36-267
36.2.1	Block Diagram	36-267
36.2.1.1	Extended Application Memory (XAM)	36-267
36.2.1.2	Family Overview	36-267

Table of Contents

36.2.2	Operational Overview	36-268
36.2.3	Extended Memory	36-269
36.2.4	Access Error Generation	36-269
36.2.5	Tiles and RAM Address Ranges	36-269
36.2.6	EMEM Initialization	36-271
36.2.7	Usage Constraints	36-271
36.2.7.1	Clock Constraints	36-271
36.2.7.2	BBB Interface	36-271
36.2.8	Power Saving	36-271
36.2.9	EMEM Configuration	36-271
36.2.10	EMEM Interfaces	36-272
36.2.10.1	LMU Interface	36-272
36.2.10.2	BBB Slave Interface	36-273
36.2.11	Feed-Through Transactions	36-273
36.2.11.1	EMEM Register Accesses	36-273
36.2.12	Address Map	36-273
36.2.13	EMEM Register Description	36-275
36.3	FFT Engine (FFT)	36-281
36.3.1	Overview	36-281
36.3.2	Feature List	36-282
36.3.3	FFT Engine Performance	36-282
36.3.4	System Performance	36-283
36.3.4.1	Data Movement Optimisation	36-283
36.3.4.2	Double Buffering Optimisation	36-284
36.3.5	Address Spaces and Access Protection	36-284
36.3.5.1	Register Address Space	36-284
36.3.5.2	Data Space	36-284
36.3.5.3	Window Coefficient Space	36-285
36.3.6	Data Value Formats	36-285
36.3.6.1	Time and Frequency Sample Format	36-286
36.3.6.2	Window Coefficient Format	36-286
36.3.7	Window Usage	36-286
36.3.7.1	Window Parameters	36-287
36.3.8	Interrupt and DMA Operation	36-287
36.3.9	FFT Engine Registers	36-289
36.3.9.1	FFT Clock Control Register	36-290
36.3.9.2	FFT Module Identification Register	36-290
36.3.9.3	FFT Control and Status Register	36-291
36.3.9.4	FFT History Registers	36-294
36.3.9.5	FFT Debug Control Registers	36-296
36.3.9.6	FFT Kernel Reset Registers	36-300

1 TC26x B-Step Introduction

This User's Manual describes the Infineon TC26x B-Step, a 32-bit microcontroller DSP, based on the Infineon TriCore Architecture.

1.1 About this Document

This document is designed to be read primarily by design engineers and software engineers who need a detailed description of the interactions of the TC26x B-Step functional units, registers, instructions, and exceptions.

This TC26x B-Step User's Manual describes the features of the TC26x B-Step with respect to the TriCore Architecture. Where the TC26x B-Step directly implements TriCore architectural functions, this manual simply refers to those functions as features of the TC26x B-Step. In all cases where this manual describes a TC26x B-Step feature without referring to the TriCore Architecture, this means that the TC26x B-Step is a direct implementation of the TriCore Architecture.

Where the TC26x B-Step implements a subset of TriCore architectural features, this manual describes the TC26x B-Step implementation, and then describes how it differs from the TriCore Architecture. Such differences between the TC26x B-Step and the TriCore Architecture are documented in the section covering each such subject.

1.1.1 Related Documentations

A complete description of the TriCore architecture is found in the document entitled "TriCore Architecture Manual". The architecture of the TC26x B-Step is described separately this way because of the configurable nature of the TriCore specification: Different versions of the architecture may contain a different mix of systems components. The TriCore architecture, however, remains constant across all derivative designs in order to preserve compatibility.

This User's Manuals together with the "TriCore Architecture Manual" are required to understand the complete TC26x B-Step micro controller functionality.

1.1.2 Text Conventions

This document uses the following text conventions for named components of the TC26x B-Step:

- Functional units of the TC26x B-Step are given in plain UPPER CASE. For example: "The QSPI supports full-duplex and half-duplex synchronous communication".
- Pins using negative logic are indicated by an overline. For example: "The external reset pin, $\overline{\text{ESR0}}$, has a dual function."
- Bit fields and bits in registers are in general referenced as "Module_Register name.Bit field" or "Module_Register name.Bit". For example: "The Current CPU Priority Number bit field CPU_ICR.CCPN is cleared". Most of the

TC26x B-Step Introduction

register names contain a module name prefix, separated by an underscore character “_” from the actual register name (for example, “QSPI0_GLOBALCON”, where “QSPI0” is the module name prefix, and “GLOBALCON” is the kernel register name). In chapters describing the kernels of the peripheral modules, the registers are mainly referenced with their kernel register names. The peripheral module implementation sections mainly refer to the actual register names with module prefixes.

- Variables used to describe sets of processing units or registers appear in mixed upper and lower cases. For example, register name “MOFCRn” refers to multiple “MOFCR” registers with variable n. The bounds of the variables are always given where the register expression is first used (for example, “n = 0-255”), and are repeated as needed in the rest of the text.
- The default radix is decimal. Hexadecimal constants are suffixed with a subscript letter “H”, as in 100_H. Binary constants are suffixed with a subscript letter “B”, as in: 111_B.
- When the extent of register fields, groups register bits, or groups of pins are collectively named in the body of the document, they are represented as “NAME[A:B]”, which defines a range for the named group from B to A. Individual bits, signals, or pins are given as “NAME[C]” where the range of the variable C is given in the text. For example: CFG[2:0] and SRPN[0].
- Units are abbreviated as follows:
 - **MHz** = Megahertz
 - **μs** = Microseconds
 - **kBaud, kbit** = 1000 characters/bits per second
 - **MBaud, Mbit** = 1000000 characters/bits per second
 - **Kbyte, KB** = 1024 bytes of memory
 - **Mbyte, MB** = 1048576 bytes of memory

In general, the k prefix scales a unit by 1000 whereas the K prefix scales a unit by 1024. Hence, the Kbyte unit scales the expression preceding it by 1024. The kBaud unit scales the expression preceding it by 1000. The M prefix scales by 1,000,000 or 1048576, and μ scales by .000001. For example, 1 Kbyte is 1024 bytes, 1 Mbyte is 1024 × 1024 bytes, 1 kBaud/kbit are 1000 characters/bits per second, 1 MBaud/Mbit are 1000000 characters/bits per second, and 1 MHz is 1000000 Hz.
- Data format quantities are defined as follows:
 - **Byte** = 8-bit quantity
 - **Half-word** = 16-bit quantity
 - **Word** = 32-bit quantity
 - **Double-word** = 64-bit quantity

1.1.3 Reserved, Undefined, and Unimplemented Terminology

In tables where register bit fields are defined, the following conventions are used to indicate undefined and unimplemented function. Furthermore, types of bits and bit fields are defined using the abbreviations as shown in [Table 1](#).

Table 1 Bit Function Terminology

Function of Bits	Description
Unimplemented, Reserved	Register bit fields named 0 indicate unimplemented functions with the following behavior. <ul style="list-style-type: none"> • Reading these bit fields returns 0. • These bit fields should be written with 0 if the bit field is defined as r or rh. • These bit fields have to be written with 0 if the bit field is defined as rw. These bit fields are reserved. The detailed description of these bit fields can be found in the register descriptions.
rw	The bit or bit field can be read and written.
rwh	As rw, but bit or bit field can be also set or reset by hardware.
r	The bit or bit field can only be read (read-only).
w	The bit or bit field can only be written (write-only). A read to this register will always give a default value back.
rh	This bit or bit field can be modified by hardware (read-hardware, typical example: status flags). A read of this bit or bit field give the actual status of this bit or bit field back. Writing to this bit or bit field has no effect to the setting of this bit or bit field.
s	Bits with this attribute are “sticky” in one direction. If their reset value is once overwritten by software, they can be switched again into their reset state only by a reset operation. Software cannot switch this type of bit into its reset state by writing the register. This attribute can be combined to “rws” or “rwhs”.
f	Bits with this attribute are readable only when they are accessed by an instruction fetch. Normal data read operations will return other values.

1.1.4 Register Access Modes

Read and write access to registers and memory locations are sometimes restricted. In memory and register access tables, the terms as defined in [Table 2](#) are used.

TC26x B-Step Introduction

In general, if an access type is not permitted under these rules (e.g. attempted write to R, attempted user mode access to SV, attempted access to E without Endinit, etc.) then a Bus Error will result, unless the access is also marked as nBE (or otherwise stated in the specific module chapter).

Table 2 Access Terms

Symbol	Description
U	Access Mode: Access permitted in User Mode 0 or 1.
	Reset Value: Value or bit is not changed by a reset operation.
SV	Access permitted only in Supervisor Mode.
R	Read-only register.
32	Only 32-bit word accesses are permitted to this register/address range.
32/16	Only 32-bit or 16-bit accesses are permitted to this register/address range.
CEx	CPUx Endinit protected register/address.
SE	Safety Endinit protected register/address.
E	Any CPU Endinit-protected register/address.
P (or P0 / P1)	Access Enable Register protected register/address. (ACCEN0/1)
PW	Password-protected register/address.
NC	No change, indicated register is not changed.
BE	Indicates that an access to this address range generates a Bus Error.
nBE	Indicates that no Bus Error is generated when accessing this address range, even though it is either an access to an undefined address or the access does not follow the given rules.
nE	Indicates that no Error is generated when accessing this address or address range, even though the access is to an undefined address or address range. True for CPU accesses (MTCR/MFCR) to undefined addresses in the CSFR range.

1.1.5 Abbreviations and Acronyms

The following acronyms and terms are used in this document:

ADC	Analog-to-Digital Converter
ALU	Arithmetic and Logic Unit

ASCLIN	Asynchronous/Synchronous Serial Controller with LIN
BCU	Bus Control Unit
BROM	Boot ROM & Test ROM
CAN	Controller Area Network
CAPCOM	Capture Compare Unit
CIF	Camera and ADC Interface
CPU	Central Processing Unit
CRC	Cyclic Redundancy Code
CSA	Context Save Area
CSFR	Core Special Function Register
CCU6	Capture Compare Unit 6
CCU	Clock Control Unit
DAP	Device Access Port
DAS	Device Access Server
DCACHE	Data Cache
DFLASH	Data Flash Memory
DMA	Direct Memory Access
DMBI	Data Memory Bus Interface
DMI	Data Memory Interface
DRLB	Data Read Line Buffer
DSPR	Data Scratch Pad RAM
DS-ADC	Delta-Sigma Analog to Digital Converter
EBU	External Bus Interface
ECC	Error Correction Code
EMI	Electro-Magnetic Interference
E-Ray	Flexray Controller
EtherMAC	Ethernet Media Access Controller
EVR	Embedded Voltage Regulator
FCE	Flexible CRC Engine
FPB	Flash Prefetch Buffer
FFT	Fast Fourier Transform Accelerator
FM-PLL	PLL with Frequency Modulation support

FPI	Flexible Peripheral Interconnect (Bus protocol)
FPU	Floating Point Unit
GPIO	General Purpose Input/Output
GPR	General Purpose Register
GPT12	General Purpose Timer 12
GTM	Generic Timer Module
HSSL	High Speed Serial Link
I2C	Inter-Integrated Circuit Controller
I/O	Input / Output
IOM	I/O Monitor Unit
JTAG	Joint Test Action Group = IEEE1149.1
LMU	Local Bus Memory Unit
MCHK	Memory Checker module
MBIST	Memory Build In Self Test
MMU	Memory Management Unit
MSB	Most Significant Bit
MSC	Micro Second Channel
MTU	Memory Test Unit
MultiCAN+	Enhanced Multiple CAN controller
NC	Not Connected
NMI	Non-Maskable Interrupt
NVM	Non Volatile Memory
OCDS	On-Chip Debug Support
OVRAM	Overlay Memory
PLL	Phase Locked Loop
PCACHE	Program Cache
PFLASH	Program Flash Memory
PMBI	Program Memory Bus Interface
PMI	Program Memory Interface
PMU	Program Memory Unit
PSI5	Peripheral Sensor Interface
PSI5-S	Peripheral Sensor Interface with Serial Interface to Phy

PSPR	Program Scratch Pad RAM
QSPI	Queued SPI Controller
RAM	Random Access Memory
RISC	Reduced Instruction Set Computing
SBCU	System Peripheral Bus Control Unit
SCU	System Control Unit
SCR	8-bit Standby Controller
SENT	Single Edge Nibble Transmission
SFR	Special Function Register
SMU	Safety Management Unit
SPB	System Peripheral Bus (based on FPI protocol)
SPD	Single Pin DAP
SPI	Synchronous Serial Controller
SRI	Shared Resource Interconnect
SRAM	Static Data Memory
SRN	Service Request Node
STM	System Timer
SWD	Supply Watchdog
TC1.6P	TriCore CPU 1.6 (High Performance variant)
TC1.6E	TriCore CPU 1.6 (High Efficiency variant)
VADC	Versatile Analog-to-Digital Converter
WDT	Watchdog Timer
XBar, XBar_SRI	Cross Bar Interconnect, based on the Shared Resource Interconnect protocol

TC26x B-Step Introduction**1.2 System Architecture of the TC26x B-Step**

The TC26x B-Step combines three powerful technologies within one silicon die, achieving new levels of power, speed, and economy for embedded applications:

- Reduced Instruction Set Computing (RISC) processor architecture
- Digital Signal Processing (DSP) operations and addressing modes
- On-chip memories and peripherals

DSP operations and addressing modes provide the computational power necessary to efficiently analyse complex real-world signals. The RISC load/store architecture provides high computational bandwidth with low system cost. On-chip memory and peripherals are designed to support even the most demanding high-bandwidth real-time embedded control-systems tasks.

Additional high-level features of the TC26x B-Step include:

- Efficient memory organization: instruction and data scratch memories, caches
- Serial communication interfaces – flexible synchronous and asynchronous modes
- Multiple channel DMA Controller – DMA operations and interrupt servicing
- Flexible interrupt system – configurable interrupt priorities and targets
- Flexible CRC Engine
- General-purpose timers
- High-performance on-chip buses
- On-chip debugging and emulation facilities
- Flexible interconnections to external components
- Flexible power-management

In addition, the following Advanced Driver Assistance System features are available in products derivatives marked as “ADAS” in the tables above.

- Camera Interface
- Extended Memory
- FFT accelerator

The TC26x B-Step is a high-performance microcontroller with two TriCore CPUs, program and data memories, buses, bus arbitration, interrupt system, DMA controller and a powerful set of on-chip peripherals. The TC26x B-Step is designed to meet the needs of the most demanding embedded control systems applications where the competing issues of price/performance, real-time responsiveness, computational power, data bandwidth, and power consumption are key design elements.

The TC26x B-Step offers several versatile on-chip peripheral units such as serial controllers, timer units, and analog-to-digital converters. Within the TC26x B-Step, all these peripheral units are connected to the TriCore CPUs / system via the System Peripheral Bus (SPB) and the Local Memory Bus (SRI). A number of I/O lines on the TC26x B-Step ports are reserved for these peripheral units to communicate with the external world.

TC26x B-Step Introduction

1.2.1 TC26x B-Step Block Diagram

Figure 1 shows the block diagram of the TC26x B-Step. Please note that not all features that are shown in the block diagram are available in all TC26x B-Step package variants.

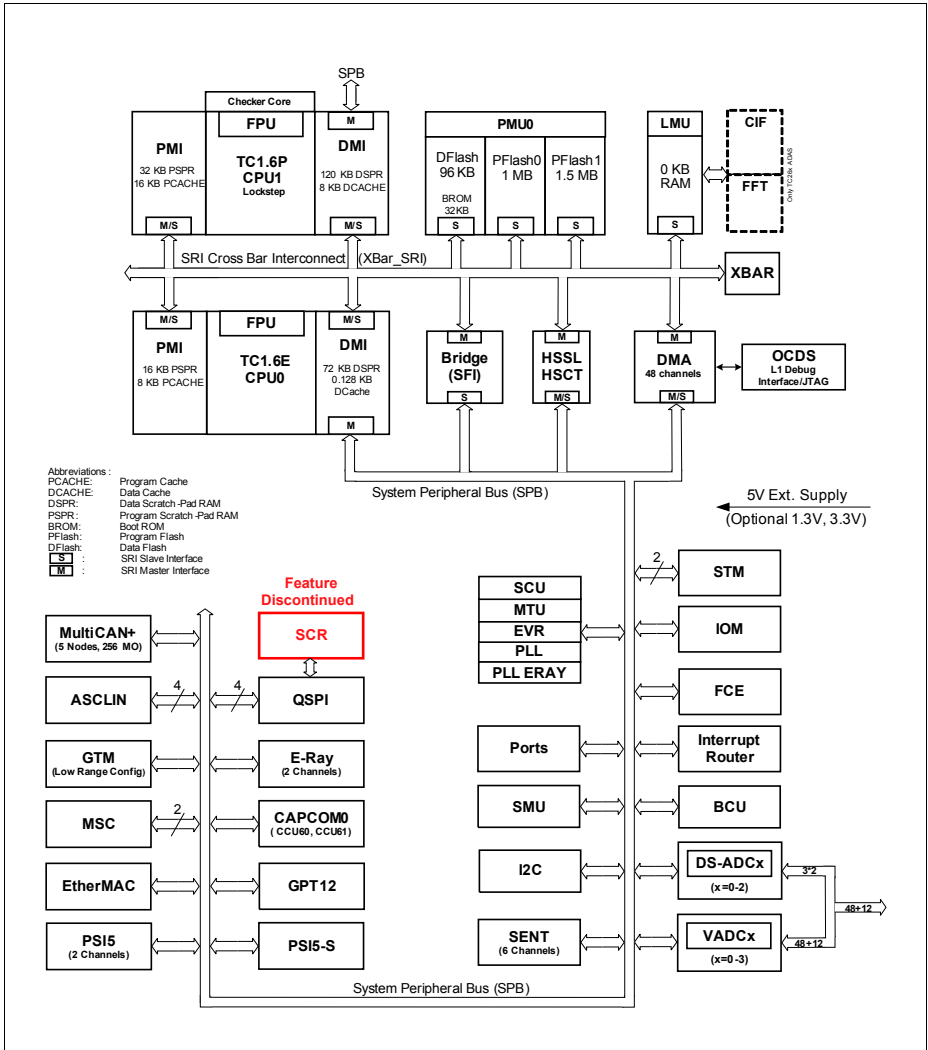


Figure 1 TC26x Block Diagram

1.3 On-Chip Debug Support (OCDS)

AURIX devices contain powerful resources for debugging and performance optimization. They provide high visibility and controllability of software, hardware and system, especially also under hard real-time constraints. The resources are either embedded in specific modules (e.g. breakpoint logic of CPUs) or part of the central CERBERUS module.

Figure 1-1 shows the AURIX family concept of the OCDS with all potentially usable pins. The same OCDS function is always on the same port pin including full DAP/JTAG functionality but the number of TGI/TGO trigger pins depends on the device type.

Trace functionality is available for TC29x with miniMCDS, for all other AURIX devices only with the associated Emulation Device (ED) with MCDS.

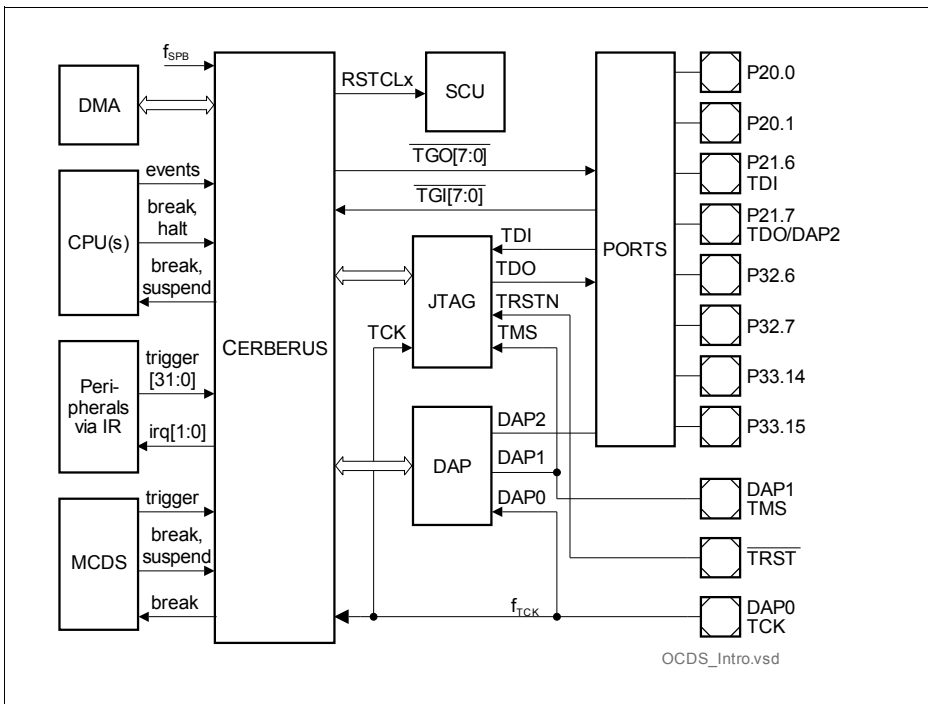


Figure 1-1 OCDS Block Diagram (Family Concept)

When acting as bus master CERBERUS shares the master interfaces of the DMA (using a different master tag) to access memories and SFRs attached to the SPB or SRI via the shortest possible path.

1.3.1 Feature List

Please also refer to debug specific features in other modules, e.g.

- Eight hardware breakpoints for TriCore, based on instruction or data address.
- Unlimited number of software breakpoints (DEBUG instruction).
- Trigger generated by the access to a specific bus address by any bus master
- Peripheral trigger and trace with OCDS Trigger Bus (OTGB)
 - Peripherals provide vectors (Trigger Sets) of their most interesting signals
 - Signals can be routed to trigger pins
 - Flexible tracing of signal vectors from one or two peripherals in parallel (ED)
- Dedicated interrupt resources to handle debug events, both local inside the CPUs (breakpoint trap, software interrupt) and global (triggered by CERBERUS), e.g. for implementing monitor programs

Central functions implemented by CERBERUS

- Run/stop and single-step execution independently for each CPU.
- Run/stop and time-step execution of the complete device using the Trigger Switch.
- Automatic suspension of the SCU watchdogs when the device is halted by the tool.
- All kinds of reset can be requested using only the tool interface.
- Halt-after-Reset for repeatable debug sessions.
- Tool access to all SFRs and internal memories independent of the CPUs.
- Bus priority of CERBERUS can be chosen dynamically to minimize real-time impact.
- Up to 8 package pins can be used optional as with Trigger In/Out (TGI/TGO).
- Central OCDS Trigger Switch (OTGS) with 7 independent Trigger Lines to collect debug events from various sources (all CPUs, all interrupt requesters, bus controllers, several complex peripherals, MCDS, trigger input pins) and distribute them selectively to all CPUs and trigger output pins (extension of the former Break Switch concept).
- Central Suspend Switch using up to three Lines of the Trigger Switch infrastructure. This allows to selectively suspend all or only part of the CPUs and peripherals instead of halting them as reaction to any debug event.
- Access to all OCDS resources also for the CPUs themselves for debug tools integrated into the application code.
- Triggered Transfer of data for simple variable tracing.
- A dedicated trigger bank (TRIG) with 96/512 independent status bits is provided to post requests at a central location from application code to the tool.
- The tool is notified automatically when the trigger bank is updated by any processor. No polling via a system bus is required.

Tool Interfaces

Several options exist for the communication channel between tools and TC26x B-Step:

- DAP and JTAG are clocked by the tool.
- Two pin DAP (Device Access Port) protocol for long connections or noisy environments.
- Three unidirectional pin DAP mode for off-chip transceiver integration (e.g. LVDS).
- Three pin DAP Wide Mode for high bandwidth needs.
- DAP bit clock can have any frequency up to 160 MHz.
- 15 MByte/s for block read or write, 30 MByte/s in Wide Mode
- Optimized random memory accesses (read word within 0.5 μ s at 160 MHz).
- CAN (plus software linked into the application code) for embedded purposes with lower bandwidth requirements.
- Four pin JTAG (IEEE 1149.1) for standard manufacturing tests.
- Hot attach (i.e. physical disconnect/reconnect of the host connection without reset) for all interfaces.
- Infineon standard DAS (Device Access Server) implementation for seamless, transparent and parallel tool access over any supported interface.
- Lock mechanism to prevent unauthorized tool access to application code.
- DAP over CAN Messages (DXCM) u, TC23x/22x only)

FAR Support

To efficiently locate and identify faults after integration of an AURIX device into a system special functions are available:

- Boundary Scan (IEEE 1149.1) via JTAG or DAP.
- SSCM (Single Scan Chain Mode) for structural scan testing of the chip itself.
- DXCPL (DAP over CAN Physical Layer) via CAN pins (AP32264)

1.3.2 Family Overview

The OCDS architecture and features are very consistent over the whole AURIX family. This makes it easy for tool partners and users to switch between different devices. [Table 1-1](#) lists all OCDS and the main Emulation Device differences of the devices, and [Table 1-2](#) the JTAG IDs. The associated ED step has the same JTAG ID.

Table 1-1 OCDS Differences of AURIX Devices

Feature	TC29x	TC27x	TC26x	TC24x	TC23x	TC22x/1x
TRIG triggers	512	512	512	512	96	96
DAP Over CAN Msg. (DXCM)	-	-	-	-	yes	yes
miniMCDs	yes	-	-	-	-	-
Emulation Device (ED)	9xED	7xED	6xED	(6xED)	3xED	(3xED)

Table 1-1 OCDS Differences of AURIX Devices (cont'd)

Feature	TC29x	TC27x	TC26x	TC24x	TC23x	TC22x/1x
ED packages B=BGA, Q=QFP	B 292 B 416 B 516	B 292 Q 176	Q 176 Q 144	Q 144	Q 144	Q 144
ED's TDI pin replaced by VDDPSB	-	Q 176	Q 176 Q 144	Q 144	-	-

Table 1-2 JTAG IDs of AURIX Devices

	A-Step	B-Step	C-Step	D-Step
TC29x	101E9083 _H	201E9083 _H		
TC27x	201DA083 _H	301DA083 _H	401DA083 _H	501DA083 _H
TC26x	101E8083 _H	201E8083 _H		
TC24x	101EA083 _H			
TC23x	10200083 _H			
TC22x	10201083 _H			
TC21x	10202083 _H			

1.3.3 Tool Interface Recommendations

AURIX devices are well supported by many tool partners for different types of tools. Standard tool interface for debug, measurement and calibration is DAP due to its reduced pin-count, higher performance (3-6x) and higher robustness (CRC) than JTAG. Please note that the full "JTAG" boundary scan functionality is also available with DAP, it is supported however only by specific tool providers. Please refer to application note AP24003 for more information about the standard DAP connector and board design for high speed DAP. [Table 1-3](#) lists all pins and considerations for connecting tools.

Table 1-3 Tool Relevant Device Pins of AURIX Family

Pins	Remark
$\overline{\text{TRST}}$	DAP: Has to be high at $\overline{\text{PORST}}$ pin release. Can be statically connected to VDDP3 (e.g. in the development phase only) with these effects: <ul style="list-style-type: none"> • Causes a static current due to pull-down element on chip • VDDSB and VDDPSB supplies need to be ensured • DXCPL is disabled JTAG: Needs to be controlled by the tool via the tool connector.
DAP0/TCK	DAP: Please consider AP24003 for high speed DAP
DAP1/TMS	
$\overline{\text{DAP2/TDO}}$ $\overline{\text{TGI3/TGO3}}$	DAP: Needed for three pin modes like high bandwidth Wide Mode. The standard DAP connector (AP24003) allows to use this pin on demand either for Wide Mode e.g. for measurement or as trigger pin for system debugging.
$\overline{\text{TDI/VDDPSB}}$ $\overline{\text{TGI2/TGO2}}$	For certain EDs in QFP packages (Table 1-1) this pin is the VDDPSB supply pin and needs to be supplied with 3.3 V (e.g. VDDP3). Please note for the “TDI is VDDPSB” case: <ul style="list-style-type: none"> • JTAG is not available for such EDs • VDDP3 connection can be hardwired. This is compatible for PD and ED, however TGI2/TGO2 is then also unavailable for the PD.
VDDSB	1.3 V supply of the ED memory. This supply can be separate of VDD if standby functionality is needed. Please consider the additional VDDSB current for your supply.
VDDPSB	3.3 V supply for the DAP pins of EDs. Can be connected to VDDP3 or supplied by the tool.
$\overline{\text{TGIx/TGOx}}$	Optional trigger pins, overlaid to port pins. Availability depends on device and package type.
AGBT_xyz	AGBT high-speed serial pins in the center ball matrix of EDs in BGA packages. Please connect to VSS if no AGBT is needed.

Note: Please ensure that VDDSB and VDDPSB are always supplied in regular operation (VDD supplied, $\overline{\text{TRST/PORST}}$ inactive) to avoid internal cross currents which can damage the ED device. For instance connect VDDSB to VDD and VDDPSB to VDDP3.

For more information please contact your Infineon support.

1.3.4 Debug Access Server (DAS)

The DAS API provides an abstraction of the physical device interface for tool access. The key paradigm of DAS is to read or write data in one or several address spaces of the target device.

DAS Features

- Standard interface for all types of tools
- Efficient and robust methods for data transfer
- Standardized system security support (authorization)
- Several independent tools can share the same physical interface
- Product chip address space is represented with DAS address map 0, EEC with 1
- Infineon's miniWiggler supports DAP, JTAG, SWD and SPD

DAS is not device specific. It can be used for all Infineon 8-, 16- and 32-bit microcontrollers with DAP, JTAG, SWD, or SPD interface. For more information please refer to www.infineon.com/DAS.

1.4 Emulation Device (ED)

The TC26xED is the Emulation Device (ED) of the TC26x Production Device (PD) and its feature-reduced derivatives. The Emulation Device comprises the unchanged Product Chip Part (SoC) and the Emulation Extension Chip (EEC) part (Figure 1-2). Both are on the same silicon. The ED is available in similar package variants as TC26x for having a single PCB with minimal overhead for supporting PD and ED.

1.4.1 Block Diagram

Figure 1-2 shows the ED block diagram. The Product Chip Part is reduced to the directly connected modules only in the drawing. On the EEC part, the AGBT module is not shown in this block diagram.

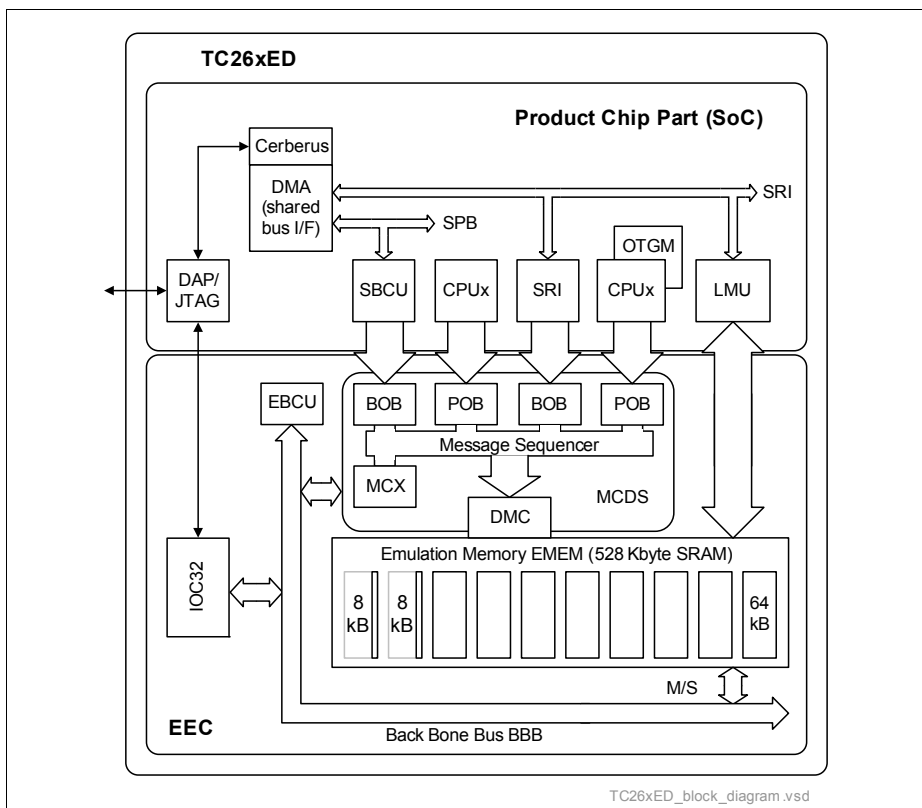


Figure 1-2 Block Diagram TC26xED

Table 1-4 TC26xED Components (Figure 1-2)

Component	Definition
AGBT	Aurora GigaBit Trace module on EEC
BBB	Back Bone Bus (same FPI protocol as SPB)
BOB	Bus Observation Block. Trace and trigger logic within MCDS
Cerberus	Central debug and tool access control unit. Includes the OTGS.
DMC	Debug Memory Controller
EBCU	Emulation/BBB Bus Control Unit
ED	Emulation Device for calibration, measurement and debug
EEC	Emulation Extension Chip (product chip part + EEC = ED)
EMEM	Emulation Memory (calibration and trace memory). Provides also together with LMU the bus bridge functionality between SRI and BBB.
FPI	Flexible Peripheral Interconnect, the protocol of SPB and BBB buses
LMU	Local Memory Unit, SRI slave for all non-core RAM
MCDS	Multi Core Debug Solution
MCX	Multi Core Cross connect
OTGM	OCDS Trigger Multiplexer. Collects interrupt and peripheral trace and trigger signals.
OTGS	OCDS Trigger Switch. Routes triggers to pins, for halt/suspend, etc.
POB	Processor Observation Block
SBCU	SPB Bus Control Unit
SoC	System on Chip (used also for “product chip part”)
SPB	System Peripheral Bus
SRI	Shared Resource Interconnect cross bar
TCM	Trace and Calibration memory part of EMEM
XTM	Extended Trace Memory part of EMEM

The Emulation Memory on the EEC is used for two conceptually different purposes: Calibration and Tracing. The size allocation for both parts can be configured in the Emulation Memory (EMEM) module.

For calibration, RAM partitions are mapped into the address ranges of the CPUs, optionally replacing parts of the TC26x’s local Flash. The feature is described in detail in the Overlay section of the TC26x manual.

Tracing on the other hand is a non intrusive tool to aid the debugging process. Matching elements from the MCDS module are provided on the EEC to translate the signals - routed there from the TC26x's CPUs and other sources - into meaningful messages. The messages are buffered and can then be uploaded to the tool via a host interface, e.g. DAP/JTAG.

1.4.2 Feature List

This section lists the features of TC26xED.

TC26xED Applications

- Software development
 - Debugging
 - Performance analysis and optimization
- Calibration
- Measurement
- Rapid prototyping

TC26xED General Features

- ED and production device have identical behavior
- ED has a package with a footprint compatible to production device
- Minimum number of ED specific pins
- Full access to the EEC part via the regular DAP/JTAG package pins
- No external emulator hardware required other than DAP/JTAG interface
- Full access to the EEC for tool software (monitor), running on TriCore, for e.g. CAN based calibration, measurement or debug
- Protection against reverse engineering by competitors and against manipulation, both for production and emulation device (field trials)
- High-speed Aurora GigaBit Trace (AGBT) interface

Emulation Memory Features

- Emulation Memory (EMEM) size is 528 KB
- EMEM has eight 64 KB TCM tiles and two 8 KB XTM tiles
- Full Emulation Memory can be used for calibration, code, constants or data storage
- Up to 512 KB can be used for trace buffering
- Continuous trace via DAP requires just the two XTM tiles (only one for AGBT)
- Support of independently operating calibration and debug tools
- FIFO functionality for continuous trace (EMEM address triggers in MCDS)
- Emulation Memory is also mapped into the address range of TriCore
- Emulation Memory can be overlaid to Flash
- Code and data fetch from Emulation Memory
- Data retention of RAM during power down by isolated standby power supply

- ECC with SECCDED (Single Error Correction, Double Error Detection)

Measurement Features

- Highly efficient SW triggering via DAP interface (TRIG)
- Fine Grained Trace Qualification (FGTQ) for trace based measurement via DAP
- Aurora GigaBit Trace (AGBT) interface for high-end trace based measurement

Debug Features

- TriCore program trace (absolute, relative, functions only)
- Continuous Compact Function Trace (CFT) via DAP
- TriCore data trace (no register file trace)
- Parallel trace of two CPUs, two SRI slaves and the SPB bus
- Full visibility of internal peripheral bus (SPB)
- Full visibility of up to two arbitrary SRI slaves
- Time aligned trace of all sources
- Trace of internal states and signals of complex peripherals
- Trace of interrupt and DMA requests and processing
- Breakpoints and watch points based on common event generation logic
- Magnitude comparators working on instruction pointers and memory addresses:
A <= IP <= B
- Masked magnitude comparators working on the data busses: DATA = "xxxx55xx"
- Sequential event logic: Counters driven by events and equipped with limit comparators are used as event sources again for breakpoint or trace qualification
- Optimized compression of buffered trace data
- Highly sophisticated complex qualification- and trigger mechanism
- Pre- and post event trace buffering ("digital oscilloscope")
- Performance counters
- Concurrent trace logging and trace data acquisition up to the bandwidth of the used host interface
- Central time stamp unit to correlate traces from different CPUs and other sources
- Halt the system or parts of it when trace memory is full
- Regular and modular structure of the control blocks and registers
- Trace debug unit power reduction modes with clock gating
- Trace data in EMEM can be decoded after unsolicited PORST
- Output of continuous trace over Aurora GigaBit Trace interface

1.4.3 Comparison to TC27xED

The TC26xED is intentionally built as close to the predecessors of the AURIX family as feasible.

Changes

- Trace Multiplexer on Product Chip Part for two instead of three CPUs
- EMEM size reduced from 1024 KB to 528 KB
- EMEM has eight 64 KB TCM tiles and two 8 KB XTM tiles
- Changed address map due to EMEM size
- Changed current consumption

1.4.4 Trace Based Measurement

Traditional trace based measurement or Data Acquisition (DAQ) requires for the device a high speed trace interface. This interface is used to output the write data and address information of all CPUs. Already for single CPU devices, this approach is a challenge, since the CPU will create ca. 64 bit data per clock cycle in case of writes. Taking into account that the CPU writes on average just every 10 to 20 cycles, this results in roughly 1 Gbit/s per CPU (Figure 1-3). In the Tool HW, this trace data is then used to create mirrored RAM(s) of the device internal RAM(s). These mirrored RAM(s) are then the basis for collecting the DAQ data at given points of time. The resulting DAQ net data rate is then in the range of Mbyte/s.

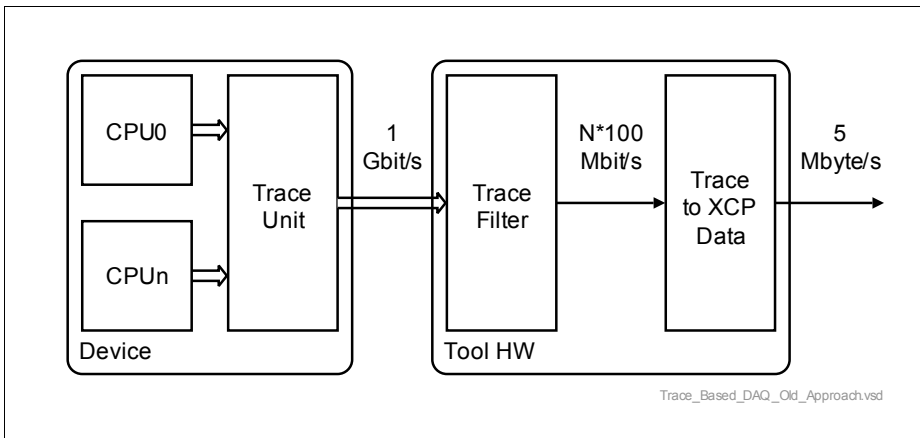


Figure 1-3 Trace Based Measurement - Traditional Approach

The basic idea for avoiding the high-bandwidth trace port is to trace only relevant CPU writes. TC26xED has for that purpose the Fine Grained Trace Qualification (FGTQ), which allows to select an arbitrary number of relevant data in RAM with 1, 2, 4, 8, 16 or 32 byte address granularity. In combination with the high DAP performance, the net XCP DAQ data rate will be up to 30 Mbyte/s. For even higher demands FGTQ can be combined with Aurora GigaBit Trace (AGBT).

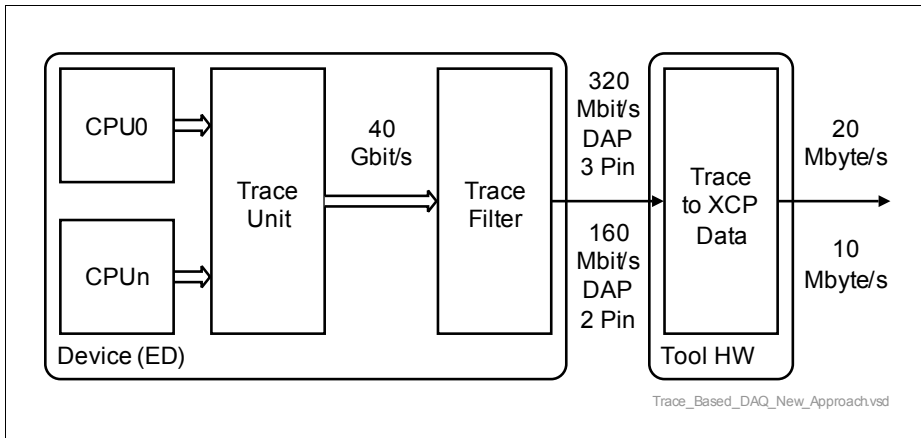


Figure 1-4 Trace Based Measurement - New Approach

1.4.5 ED Design and Layout

The TC26x design is extended on top-level with the EEC part on the same silicon (**Figure 1-2**). A clear requirement and guideline from this implementation is the reuse without changes of the SoC Product Chip Part, which guarantees complete compatibility between emulation device and production device. The ED chip layout is as similar as possible to the product chip.

ED as Bare Die for Hybrid ECUs

TC26xED will be also available as bare die for mounting on ceramic hybrids. The hybrid layout needs to accommodate the enlarged die area and the shifted pads with a combi-layout.

1.4.6 Emulation System Components

1.4.6.1 Emulation Memory (EMEM)

The Emulation Memory contains RAM blocks (tiles) which can be alternatively used as Trace or Calibration Memory (TCM). The size of a tile is 64 Kbyte.

Features

- 528 Kbyte SRAM in total
- Standby power supply

- Calibration overlay block size configurable from 32 byte to 128 Kbyte
- Trace Memory size can be configured from 0 to 512 Kbyte in 64 Kbyte steps
- Two additional XTM tiles with just 8 Kbyte physical RAM each for supporting trace based measurement without reducing the Calibration Memory size
- ECC with SECDED (Single Error Correction, Double Error Detection)

1.4.6.2 Calibration Memory

The Calibration Memory uses two interfaces. One is to the Product Chip Part (LMU Interface), the other to the Backbone Bus (BBB) of the EEC.

LMU Interface

The Calibration Memory can be accessed from the Product Chip Part directly over the LMU Interface. The Calibration Memory is always accessible directly within a specific address range. Additionally read accesses to the Program Flash can be redirected to the Calibration Memory. In this overlay mode, read accesses to up to 32 different address ranges, defined by base and size, are forwarded to the Calibration Memory instead of the Flash. For further information please refer to “Data Access Overlay (OVC)”.

The overlay mechanism is limited to data accesses, code fetch is not supported. The overlay mechanism replaces the data of load/store accesses only. Note that immediate constants, embedded in the program code are not overlaid. If program code is to be executed from the Calibration Memory, it needs to be linked/located to the address range where the Calibration Memory is mapped directly.

EEC Backbone Bus Interface

- Random read and write of Calibration Memory in parallel to LMU accesses
- The accesses from the LMU interface have priority

1.4.6.3 Trace Memory

Trace Memory is operated at a much higher access rate than Calibration Memory. It is a frequent case, that trace data is recorded over a longer period with a sustained rate of more than 50% of the trace interface’s maximum bandwidth. Since power dissipation scales with the access rate and memory size, this behavior affects the maximum Trace Memory size and the ambient temperature and power supply for this mode.

Features

- Trace bandwidth max. 40 GBit/s (MCDS clock x memory bus width of 256 bits)
- Flexible trace operation modes:
 - Tracing from event
 - Tracing up to event
 - Tracing around event (buffer holds traces up to and after event)

- If more than one memory tile are allocated to tracing, the tiles currently not written can be uploaded to the emulation tool for continuous tracing. In this mode the trace buffer depth is only limited by the transfer speed of the used interface.

1.4.7 MCDS

The prime target for the EEC is to cater for the real-time tracing of the TC26x's CPUs. Nonetheless all event generating logic can be used for breakpoint generation via the central break switch as well. It should be noted that there is a certain latency using this route.

Processor Observation Block (POB) for TriCore

- 6 range comparators on the IP (fan type)
- 8 range comparators on the write or read address
- 4 masked range comparators on the data value written
- One fine grained address trigger
- Dedicated trigger inputs for TriCore's 8 OCDS L1 comparators
- Thread awareness: 2 comparators to restrict tracing to certain threads only
- Trace of data read and write address
- Trace of the data value written
- Full non-cached data access visibility with trace at SRI and SPB bus
- Watch point traces based on all before mentioned comparators
- Debug status message based on the execution mode of the CPU
- Complete program trace (absolute, relative, subroutine only)
- Dedicated programmable trace enable generator for each trace unit, using all local comparators as potential sources

Bus Observation Block (BOB) for System Peripheral Bus (SPB)

- 4 range comparators on the SPB address
- 4 masked range comparators on the SPB data bus
- 4 masked range comparators on the SPB operation code/mastership
- Watch point traces based on all before mentioned comparators
- Ownership trace (bus master) derived from bus arbiter
- Data trace with or without address information for Read and/or Write accesses
- Dedicated programmable trace enable generator for each trace unit, using all local comparators as potential sources

Bus Observation Block (BOB) for SRI Cross Connect

Two of the slaves can be observed concurrently.

- 2*4 range comparators on the SRI transaction address
- 2*4 masked range comparators on the SRI transaction data

- 2*4 masked range comparators on the SRI transaction originator (mastership)
- 2*1 fine grained address triggers
- Watch point traces based on all before mentioned comparators
- Ownership trace derived from transaction originator
- Data trace with or without address information for Read and/or Write access
- Dedicated programmable trace enable generator for each trace unit, using all local comparators as potential sources

Multi Core Cross-Connect (MCX)

- Gets 4 programmable pretrigger signals from each observation block
- One dedicated global trace enable for each trace unit
- 32 universal 16 bit counters, using programmable combinations of pretriggers as count and clear signals
- Performance counters can be configured for ca. 80 sources
- Limit comparators based on these counters, generating further pretriggers
- Global trace enables are sums of products (four multi-input ANDs ORed together)
- Global break generation based on all available pretriggers - including the counters'
- Bidirectional interface to the break switch on the Product Chip Part
- Global time stamp messages, based on emulation or reference clock.

1.4.8 DAP/JTAG based Tool Interface (IOC32)

- Uses the DAP/JTAG pins of the Product Chip Part (IO Client concept)
- Generic serial DAP/JTAG based link to access the complete address space
- Very robust, allows to debug also unfriendly systems (power down, reset etc.)
- External tool controls all transactions
- Bit clock up to 160 MHz for DAP and optional two DAP data pins in Wide Mode
- Block read and write support
- Hot attach to a running system
- Internal user software can lock the interface for security reasons

1.4.9 Aurora GigaBit Trace (AGBT)

The AGBT module outputs the MCDS trace data over a 2.5 Gbit/s differential link, which is compliant with the Aurora I/F of Xilinx FPGAs. [Figure 1-5](#) shows the block diagram of the ED with the AGBT module. The AGBT module uses an 8 Kbyte XTM tile of the EMEM as FIFO buffer, all other EMEM tiles can be used e.g. for Calibration.

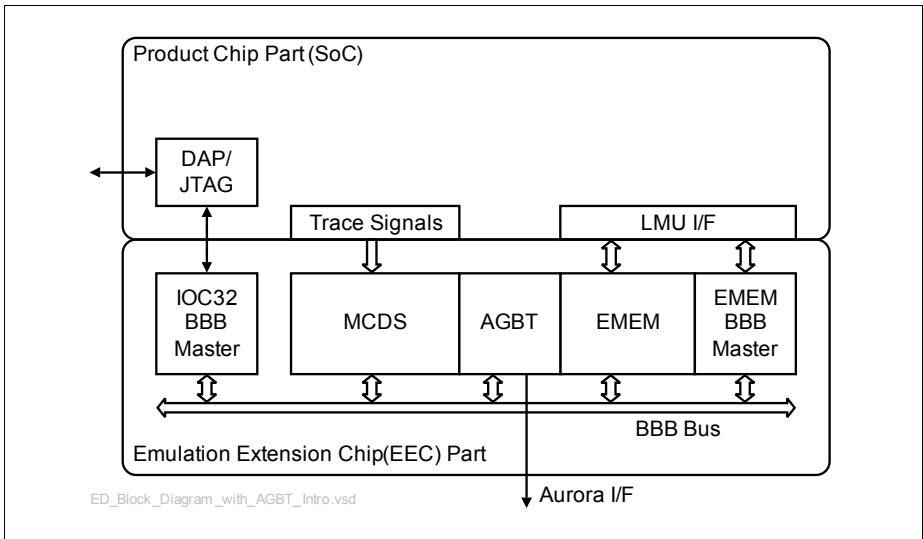


Figure 1-5 ED with AGBT Module

2 On-Chip System Buses and Bus Bridges

The TC26x B-Step has two independent on-chip buses:

- Shared Resource Interconnect (SRI)
- System Peripheral Bus (SPB)

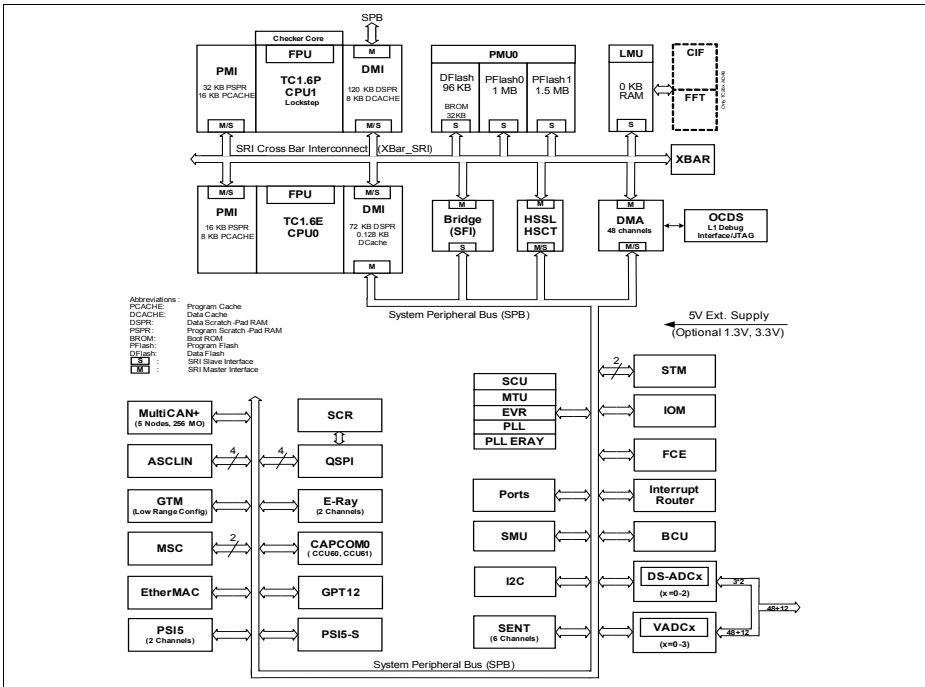


Figure 2-1 On Chip Buses in TC26x B-Step Processor Subsystem

The SRI connects the TC1.6 CPUs, the main high bandwidth peripherals and the general purpose DMA modules to its local resources for instruction fetches and data accesses.

The System Peripheral Bus connects the TC1.6 CPUs and the general purpose DMA modules to the medium and low bandwidth peripherals. The TC1.6P CPU has access to the SPB via the SRI to SPB bridge.

Note: The TC1.6 has one SRI slave interface that provides access to the TC1.6 SRAMs, SFRs and CSFRs.

On-Chip System Buses and Bus Bridges**2.1 What is new**

Major differences of the TC26x B-Step XBar Bus System architecture compared to previous TC1.6 based products:

- Empty list (this is the initial version of Aurix product)

2.2 SRI Crossbar (XBar_SRI)

2.2.1 Introduction

The Shared Resource Interconnection (SRI) is the high speed system bus for TriCore1.6.x CPU based devices. The central module of the interconnect is the XBar_SRI which connects all components in one SRI system. The XBar_SRI handles, arbitrates and forwards the communication between all connected SRI-Master and SRI-Slave peripherals.

The XBar_SRI supports parallel transaction between different SRI-Master and SRI-Slave peripherals. It supports also pipe lined requests from the SRI-Master interfaces and pipeline address phases to the connected SRI-Slave interfaces.

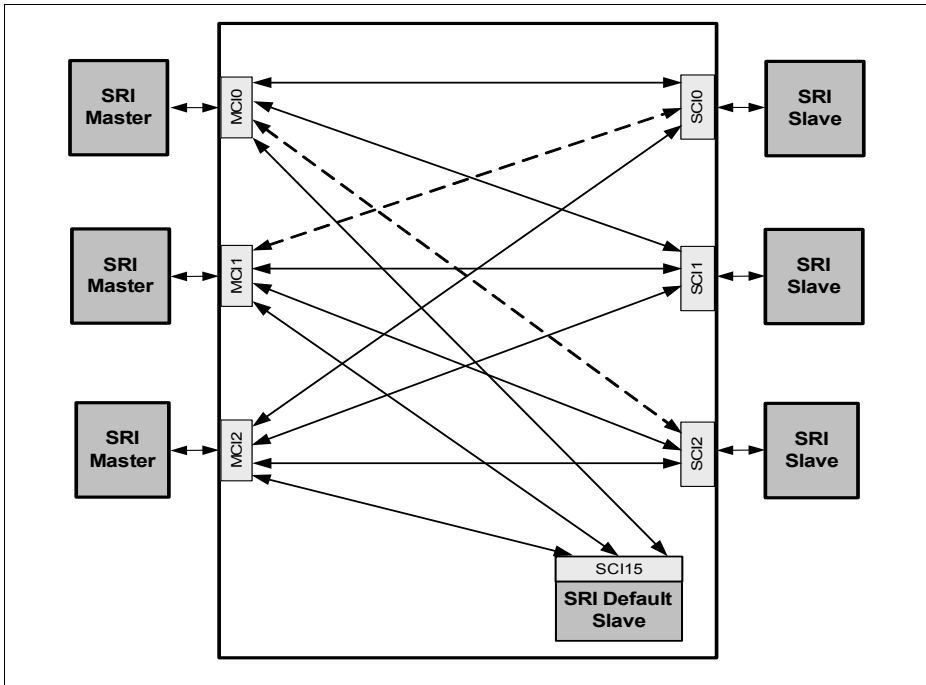


Figure 2-2 XBAR_SRI point to point connection scheme

The XBar_SRI provides SRI Slave Interfaces (SCIx) to connect SRI Slave modules and SRI Master Interfaces (MCix) to connect SRI Master models to the XBar_SRI. The XBar_SRI includes an Default SRI Slave that provides access to the XBar_SRI control

On-Chip System Buses and Bus Bridges

registers and that takes over all SRI transactions to address outside the connected SRI Slave address ranges. The XBar_SRI includes also one Arbiter module per connected SRI slave module and the infrastructure for the enabled read/write data paths. Each connected SRI slave module as well as the default slave have an related arbiter module within the XBar_SRI.

Please note that only these SRI-Master <-> SRI-Slave connections are implemented that are required for the system functionality (example: connection between PMI Master and PMI Slave is not implemented, see also [Table 2-7](#)).

For performance optimization the XBar_SRI includes arbitration schemes that allows to configure SRI master priorities for each SRI slave individually (arbiter functionality). For debug support on system level the XBar_SRI includes debug support for SRI-Error and SRI-Transaction ID errors (local SRI slave module support in the related arbiter, global control in the Default Slave) .

Table 2-1 SRI Bus Terms

Term	Description
Agent	An SRI agent is any master or slave device which is connected to the SRI Bus.
Master	An SRI master device is an SRI agent which is able to initiate transactions on the SRI.
Slave	An SRI slave device is an SRI agent which is not able to initiate transactions on the SRI. It is only able to handle operations that are dedicated to it by SRI CrossBar (XBar_SRI).
XBar_SRI	The SRI CrossBar (XBar_SRI) provides the interconnects between connected Master and Slaves. The XBar_SRI includes arbitration mechanisms and debug capabilities. The XBar_SRI has 16 Master Connection Interfaces (MCI0 - MCI15) to connect SRI master devices to it and 15 Slave Connection Interfaces (SCI0 - SCI14) to connect SRI slave devices to it.
MCI	Each Master is connected via one Master Connection Interface (MCI x, x = 0 ... 15). The XBar_SRI control registers include control and debug informations related to the Master Connection Interfaces MCI x (x = 0 ... 15).
SCI	Each Slave is connected via one Slave Connection Interface (SCI x, x = 0 ... 14). The XBar_SRI control registers include control and debug informations related to the Slave Connection Interfaces SCI x (x = 0 ... 14). The XBar_SRI default slave is connected to SCI 15.

2.2.1.1 XBar_SRI Features

XBar_SRI feature overview:

- Single/Block Data Read Transaction Support (8/16/32/64 Bit)
- Single/Block Data Write Transactions Support (8/16/32/64 Bit)
- Read Modify Write Support
- Supports pipelined requests from SRI master peripherals
- Supports pipelined address phases to SRI slave peripherals
- Single arbiter module for each connected Slave device
- Arbitration priority scheme can be configured for each Slave device individually
- Flexible arbitration schemes (priority, two round robin groups, starvation prevention)
- Breakpoint signal generation based on SRI transactions (OCDS Level 1)
- Configurable MCDS trace interface
- Information integrity support covering SRI address phase signals and the transmitted read/write data
- SRI Address Phase includes Supervisor Mode information (covered by SRI information integrity support)

2.2.2 SRI Transactions

Each SRI transaction consists of:

- one request phase
- one or multiple data phases if not finished with error acknowledge

An SRI master that is requesting for access to an SRI slave is providing all necessary informations about the transaction in parallel with the request. This means that there are no separate on chip bus request and on chip bus address phases. An SRI request/address phase consists of:

- request signal
- lock signal (indicating a read modify write transaction)
- 32-bit address
- 4-bit SRI Op-Code (kind of single data or burst transaction)
- read/write signal (read, write or read modify transaction)
- supervisor mode signal
- Transaction ID (consists of a unique 6-bit Master TAG ID and a 2 bit running number increased for each new transaction of this master)
- 8-bit address phase ECC¹⁾ (Error Correction Code covering all address phase signals with the exception of request and grant)
- one or multiple data phases if not finished with error acknowledge

1) In the current implementation the Error Correction Code is only used for error detection. Detected errors are reported to the SMU but not corrected.

On-Chip System Buses and Bus Bridges

After the request for a slave access was granted by the related XBar_SRI arbiter module, the transaction can be either finished with error acknowledge or with the read/write data phases as defined during the request/address phase.

Each data phase ends with the transmission of data phase informations including:

- Read or Write Transaction ID (must be equal to the related Transaction ID otherwise the address phase is invalid)
- 8-bit read/write data ECC¹⁾ (Error Correction Code covering the 64-bit read/write data and the bits [23:3] of the related address.
- 64 bit read or write data. In case of an 8-bit, 16-bit or 32-bit read/write transaction the unused bits are filled with 'don't care' data. The read/write data ECC¹⁾ covers the 64 data bits as transferred (including the possible don't care data).

2.2.3 SRI Op-Codes

The SRI Op-Code defines for a SRI transaction the number of data phases, the addressing mode in case of a multi beat transaction and valid bytes in case of a single data transaction.

Table 2-2 Operation Code Encoding

sri_opc[3:0]	Identifier	Description
0000	SDTB	Single Data Transfer Byte (8 bit)
0001	SDTH	Single Data Transfer Half-Word (16 bit)
0010	SDTW	Single Data Transfer Word (32 bit)
0011	SDTD	Single Data Transfer Double-Word (64 bit)
0100	-	Reserved
0101	-	Reserved
0110	-	Reserved
0111	-	Reserved
1000	BTR2	Block Transfer Request (2 transfers) Wrap Around Address Mode is used.
1001	BTR4	Block Transfer Request (4 transfers) Wrap Around Mode is used.
1010	BTRL2 ¹⁾	Block Transfer Request (2 transfers) Linear Address Mode is used.
1011	BTRL4 ¹⁾	Block Transfer Request (4 transfers) Linear Address Mode is used.
1100	-	Reserved

Table 2-2 Operation Code Encoding

sri_opc[3:0]	Identifier	Description
1101	-	Reserved
1110	-	Reserved
1111	-	Reserved

1) The SRI implementation in the TC26x B-Step does not support bursts with linear addressing modes.

2.2.4 SRI Error Conditions

The `sri_err_n` signal is used by the slave during a transaction to signal the corresponding master that an error has happened which results in an immediate termination of the current transaction. Errors during transaction are tracked by the `XBar_SRI` and are signalled to via an interrupt and directly to the SMU.

Only the following error conditions are supported and recognized by SRI slaves:

- access level is incorrect (user/supervisor)
- unmapped address access from an SRI master¹⁾
- unsupported op-code
- reserved op-code

An SRI error can be generated by the application SW e.g. with an access to a reserved address (see chapter Memory Map):

2.2.5 SRI Transaction ID Error Conditions

Transaction ID is an identifier connected to all phases of a transaction in order to make the transaction unique in the SRI system during the transactions live time.

The transaction ID is used by SRI masters and slaves to identify problems in the SRI system that results in data packets received by a master or slave that do not match the corresponding arbitration/address phase. If the read/write transaction identifier doesn't match with the previous send transaction identifier in the address phase, this is identified as a transaction ID error and tracked by the `XBar_SRI`.

In situations where at the data source side (master for write transactions, slave for read transactions) a data phase has to be invalidated (e.g. detection of an not correctable SRAM ECC error) and invalid transaction ID is send in order to invalidate the data phase.

An SRI Transaction ID error condition can be generated by injecting an non correctable error into one of the SRAMs (e.g. CPU Instruction Scratch Pad SRAM) and than reading the corrupted data by a CPU.

1) The accesses to unmapped slaves is checked by a default slave.

2.2.6 Operational Overview

This chapter describes the functionality of the XBar_SRI module in order to enable the user to configure the XBar_SRI control registers and to use the XBar_SRI debug resources.

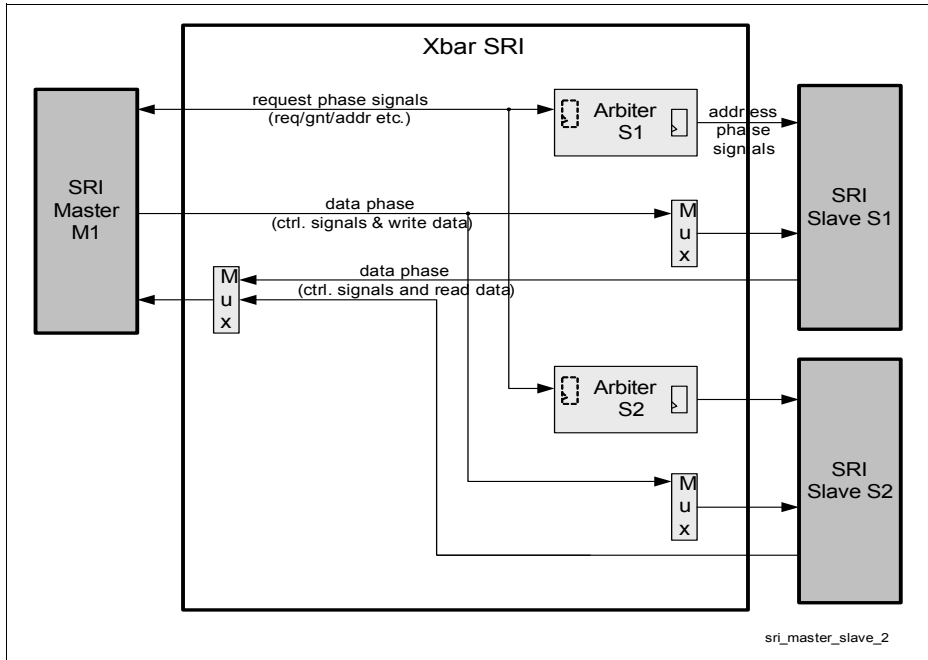


Figure 2-3 XBAR_SRI connections between SRI Master and SRI Slaves

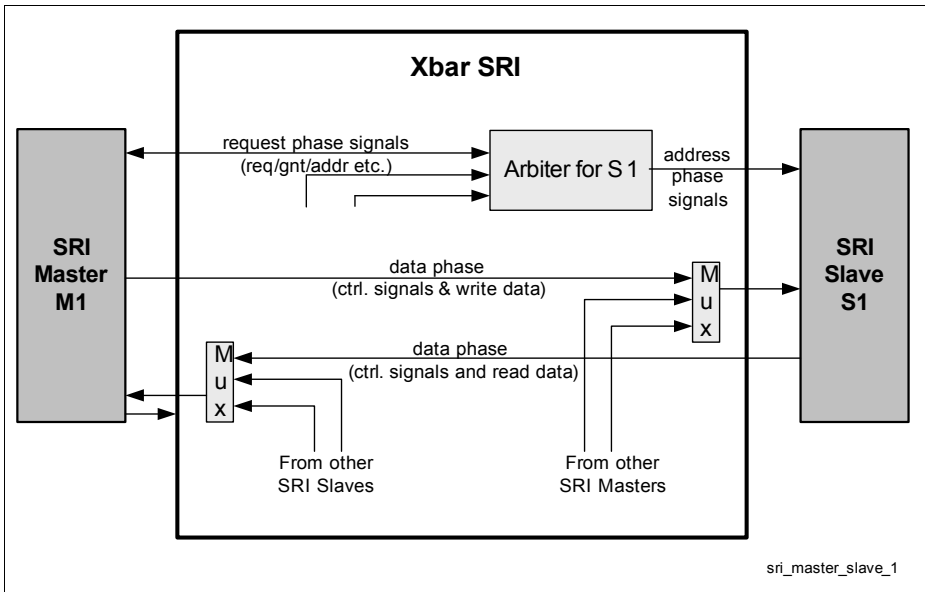


Figure 2-4 XBAR_SRI connections between SRI Slave and SRI Master modules

2.2.6.1 Functional Blocks

The XBar_SRI represents the highest level of the hierarchical SRI-Bus system. Since, it is closest to the TC1.6.xcore, peripherals critical to CPU performance can be attached to it.

The XBar_SRI module is partitioned into blocks for arbitration and data path control which are necessary for each XBar_SRI master- or slave interface and one block that covers the default slave - and debug functionality.

The XBar_SRI module can handle and process several transaction of different master in parallel if the masters requests different slaves.

XBar_SRI Master Connection Interface (MCI)

Each SRI master module in the system is mapped to one or more XBar_SRI Master Connection Interfaces (MCI). Each MCI is related within the XBAR_SRI module to a default arbitration priority and to register control bits and register control bit fields. So each SRI master in the system is mapped to an XBar_SRI internal default arbitration priority and to master related control register bits and bit fields via its MCI number (see also see also [Table 2-4](#) and [Table 2-6](#)).

On-Chip System Buses and Bus Bridges

XBar_SRI Slave Connection Interface (SCI)

Each SRI slave module in the system is mapped to one XBar_SRI Slave Connection Interfaces (SCIx). Each SCI is related within the XBAR_SRI module to one arbiter module with its arbiter module control register and to register control bits and register control bit fields related to this SCIx. So each SRI slave module in the system is mapped to an XBar_SRI internal arbiter module and to slave related control register bits and bit fields via its SCI number (see also see also [Table 2-7](#)).

XBar_SRI Arbiter

Each XBar_SRI slave connection interface is mapped to exactly one dedicated instantiation of the slave arbiter module in the XBar_SRI. This module includes all required functionality for the following tasks:

- Arbitration: the module includes all required functionality for the arbitration. This includes SRI address decoding, SRI request arbitration, SRI request starvation algorithm, SRI address phase generation and control registers for the priority of the connected master connection interfaces and an FSM to detect the end of the current SRI transaction.
- Debugging: the module tracks the SRI transactions to the dedicated slave connection interface for SRI errors. The transaction information of the first transaction where the SRI protocol error is captured in arbiter internal control registers. The module tracks the requests from all masters to detect starvation of masters in the arbitration rounds.
- Error signalling: The first SRI error or starvation error is signalled to the default slave module via two sideband signals.
- XBar_SRI control bus interface: the module has a slave interface to the XBar_SRI control bus. The slave arbiter decodes the address of each new control bus transaction and, if addressed, processes the read/write transaction to the module internal control registers.

Default Slave

The XBar_SRI default slave module is a XBar_SRI internal SRI slave module with its own, dedicated arbiter module inside the XBar_SRI. For accesses to the XBar_SRI control registers only SRI single data transactions of word size are supported. Other SRI op-codes are not supported by the default slave module and acknowledged with an error. The default slave module includes all required functionality for the following tasks:

- As a SRI default slave, it deals with all transactions that are directed to a nonexisting slave in the SRI system as it is described in the SRI protocol. The purpose of the SRI default slave in that situation is to guarantee a defined behavior by terminating these SRI transactions with an error.
- The XBar_SRI default slave module is the SRI interface to all XBar_SRI internal diagnostic- and control registers.

On-Chip System Buses and Bus Bridges

- The XBar_SRI default slave module samples the mci_id_err_n and sci_id_err_n signals from all XBar_SRI master and slave connection interface modules. If the default slave module detects mci_id_err_n and sci_id_err_n pulses it generates an interrupt to the system. The sampling of each mci_id_err_n and sci_id_err_n signal from an arbiter can be disabled via the default slave interrupt control register IDINTEN.
- The XBar_SRI default slave module includes the interrupt node control structure and the corresponding control register.

2.2.7 Functional Overview

2.2.7.1 Arbitration Block

The arbiter has access to all arbitration-/address phase signals from the master connector interfaces he is enabled for, therefore he sees requests from all master connector interfaces in parallel.

In order to check if a master request is addressed to 'its' slave connector interface for the next transaction the arbiter checks if the address transmitted together with the request from the master matches with the allocated address area of this slave via address decoding. Due to the fact that a master can access any slave for a transaction all arbiters of the XBar_SRI are checking the requested address from a requesting master in parallel.

Due to the fact that multiple masters can request for one slave in parallel, each slave has to decode the addresses from all the master connector interfaces it is enabled for in parallel.

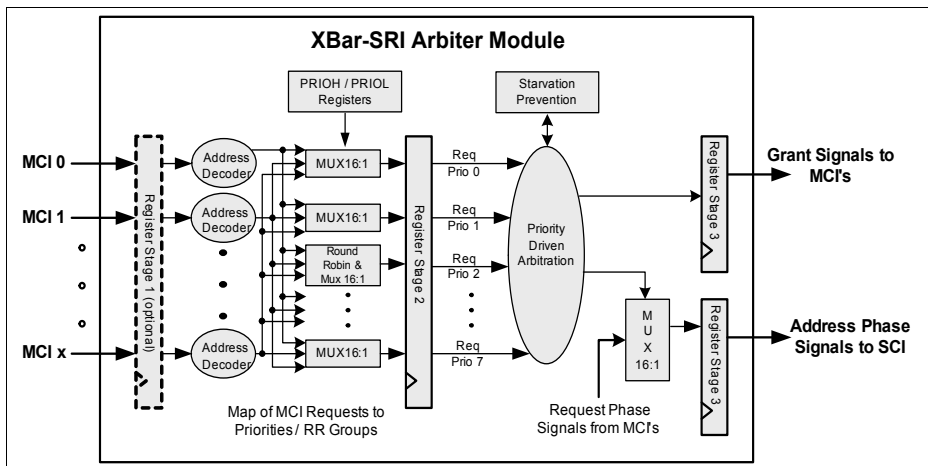


Figure 2-5 XBAR_SRI arbiter scheme

Address Map Checking

The arbiter performs the address comparison for all pending requests from the connected SRI master modules to its slave module in parallel.

On-Chip System Buses and Bus Bridges

Arbitration

A transaction request from a master that matches the address area of a slave connection interface takes part in the next arbitration cycle.

The arbitration is divided into two levels:

- priority driven arbitration
- arbitration within round robin groups (priority 2 and 5)

After reset all enabled master connector interface are mapped to the priorities 2 or 5 (also see also [Table 2-4](#)). The priorities and the priority algorithm of the master connection interfaces can be programmed for each arbiter individually. The programming of the master priorities can be done by SRI read/write transaction via the default slave module.

The highest priority for an arbiter is 0 and the lowest is 7.

According to the transaction rules described in the SRI protocol specification the arbiter asserts the `sri_gnt_x` signal to grant the slave to this winning master.

In parallel - or after granting the master, the arbiter sends the address phase for the next transaction to the slave by propagating all necessary informations via the slave connection interface to the slave. The arbiter sends the address phase in parallel with the grant or delayed depending on the address phase pipeline status of the corresponding SRI slave.

For debug purposes, the arbiter samples all necessary transaction informations and provides them to the `XBar_SRI` default slave module if an SRI error happened and the protocol error feature was enabled.

Arbitration Algorithms

The arbiter related to a slave connection interface (SCI) can be connected to all master connection interfaces (MCI), see also [Chapter 2.2.8.3](#). The priority of each MCI can be controlled via the arbiter priority registers (PRIOx). The priority of a master is defined by 3 bit field in the `PRIOL / PRIOH` register that is related to this master / its MCI number. This can be configured individually for each arbiter so the same master can be handled with different priorities for accesses to different slaves. After reset all enabled MCI are configured with the priority 2 or 5 (Round Robin group).

Please Note:

It must be ensured that two enabled masters don't have the same priority, with the exception of priority 2 and priority 5 (round robin group priorities).

For changing the priorities during runtime (switching the priorities), all master connection interfaces that should be remapped have to be mapped to the round robin group priorities (2 / 5) before mapping them to their new priorities can be done. This will prevent situations where two masters have the same priority but not a round robin one.

On-Chip System Buses and Bus Bridges
Priority Driven Arbitration

The general arbitration algorithm is priority driven where priority 0 is the highest priority and 7 the lowest one. If multiple masters are requesting for one slave, the master with the highest priority will win the next arbitration round (see also 'starvation prevention').

Round Robin Groups

The arbiter arbitrates in general priority driven, where Priority 2 and priority 5 contain a second 'arbitration' layer. Both priorities can be used as round robin priorities for a group of max. 8 MCI's each. If only one master is mapped to a round robin group priority, the master's request will be treated as normal master request with the priority of the round robin group. If more than one master is mapped to the priority of a round robin group, the requests of the mapped masters will be handled by the round robin algorithm, the winning request of the round robin group arbitration has the priority of the round robin group within the priority driven arbitration.

After the winner of a round robin group is granted, the round robin group starts with a new arbitration round which means the requesting MCI's of the round robin group with the next highest MCI ID number will win the next round robin arbitration round. If there is no requesting MCI with a higher ID number in the round robin group, the algorithm will start with the MCI with the lowest MCI ID number that is mapped to the group, going to the next higher MCI ID number and so on.

Request Latency

If no other request is pending, a request from an MCI has a latency of 1 clock cycle, starting with the detection of the SRI master request on the bus, ending with the SRI grant to the requesting MCI and the SRI address phase to the addressed SCI.

Table 2-3 XBar_SRI Request Latency

Clock Cycle Nr.:	Task(s)
0 (Default Slave)	sri_req_n is sampled at XBar_SRI MCI (Only valid for access to the XBar_SRI control registers)
1	address decoding, mapping of MCI's to priorities / round robin groups and starvation prevention. Round robin arbitrations, priority driven arbitration, mux request phase signals to the address phase registers masking sri_req_n from MCIX after granting the MCIX
2	sri_gnt_n to the requesting MCI, SRI address phase to SCI.

Grant -> New Request Latency (Request Dead Time)

After an MCI was granted, it takes 2 clock cycles before a new request from the same MCI will participate in a new arbitration round even when the SRI master has requested

On-Chip System Buses and Bus Bridges

permanently. This request dead time is a result of the synchronous sri_req_n de-assertion and the XBar_SRI request latency.

Default MCI x Priorities after Reset

After Reset all Master Connection interfaces are mapped to the round robin groups (priority 2 and 5, lower number means higher priority). [Table 2-4](#) shows the default priority of the MCI with the related coding in the XBAR_PRIOH and XBAR_PRIOL registers. After reset, the priority scheme of the MCIs can be re-configured via the XBAR_PRIOH and XBAR_PRIOL registers, for each SCI (accesses to the Slave connected to an SCI) individually. See also [Chapter 2.2.8.1](#),

Note: If multiple CPUs are connected to the Aurix_Bus it is proposed to give the CPU master interfaces (DMI and PMI) the same round robin priority, e.g. 5. This ensures a fair arbitration between CPU access conflicts to the same on chip resource, e.g. Flash.

Table 2-4 Default MCI Priority in the XBar_SRI Arbiters

Priority	Coding	Round Robin Group	Default Mapping:	Comments
0	000	-	-	
1	001	-	-	
2	010	Yes, max 8 master	MCI 0-7	Maximal 8 MCI can be mapped to the priority 2 e.g. can have the priority '010'.
3	011	-	-	
4	100	-	-	
5	101	Yes, max. 8 master	MCI 8-15	Maximal 8 MCI can be mapped to the priority 8 e.g. can have the priority '101'.
6	110	-	-	
7	111	-	-	

Starvation Prevention

Starvation can occur when masters with high priority continuously request a dedicated slave, preventing other masters with lower priority from getting access to this slave. To prevent such a situation, a starvation counter has been implemented in each arbiter. This mechanism allows the promotion of weak masters.

On-Chip System Buses and Bus Bridges

Each time the starvation counter in a slave module has an underflow, all pending requests to this arbiter will be entered in the arbiter's request list. This even applies to all masters mapped to a round robin group.

Next time when the starvation counter has an underflow, the masters of this request list will be entered in the arbiter's request promotion list if:

- The request was not granted during the last starvation period

The masters in that request promotion list will be the next to be granted independent of their priorities. The masters in the request promotion list will be granted one after the other, starting with the master which has the lowest MCI number in this list if there are more than one. The master promotion list has the highest priority within the arbiter's main arbitration algorithm, therefore all masters in the promotion list will be granted before the arbiter switches back to its 'normal' arbitration. A master is removed from both lists when it is granted.

If several masters are mapped to a round robin priority, all masters of that round robin arbitration round will be entered in the request list/request promotion list when not granted.

The value controlling the counter period is programmable. After reset the starvation counter has a value of zero. On a starvation counter underflow it is reloaded with the content of the arbiter control register ARBCON.SPC. An underflow occurs in the clock cycle when the counter tries to count down from zero.

The number of arbitration cycles a master must wait for the slave varies. But it's guaranteed to be no more than $2 \times \text{ARBCON.SPC}$ until the master is promoted to the request promotion list.

Each time there is an underflow it is checked if there are any masters in the request promotion list that were not granted since the last underflow. This can happen if there are too many/too long transactions to be promoted compared to ARBCON.SPC value. In this case an error is generated via the `xcb_sc_err_n` signal to the default slave if the feature was enabled (bit ARBCONx.SCERREN set).

If currently a RMW transaction is processed the starvation counter is stopped before the next overflow. The starvation counter is released again after the address phase of the write part is generated or an error for the RMW transaction was received.

If currently a Read-Modify-Write (RMW) transaction is processed by the slave the starvation protection counter is stopped until this RMW is finished.

2.2.7.2 Default Slave

The default slave serves three features in the XBar_SRI implementation:

On-Chip System Buses and Bus Bridges

Control and Configuration Registers Interface

The XBar_SRI default slave module handles all read and write transactions to the debug- and control registers of the XBar_SRI. For this purpose, the default slave module has its own address space which must be mapped into the system address space by an address decoder, provided by the customer.

For a detailed description of the registers see [Chapter 2.2.9](#).

Non Existing Addresses

The XBar_SRI internal default slave module has its own arbiter module.

If an SRI master sends a request with a non existing address¹⁾ to the XBar_SRI, the transaction will be directed to the default slave. The default slave finishes the transaction with error following the SRI protocol rules, which activates the error tracking mechanism of the arbiter. As a result of the error, the default slave module signals this incorrect behavior to the system by generating an interrupt.

A write from an SRI master to a non existing address on the System Peripheral Bus (SPB) will be handled by the Bus Control Unit on the SPB only. A read from an SRI master to a non existing address on the SPB will be handled by the by the Bus Control Unit on the SPB and also captured by the XBar_SRI arbiter as it will see the transaction as read transaction finished with Error Acknowledge.

Error Handling

If an arbiter detects an SRI protocol error during a transaction with a corresponding slave the involved arbiter samples all relevant information of the transaction in the arbiter internal diagnostic registers (ERRADDRx and ERRx) and signals this event via `xcb_sri_err_n` or `xcb_sri_err_d_n` sideband signal to the default slave module.

Note: The two error registers ERRx and ERRADDRx in each arbiter are updated with the content of the currently processed transaction in the data phase. The registers are only updated if they are not locked due to a pending protocol error and `sri_ready_n` was asserted in parallel with `sri_err_n` deasserted.

The error capture registers in the arbiter can't be changed until the interrupt was acknowledged to the slave arbiter module (set the ARBCON.INTACK) via the default slave.

The stored informations can be read from the default slave module with SRI single data read transactions, the acknowledge can be sent to the default slave via a write to a specific address.

1) non existing address = all Reserved address ranges described in the Memory Map chapter

On-Chip System Buses and Bus Bridges

This can result in a loss of interrupt data information of the same source beyond the first until the write was processed but as all SRI errors must be analyzed and fixed before the product will work, these kind of errors can be analyzed one after the other.

In case that a master or a slave signals an ID error to the XBar_SRI the default slave marks the source of the ID error by setting the according bit in register IDINTSAT if this feature was enabled for that specific master and/or slave interface.

As a result, the default slave generates an interrupt to the system.

Note: While a status bit for a error source is set in either the INTSAT or IDINTSAT register a new error from this particular source doesn't generate a new interrupt.

1. *Each participant in the SRI-Bus system has three interrupt sources; protocol errors, ID errors and time-out due to starvation. All error from the SRI-Bus components are combined in the single SRN of the XBar_SRI.*

Each arbiter has two 32 bit registers containing the error information. This registers are accessed as all other registers, via the default slave module.

The default slave module has one service request node (SRN) to start interrupts for detected SRI errors. Protocol errors, starvation errors as well as ID errors are handled together by this node.

2.2.7.3 Register Access Protection

The XBar_SRI registers are protected by a master TAG ID based access protection mechanism. Each on chip resource with direct or indirect bus master capability has a unique master TAG ID that can be use to identify the master of an on chip bus transaction.

Access Enable:

TAG ID based protection means that on chip bus write access to the XBar_SRI control registers can be disabled for each master TAG ID individually (with the exception of the Access Enable registers itself, which is Safety Endinit protected). For a disabled master TAG ID, write access will be disconnected with error acknowledge, read access will be processed.

Access Enable Registers (XBAR_ACCEN1/0):

-> defines which master TAG ID is allowed to write to XBar_SRI control registers

-> are Safe Endinit write protected

The Access Enable registers (XBAR_ACCEN1/0) are Safe Endinit protected. The Safe Endinit system status is defined by a Watchdog Unit in the System Control Unit (SCU)

On-Chip System Buses and Bus Bridges

After reset, all XBAR_ACCEN1/0 access enable bits and access control bits are enabled, access protection mechanism has to be configured and checked to bring the system in a safe state.

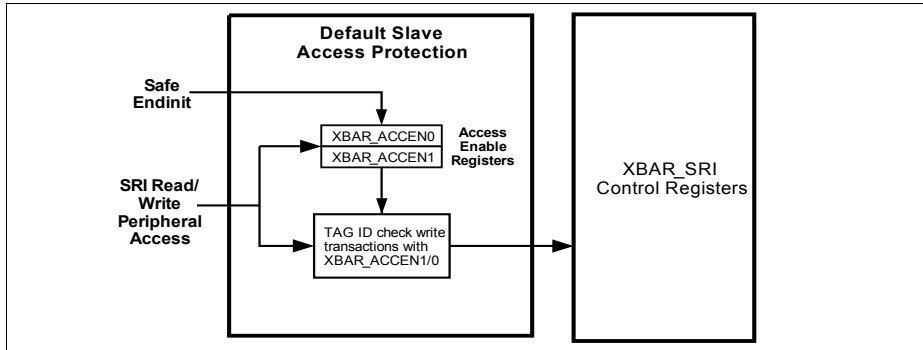


Figure 2-6 XBar_SRI: Control Register Access Protection scheme

2.2.7.4 SRI ECC Error Handling

The SRI protocol provides information integrity support covering:

- The address phase of an SRI transaction
- The transmitted read and write data

The integrity support mechanism is using Error Correction Code (ECC).

Note: In the current implementation the Error Correction Code (ECC) is only used for error detection. Detected errors are reported to the SMU but not corrected.

Detected SRI address phase ECC errors:

- If an SRI slave detects an SRI address phase ECC error it finishes the transaction with SRI error acknowledge and does not further process it (see also [Chapter 2.2.7.5](#)). Error is signalled to the SMU.

Detected SRI read data ECC errors:

- PMI and DMI: a bus error trap will be generated but only if and when a received 64 bit read data is really used by the CPU. This as PMI / DMI can read complete cache lines (4x64 bit) where the critical 64 bit is used, the other 3 x 64 bit are speculative reads and might not be used by the CPU later.
- DMA: the DMA ignores read data ECC errors but signals the error to the SMU.
- SFI: the SFI SRI master ignores read data ECC errors but signals the error to the SMU

Detected SRI write data ECC errors:

- An detected write data ECC error is signalled to the SMU

On-Chip System Buses and Bus Bridges

- SFI: the SFI slave ignores the write data ECC errors but signals the error to the SMU
- SRI write data ECC errors to SRAMs will not be executed by the related slave modules.

It can happen that a data during a transaction shows an ECC error inside the data source peripheral (master during write, slave during read). Example: A burst read transaction from a memory slave peripheral where the second, third or fourth data taken from the SRAM shows inside the slave/master an ECC error. In this situation a slave can finish a read transaction with error acknowledge (if the error is detected before the first data phase) or a slave / master can invalidate the related data phase with an invalid read / write transaction ID error (see also [Chapter 2.2.7.5](#)).

ECC Code

The ECC codes used for the IR Error Detection mechanism is a Hsiao 22_5 code with DED (double error detection) capability:

- SRI Address Phase informations are protected by the 64_8 code
- SRI Read / Write data are protected by the 96_8 code

```

        CONSTANT      code_matrix      :      matrix_vec96_8      :=(
"0110011011010111111001011011110100010111000001000010001001011100
100010001010110000001011101010111",
"000110110110101011101010110111100111011100001000010001001010101
000100010001010100010101110101011",
"101101011000110101010011011011110011111100010000100010010011010
000000100101101110100010000111101",
"000110011111000110111100011110111100111100100001000100011100000
101001010110001011000100011001110",
"110111100011111000111111000011110101001010000100001111100000001
010001101000000011111000111110000",
"1110000000111111101111111100011100101010000011111000000000010
11111000000000110111111000000000",
"1110111111000000000111111111101111000011111100000000000000011
01111111111110000000000000000000",
"111111111111111111000000000000111110011111111111111111111111100
000000000000000000000000000000");

```

```

        CONSTANT      code_matrix      :      matrix_vec64_8      := (
"000101110000010000100010010110010001000101011000000101110101011
1",
"011101110000100001000100101010100010001000100101010001010111010101
1",
"001111110001000010001001001101000000010010110111010001000011110

```

```

1" ,
"110011110010000100010001110000010100101011000101100010001100111
0" ,
"10101001010000100001111000000101000110100000001111100011111000
0" ,
"11001010100000111110000000000101111100000000110111111100000000
0" ,
"111100001111110000000000000001101111111111100000000000000000
0" ,
"111111001111111111111111111100000000000000000000000000000000
0" );

```

2.2.7.5 Error Tracking Capability

The XBar_SRI tracks all SRI transactions for SRI protocol errors. Additionally it tracks informations about starvation errors and SRI transaction ID errors. This is done by all arbiters and the default slave in parallel as the XBar_SRI supports the processing of multiple transactions from master and slaves in parallel which can result in parallel events at different slave connector interfaces.

For this purpose, each arbiter has two error/debug registers where it samples the transaction informations of the transaction where the first protocol error happened.

Note: Protocol errors and debug trigger events can lock the error/debug registers. Only the first event of both sources can lock the registers. All other events are not captured with this two error/debug registers unless the lock was released in the meantime.

Further protocol errors will be ignored by an arbiter that has detected an protocol error until the tracking mechanism is re-activated via the arbiter internal control register. A detected protocol error or starvation error is signalled by the arbiter to the default slave module via two separate sideband signals for SRI- and starvation error. The default slave samples the sideband signals pulses in an error status register INTSAT. As each slave has its own sideband signals, the default slave has the information which arbiter has detected an SRI protocol or starvation error. Each error signal can be masked individually by control registers in the arbiter modules. All debug registers are accessible via the SRI-Bus interface of the default slave.

For transaction ID errors of write transactions the SCI propagates the sci_id_err_n signal via the sri_wr_tr_id_err signal from the slave to the default slave module. The default slave sets the assigned bit in register IDINTSAT.

For transaction ID error of read transactions the MCIs propagate the sri_id_err_n signal via the sri_rd_tr_id_err signal from the master to the default slave module. The default slave sets the assigned bit in register IDINTSAT.

On-Chip System Buses and Bus Bridges

Once an error was signalled from an arbiter or an MCI/SCI to the default slave, the default slave module sends an interrupt request to the system. The system can read out the error status registers in the default slave module to find out which arbiter(s) or master/slave have detected an error, then the system can start with more detailed diagnostics by reading out the error/debug registers in the arbiter for a protocol error. The error informations captured for an SRI protocol error allows the identification of the master via the sampled master tag ID and the final destination via the sampled target address. Additionally the arbiter samples the op-code and the sri_rd_n, sri_wr_n and sri_svm control signals of the transaction.

In addition to the SRI transaction information, the XBar_SRI captures sideband signal informations (**XBAR_ERRx (x = 0-1)**.MCI_SBS and **XBAR_ERRD**.MCI_SBS). In the TC26x B-Step these sideband signals are used by the DMA SRI master interface to provide informations about the requestor of a transaction, in parallel to the SRI request phase. **Table 2-5** shows the encoding of the MCI_SBS bit field for the encoding of the TC26x B-Step.

Table 2-5 Encoding of ERRx.MCI_SBS in the TC26x B-Step

MCI_SBS[7:0]	Bit field encoding
MCI_SBS[7:0]	<p>This bit field is only valid if the master TAG ID of the address phases is related to one of the DMA hardware resource groups.</p> <p>MCI_SBS[7:0] is showing the number of the DMA channel that initiated the transaction.</p>

After reading all relevant error informations, the error/debug tracking mechanism can be reactivated.

2.2.7.6 Debug Trigger Event Generation (OCDS Level 1)

This functionality can be used to generate breakpoints on selectable and configurable events in the SRI-Bus traffic.

The debug trigger event generation is controlled via the three debug registers DBCON, DBADD and DBMADD and the three status registers DBSAT, ERR and ERRADDR.

All DBXXX registers reset only with the debug reset (Class 1 reset).

The register DBSAT collects all debug trigger events from all arbiters in the XBar_SRI module. When an arbiter generates a debug trigger event the according bit in register DBSAT is set.

The DBCON register defines the debug trigger event conditions for each arbiter individually. Several individual break conditions can be combined by enabling them in parallel. Possible break conditions are:

- Write transactions

On-Chip System Buses and Bus Bridges

- If bit `DBCON.WREN` is set only transactions with an asserted `sri_wr_n` signal can generate a debug event.

Note: Only if `DBCON.WREN` and `DBCON.RDEN` are set together a RMW transaction generate a debug trigger event.

- Read transactions
 - If bit `DBCON.RDEN` is set only transactions with an asserted `sri_rd_n` signal can generate a debug event.

Note: Only if `DBCON.WREN` and `DBCON.RDEN` are set together a RMW transaction generate a debug trigger event.

- Supervisor mode transactions
 - If bit `DBCON.SVMEN` is set only transactions with an asserted `sri_svm` signal can generate a debug event.
- Transactions from a dedicated master
 - If bit `DBCON.MASEN` is set only transactions initiated by master `DBCON.MASTER` can generate a debug event.
- Transactions accessing a defined address area
 - If bit `DBCON.ADDEN` is set only transactions accessing an address in the selected address area can generate a debug event. The selected address area is defined by the registers `DBADD` and `DBMADD`. Register `DBADD` defines one global 32-bit address that is compared with `sri_addr[31:0]` for all bits where `DBMADDR` is set to '1'.

All enabled debug trigger conditions are combined by a logical AND.

Example: If `DBCON.WREN` and `DBCON.SVMEN` are set and all other enables are cleared a debug trigger event is only generated for a write transaction operating in the supervisor mode.

Additionally a debug trigger event is generated if `DBCON.ERREN` is set and an error occurs. Please note that an error occurs only when the generation for this source is enabled in the linked registers `ARBCON` or `IDINTEN`.

The result of the logical AND of the first five debug trigger event options is combined with the result of the error debug trigger event by a logical OR.

A debug event is signaled to the default slave. The default slave combines all `XBar_SRI` arbiter debug event signals with its own and generates the debug event signal that is sent to the OTGM module.

The debug event signal to the OTGM will be asserted for as long at least one condition inside an `XBar` module or the `XBar` default slave module is met.

Debug Trigger events inside the `XBar_SRI` are sampled in the register `DBSAT`. Additionally an interrupt can be generated for debug trigger events by the `XBar_SRI`.

When debug condition is reached in one of the `XBar_SRI` arbiter modules, informations of the transaction that matched to the debug condition is captured in the two error/debug

On-Chip System Buses and Bus Bridges

capture registers ERRx and ERRADDRx. If the two registers are already locked due to an earlier action the capturing is not performed.

Writing to DBCON.REARM will rearm the feature, this also sets DBCON.ARM.

2.2.7.7 Interrupt and Debug Events of the XBar_SRI Module

There are some interactions between interrupt them self and debug events. In general due to the nature of the crossbar concept several interrupts from the same or a different source (arbiter, MCI or SCI) can occur. One interrupt could occur several times before a service request routine is initiate. Additionally can all interrupts occur in parallel to one or more debug trigger events.

All following examples assume a time interval without any acknowledge either from a service routine or a debug routine and all consecutive interrupts/events come from the same source.

Two Consecutive Protocol Errors

The first protocol error is captured as normal together with a generation of an interrupt to the system. The second protocol error is not captured as the two registers ERR and ERRADD are already locked and no interrupt is generated.

Two Consecutive Starvation Errors

The first starvation error generate an interrupt to the system. The second starvation error will not generate an interrupt to the system.

Two Consecutive Transaction ID Errors

The first transaction ID error generate an interrupt to the system. The second transaction ID error will not generate an interrupt to the system.

Two Consecutive Debug Trigger Events

The first debug trigger event is captured as normal together with a generation of debug trigger event signal to the system. The second debug trigger event is not captured as the two registers ERR and ERRADD are already locked and a debug trigger event signal is generated.

Protocol Error followed by a Debug Trigger Event

The first protocol error is captured as normal together with a generation of an interrupt to the system. The later debug trigger event is not captured as the two registers ERR and ERRADD are already locked and a debug trigger event signal is generated to the system.

On-Chip System Buses and Bus Bridges

Debug Trigger Event followed by a Protocol Error

The first debug trigger event is captured as normal together with a generation of debug trigger event signal to the system. The later protocol error is not captured as the two registers ERR and ERRADD are already locked and no interrupt is generated.

Starvation/Transaction ID Error followed by a Debug Trigger Event

The first starvation/transaction ID error generate an interrupt to the system. The later debug trigger event is captured to the two registers ERR and ERRADD and a debug trigger event signal is generated to the system.

Debug Trigger Event followed by a Starvation/Transaction ID Error

The first debug trigger event is captured as normal together with a generation of debug trigger event signal to the system. The later starvation/transaction ID error generates an interrupt to the system.

Releasing the Lock from registers ERR and ERRADD

If the ERR and ERRADD registers are locked only from one even only (protocol error or debug trigger event) the lock can be releasing by:

- Writing a one to ARBCONx.INTACK when the registers are locked by a protocol error
- Writing a one to DBCONx.REARM when the registers are locked by a debug trigger event

If both, a protocol error and a debug trigger event occurred since the lock was released the last time both locks have to be released

- Writing a one to DBCONx.REARM AND Writing and to ARBCONx.INTACK when the registers are locked by a debug trigger event AND a protocol error

2.2.8 Implementation of the Cross Bar (XBar_SRI) in the TC26x B-Step

This chapter describes the SRI Interconnect implementation in the TC26x B-Step. The knowledge of the specific implementation (e.g. the connection of the SRI master / slave devices to the Interconnect) is necessary in order to:

- map error informations to the connected slave devices
- define the arbitration scheme for accesses to the connected slave devices
- map XBar_SRI (arbiter) control registers to connected slave devices

This chapter includes three tables that are describing: the relationship (mapping) of

- the relationship (mapping) of TC26x B-Step SRI master devices to the XBar_SRI Master Connection Interfaces MCI 0 - MCI 15 ([Table 2-6](#))
- the relationship (mapping) of TC26x B-Step SRI slave devices to the XBar_SRI Slave Connection Interfaces SCI 0 - SCI 15 ([Table 2-7](#))

On-Chip System Buses and Bus Bridges

- the point to point connections between TC26x B-Step SRI master and slave devices ([Table 2-7](#))

2.2.8.1 Mapping of SRI Master Modules to XBar_SRI Master Interfaces

[Table 2-6](#) shows the mapping of master devices to the XBar_SRI Master Interfaces (MCI 0 - MCI 15). Most of the XBar_SRI control registers are related to the XBar_SRI Slave Interfaces (SCI 0 - SCI 15) or the XBar_SRI Master Interfaces. Therefore it is important to know which TC26x B-Step SRI master device relates to which XBar_SRI Slave Interface.

Example 1:

The XBar_SRI includes error registers where each MCI is represented with 1 bit, showing if during the transfers requested by the master devices connected to the MCI's an error situation occurred (e.g. [XBAR_IDINTSAT](#)).

Example 2:

The XBar_SRI includes one arbiter module per connected SRI slave device. Each arbiter module includes a four bit field where the priority requests from connected MCI can be defined. If the access priority of the DMI SRI master to one SRI slave device has to be changed, this can be done via the bit field related to MCI 4 in the arbiter control register related to this SRI slave (control registers: XBAR_PRIOHx, XBAR_PRIOLx, XBAR_PRIODx).

Table 2-6 Mapping of TC26x B-Step SRI master devices to MCI

XBar_SRI Master Connection Interface	MCI Number	Priority after Reset	Connected SRI master device
MCI 0	0	2	DMA
MCI 1	1	-	-
MCI 2	2	-	-
MCI 3	3	-	-
MCI 4	4	2	HSSL
MCI 5	5	2	SFI (ETH access to SRI)
MCI 6	6	2	-
MCI 7	7	-	-
MCI 8	8	5	CPU1.DMI
MCI 9	9	5	CPU1.PMI
MCI 10	10	-	-
MCI 11	11	-	-

On-Chip System Buses and Bus Bridges
Table 2-6 Mapping of TC26x B-Step SRI master devices to MCI

XBar_SRI Master Connection Interface	MCI Number	Priority after Reset	Connected SRI master device
MCI 12	12	5	CPU0.DMI
MCI 13	13	5	CPU0.PMI
MCI 14- MCI 15	14-15	-	-

Note: If multiple CPUs are connected to the Aurix_Bus it is proposed to give the CPU master interfaces (DMI and PMI) the same round robin priority, e.g. 5. This ensures a fair arbitration between CPU access conflicts to the same on chip resource, e.g. Flash.

2.2.8.2 Mapping of SRI Slave modules to XBar_SRI Slave Interfaces

Table 2-7 shows the mapping of slave modules to the XBar_SRI Slave Interfaces (SCI 0 - SCI 15). Most of the XBar_SRI control registers are related to the XBar_SRI Slave Interfaces (SCI 0 - SCI 15) or the XBar_SRI Master Interfaces. Therefore it is important to know which TC26x B-Step SRI slave device relates to which XBar_SRI Slave Interface or arbiter module.

Example 1:

The XBar_SRI includes error registers where each SCI is represented with 1 bit, showing if during the transfers requested by the master devices connected to the MCI's an error situation occurred (e.g. **XBAR_DBSAT**, **XBAR_IDINTSAT**, **XBAR_IDINTEN**).

Example 2:

The XBar_SRI includes one arbiter module per connected SRI slave device. Each arbiter module includes error capturing resources and breakpoint capabilities. These can be used e.g. to analyze accesses to the connected slave device that were answered with error acknowledge by the slave device (e.g. **XBAR_ERRx (x = 0-1)**, **XBAR_ERRADDRD**).

Table 2-7 Mapping of TC26x B-Step SRI slave devices to SCI

XBar_SRI Slave Connection Interface (SCI)	Connected SRI master device
SCI 0	CPU0 (PSPR, DSPR, SFR, CSFR, PCache RAM, PTAG)
SCI 1	CPU1 (PSPR, DSPR, SFR, CSFR, PCache RAM, PTAG)
SCI 2 - 3	-
SCI 4	LMU (EMEM)

On-Chip System Buses and Bus Bridges

Table 2-7 Mapping of TC26x B-Step SRI slave devices to SCI

XBar_SRI Slave Connection Interface (SCI)	Connected SRI master device
SCI 5	-
SCI 6	PMU0: DFlash, BootROM, CTRL Reg
SCI 7	PMU0: PFlash0
SCI 8	PMU0: PFlash1
SCI 9 - 14	-
SCI 15	XBar_SRI Default Slave

2.2.8.3 TC26x B-Step SRI Master / Slave Interconnection Matrix

Table 2-7 shows the SRI master to SRI Slave interconnects that are implemented in the TC26x B-Step (‘x’ in a table cell means connection implemented. ‘-’ in a table cell means connection not implemented).

		DMA	HSSL	SFI	GPU1: DMI	GPU1: PMI	GPU0: DMI	GPU0: PMI
		MSC0	MSC4	MSC5	MSC8	MSC9	MSC12	MSC13
CPU0	SSC0							
CPU1	SSC1							
LMU	SSC4							
PMU0: Dflash /BROM	SSC6							
PMU0: Pflash0	SSC7							
PMU0: Pflash1	SSC8							

TC26x_xbar_connect

Figure 2-7 TC26x B-Step: SRI Master / Slave Interconnection Matrix

2.2.8.4 Connection Master-Slave in XBar_SRI

On-Chip System Buses and Bus Bridges

As the SRI-Bus protocol is a point to point based bus implementation multiplexer inside the data paths in front of the MCIs and arbiter/SCIs are required (see [Figure 2-3](#) and [Figure 2-4](#)).

The write data path multiplexer in front of the SCIs are controlled by the related arbiter modules. The read data path multiplexers in front of the MCIs are controlled by all arbiters.

Master - and Slave side MUX

During an SRI transaction, the corresponding arbiter has to establish the data path connection from his slave connection interface to the corresponding master connection interface in order to enable the master to receive the SCI control signals , if it is a read transaction the read data, send by the slave.

On-Chip System Buses and Bus Bridges

2.2.9 SRI Crossbar Registers

Figure 2-8 and Table 2-8 are showing the address maps with all registers of the SRI Crossbar (XBar_SRI) module.

List of used Reset Class abbreviations:

- Reset Class 1 -> Debug Reset (see description in the chapter SCU / Reset Types)
- Reset Class 3 -> Application Reset (see description in the chapter SCU / Reset Types)

XBar_SRI Unit Register Overview

Identification Register	Default Slave Register	Arbiter Register SCIx	Arbiter Register Default Slave
XBAR_ID	XBAR_DBSAT	XBAR_EXTCONy	XBAR_EXTCOND
	XBAR_INTSAT	XBAR_ARBCONx	XBAR_ARBCOND
	XBAR_IDINTSAT	XBAR_PRIOHx	XBAR_PRIOHD
	XBAR_IDINTEN	XBAR_PRIOLx	XBAR_PRIOLD
		XBAR_ERRADDRx	XBAR_ERRADDRD
		XBAR_ERRx	XBAR_ERRD
		XBAR_DBCONx	XBAR_DBCOND
		XBAR_DBADDx	XBAR_DBADD
		XBAR_DBMADDx	XBAR_DBMADD

XBar_reg_its

Figure 2-8 TC26x B-Step Control Registers

List of used Reset Class abbreviations:

- Reset Class 1 -> Debug Reset (see description in the chapter SCU / Reset Types)
- Reset Class 3 -> Application Reset (see description in the chapter SCU / Reset Types)

Note: Addresses listed in column “Offset Address” of Table 2-8 are word (32-bit) addresses.

Note: XBar_SRI registers can be accessed only with SDTW (32 bit) transactions. 8, 16 bit and RMW transactions are not supported.

On-Chip System Buses and Bus Bridges
Table 2-8 Registers Address Space - XBar_SRI Register Address Space

Module	Base Address	End Address	Note
XBAR	F870 0000 _H	F870 04FF _H	

Table 2-9 Registers Overview - Aurix_Bus Module Control Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
-	Reserved	400 _H - 404 _H	BE	BE	-	-
ID	Identification Register ²⁾	408 _H	U, SV	BE	-	Page 2-36
DBSAT	Debug Trigger Event Status Register ²⁾	40C _H	U, SV	SV, P	1	Page 2-37
INTSAT	Arbiter Interrupt Status Register ²⁾	410 _H	U, SV	SV, P	3	Page 2-38
IDINTSAT	ID Interrupt Status Register ²⁾	414 _H	U, SV	SV, P	3	Page 2-41
IDINTEN	ID Interrupt Enable Register ²⁾	418 _H	U, SV	SV, P	3	Page 2-44
-	Reserved	41C _H - 4F4 _H	BE	BE	-	-
ACCEN1	Access Enable Register 1	4F8 _H	U, SV	SV, SE	3	Page 2-70
ACCEN0	Access Enable Register 0	4FC _H	U, SV	SV, SE	3	Page 2-69
EXTCOND	External Slave Control (SFI control registers)	000 _H	U, SV	SV, P	3	Page 2-45
ARBCOND	Arbiter Control Register Default Slave	004 _H	U, SV	SV, P	3	Page 2-47
PRIOHD	Arbiter Priority Register High Default Slave	008 _H	U, SV	SV, P	3	Page 2-49
PRIOLD	Arbiter Priority Register Low Default Slave	00C _H	U, SV	SV, P	3	Page 2-51
ERRADDRD	Arbiter Address Error/Debug Capture Register Default Slave	010 _H	U, SV	SV, P	3	Page 2-53

On-Chip System Buses and Bus Bridges
Table 2-9 Registers Overview - Aurix_Bus Module Control Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
ERRD	Arbiter Error/Debug Capture Register Default Slave	014 _H	U, SV	SV, P	3	Page 2-54
DBCON	Arbiter x Debug Control Register	018 _H	U, SV	SV, P	1	Page 2-56
DBADD	Arbiter x Debug Address Register	01C _H	U, SV	SV, P	1	Page 2-59
DBMADD	Arbiter x Debug Mask Address Register	020 _H	U, SV	SV, P	1	Page 2-63
-	Reserved	024 _H - 040 _H	BE	BE	-	-
ARBCON0	Arbiter 0 Control Register	044 _H	U, SV	SV, P	3	Page 2-47
PRIOH0	Arbiter 0 Priority Register High	048 _H	U, SV	SV, P	3	Page 2-49
PRIOL0	Arbiter 0 Priority Register Low	04C _H	U, SV	SV, P	3	Page 2-51
ERRADDR0	Arbiter 0 Address Error/Debug Capture Register	050 _H	U, SV	SV, P	3	Page 2-53
ERR0	Arbiter 0 Error/Debug Capture Register	054 _H	U, SV	SV, P	3	Page 2-54
DBCON0	Arbiter 0 Debug Control Register	058 _H	U, SV	SV, P	1	Page 2-56
DBADD0	Arbiter 0 Debug Address Register	05C _H	U, SV	SV, P	1	Page 2-59
DBMADD0	Arbiter 0 Debug Mask Address Register	060 _H	U, SV	SV, P	1	Page 2-64
-	Reserved	064 _H - 080 _H	BE	BE	-	-
ARBCON1	Arbiter 1 Control Register	084 _H	U, SV	SV, P	3	Page 2-47
PRIOH1	Arbiter 1 Priority Register High	088 _H	U, SV	SV, P	3	Page 2-49

On-Chip System Buses and Bus Bridges
Table 2-9 Registers Overview - Aurix_Bus Module Control Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
PRIOL1	Arbiter 1 Priority Register Low	08C _H	U, SV	SV, P	3	Page 2-51
ERRADDR1	Arbiter 1 Address Error/Debug Capture Register	090 _H	U, SV	SV, P	3	Page 2-53
ERR1	Arbiter 1 Error/Debug Capture Register	094 _H	U, SV	SV, P	3	Page 2-54
DBCON1	Arbiter 1 Debug Control Register	098 _H	U, SV	SV, P	1	Page 2-56
DBADD1	Arbiter 1 Debug Address Register	09C _H	U, SV	SV, P	1	Page 2-60
DBMADD1	Arbiter 1 Debug Mask Address Register	0A0 _H	U, SV	SV, P	1	Page 2-65
-	Reserved	0A4 _H - 140 _H	BE	BE	-	-
ARBCON4	Arbiter 4 Control Register	144 _H	U, SV	SV, P	3	Page 2-47
PRIOH4	Arbiter 4 Priority Register High	148 _H	U, SV	SV, P	3	Page 2-49
PRIOL4	Arbiter 4 Priority Register Low	14C _H	U, SV	SV, P	3	Page 2-51
ERRADDR4	Arbiter 4 Address Error/Debug Capture Register	150 _H	U, SV	SV, P	3	Page 2-53
ERR4	Arbiter 4 Error/Debug Capture Register	154 _H	U, SV	SV, P	3	Page 2-54
DBCON4	Arbiter 4 Debug Control Register	158 _H	U, SV	SV, P	1	Page 2-56
DBADD4	Arbiter 4 Debug Address Register	15C _H	U, SV	SV, P	1	Page 2-61
DBMADD4	Arbiter 4 Debug Mask Address Register	160 _H	U, SV	SV, P	1	Page 2-66
-	Reserved	164 _H - 1C0 _H	BE	BE	-	-

On-Chip System Buses and Bus Bridges
Table 2-9 Registers Overview - Aurix_Bus Module Control Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
ARBCON6	Arbiter 6 Control Register	1C4 _H	U, SV	SV, P	3	Page 2-47
PRIOH6	Arbiter 6 Priority Register High	1C8 _H	U, SV	SV, P	3	Page 2-49
PRIOL6	Arbiter 6 Priority Register Low	1CC _H	U, SV	SV, P	3	Page 2-51
ERRADDR6	Arbiter 6 Address Error/Debug Capture Register	1D0 _H	U, SV	SV, P	3	Page 2-53
ERR6	Arbiter 6 Error/Debug Capture Register	1D4 _H	U, SV	SV, P	3	Page 2-54
DBCON6	Arbiter 6 Debug Control Register	1D8 _H	U, SV	SV, P	1	Page 2-56
DBADD6	Arbiter 6 Debug Address Register	1DC _H	U, SV	SV, P	1	Page 2-61
DBMADD6	Arbiter 6 Debug Mask Address Register	1E0 _H	U, SV	SV, P	1	Page 2-66
-	Reserved	1E4 _H - 200 _H	BE	BE	-	-
ARBCON7	Arbiter 7 Control Register	204 _H	U, SV	SV, P	3	Page 2-47
PRIOH7	Arbiter 7 Priority Register High	208 _H	U, SV	SV, P	3	Page 2-49
PRIOL7	Arbiter 7 Priority Register Low	20C _H	U, SV	SV, P	3	Page 2-51
ERRADDR7	Arbiter 7 Address Error/Debug Capture Register	210 _H	U, SV	SV, P	3	Page 2-53
ERR7	Arbiter 7 Error/Debug Capture Register	214 _H	U, SV	SV, P	3	Page 2-54
DBCON7	Arbiter 7 Debug Control Register	218 _H	U, SV	SV, P	1	Page 2-56
DBADD7	Arbiter 7 Debug Address Register	21C _H	U, SV	SV, P	1	Page 2-62

On-Chip System Buses and Bus Bridges
Table 2-9 Registers Overview - Aurix_Bus Module Control Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
DBMADD7	Arbiter 7 Debug Mask Address Register	220 _H	U, SV	SV, P	1	Page 2-67
-	Reserved	224 _H - 240 _H	BE	BE	-	-
ARBCON8	Arbiter 8 Control Register	244 _H	U, SV	SV, P	3	Page 2-47
PRIOH8	Arbiter 8 Priority Register High	248 _H	U, SV	SV, P	3	Page 2-49
PRIOL8	Arbiter 8 Priority Register Low	24C _H	U, SV	SV, P	3	Page 2-51
ERRADDR8	Arbiter 8 Address Error/Debug Capture Register	250 _H	U, SV	SV, P	3	Page 2-53
ERR8	Arbiter 8 Error/Debug Capture Register	254 _H	U, SV	SV, P	3	Page 2-54
DBCON8	Arbiter 8 Debug Control Register	258 _H	U, SV	SV, P	1	Page 2-56
DBADD8	Arbiter 8 Debug Address Register	25C _H	U, SV	SV, P	1	Page 2-63
DBMADD8	Arbiter 8 Debug Mask Address Register	260 _H	U, SV	SV, P	1	Page 2-68
-	Reserved	264 _H - 3FF _H	BE	BE	-	-

1) The absolute register address is calculated as follows:
Module Base Address ([Table 2-9](#)) + Offset Address (shown in this column)

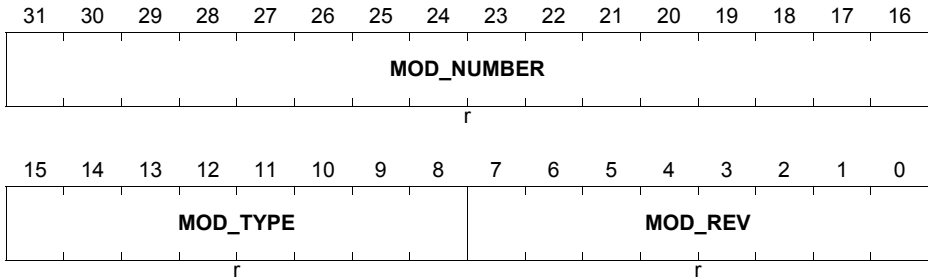
2) This register is located inside the default slave

On-Chip System Buses and Bus Bridges
2.2.9.1 TC26x B-Step Control Registers

The identification register allows the programmer version-tracking of the module. The table below shows the identification register which is implemented in the LBCU module.

XBAR_ID

Module Identification Register (408_H) **Reset Value: 0004 D0XX_H**



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MOD_TYPE	[15:8]	r	Module Type The bit field is set to C0 _H which defines the module as a 32-bit module.
MOD_NUMBER	[31:16]	r	Module Number Value This bit field defines a module identification number. The value for the LBCU module is 000F _H .

On-Chip System Buses and Bus Bridges

XBAR_DBSAT
Debug Trigger Event Status Register (40C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
rw															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCID			0				SCI8	SCI7	SCI6	0	SCI4	0		SCI1	SCI0
rwh			rw				rwh	rwh	rwh	rw	rwh	rw		rwh	rwh

Field	Bits	Type	Description
SCIn (n = 0-1)	n	rwh	SCI Debug Trigger Event Status 0 _B No Debug Trigger Event was detected for SCIn by its arbiter. 1 _B A Debug Trigger Event was detected for SCIn by its arbiter. Writing a '1' to this bit clears the bit.
SCI4	4	rwh	SCI Debug Trigger Event Status 0 _B No Debug Trigger Event was detected for SCI4 by its arbiter. 1 _B A Debug Trigger Event was detected for SCI4 by its arbiter. Writing a '1' to this bit clears the bit.
SCIn (n = 6-8)	n	rwh	SCI Debug Trigger Event Status 0 _B No Debug Trigger Event was detected for SCIn by its arbiter. 1 _B A Debug Trigger Event was detected for SCIn by its arbiter. Writing a '1' to this bit clears the bit.
SCID	15	rwh	Default Slave Debug Trigger Event Status 0 _B No Debug Trigger Event was detected for the default slave by its arbiter. 1 _B A Debug Trigger Event was detected for the default slave by its arbiter. Writing a '1' to this bit clears the bit.

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
0	[31:16], [14:9], 5, [3:2]	rw	Reserved Read as 0; must be written with 0.

Note: This register is not reset with the normal system reset as all other registers in the XBar_SRI. This register is only reset with the special debug reset.

XBAR_INTSAT
Arbiter Interrupt Status Register
(410_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRS CID			0				PRS CI8	PRS CI7	PRS CI6	0	PRS CI4	0		PRS CI1	PRS CI0
rwh			r				rwh	rwh	rwh	r	rwh	r		rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCS CID			0				SCS CI8	SCS CI7	SCS CI6	0	SCS CI4	0		SCS CI1	SCS CI0
rwh			r				rwh	rwh	rwh	r	rwh	r		rwh	rwh

Field	Bits	Type	Description
SCSCIn (n = 0-1)	n	rwh	Starvation Error from SCIn Status 0_B No starvation error is pending from SCIn 1_B A starvation error is pending from SCIn Writing a zero to the bit leaves the content unchanged. Writing a one to the bit clears it. In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
SCSCI4	4	rwh	<p>Starvation Error from SCI4 Status</p> <p>0_B No starvation error is pending from SCI4</p> <p>1_B A starvation error is pending from SCI4</p> <p>Writing a zero to the bit leaves the content unchanged.</p> <p>Writing a one to the bit clears it.</p> <p>In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</p>
SCSCIn (n = 6-8)	n	rwh	<p>Starvation Error from SCIn Status</p> <p>0_B No starvation error is pending from SCIn</p> <p>1_B A starvation error is pending from SCIn</p> <p>Writing a zero to the bit leaves the content unchanged.</p> <p>Writing a one to the bit clears it.</p> <p>In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</p>
SCSCID	15	rwh	<p>Starvation Error from Default Slave Status</p> <p>0_B No starvation error is pending from default slave</p> <p>1_B A starvation error is pending from default slave</p> <p>Writing a zero to the bit leaves the content unchanged.</p> <p>Writing a one to the bit clears it.</p> <p>In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</p>
PRSCIn (n = 0-1)	n+16	rwh	<p>Protocol Error from SCIn Status</p> <p>0_B No protocol error is pending from SCIn</p> <p>1_B A protocol error is pending from SCIn</p> <p>Writing a zero to the bit leaves the content unchanged.</p> <p>Writing a one to the bit clears it.</p> <p>In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</p>

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
PRSCI4	20	rwh	Protocol Error from SCI4 Status 0 _B No protocol error is pending from SCI4 1 _B A protocol error is pending from SCI4 Writing a zero to the bit leaves the content unchanged. Writing a one to the bit clears it. In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.
PRSCIn (n = 6-8)	n+16	rwh	Protocol Error from SCIn Status 0 _B No protocol error is pending from SCIn 1 _B A protocol error is pending from SCIn Writing a zero to the bit leaves the content unchanged. Writing a one to the bit clears it. In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.
PRSCID	31	rwh	Protocol Error from Default Slave Status 0 _B No protocol error is pending from default slave 1 _B A protocol error is pending from default slave Writing a zero to the bit leaves the content unchanged. Writing a one to the bit clears it. In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.
0	[30:25], 21, [19:18], [14:9], 5, [3:2]	r	Reserved Read as 0; should be written with 0.

Note: Only the bits assigned to configured SCIs are implemented. Not implemented bits treated as reserved bits, read as '0', should be written with '0'.

On-Chip System Buses and Bus Bridges

XBAR_IDINTSAT
Transaction ID Interrupt Status Register(414_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	IDM CI13	IDM CI12	0	0	IDM CI9	IDM CI8	0	0	IDM CI5	IDM CI4	0	0	0	0	IDM CI0
r	rwh	rwh	r	r	rwh	rwh	r	r	rwh	rwh	r	r	r	r	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDSC ID	0	0	0	0	0	IDSC I8	IDSC I7	IDSC I6	0	IDCS CI4	0	0	0	IDSC I1	IDSC I0
rwh	r	r	r	r	r	rwh	rwh	rwh	r	rwh	r	r	r	rwh	rwh

Field	Bits	Type	Description
IDSCIn (n = 0-1)	n	rwh	Transaction ID Error from SCIn Status 0 _B No transaction ID error is pending from SCIn 1 _B A transaction ID error is pending from SCIn Writing a zero to the bit leaves the content unchanged. Writing a one to the bit clears it. <i>Note: In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</i>
IDCSCI4	4	rwh	Transaction ID Error from SCI4 Status 0 _B No transaction ID error is pending from SCI4 1 _B A transaction ID error is pending from SCI4 Writing a zero to the bit leaves the content unchanged. Writing a one to the bit clears it. <i>Note: In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</i>

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
IDSCIn (n = 6-8)	n	rwh	<p>Transaction ID Error from SCIn Status</p> <p>0_B No transaction ID error is pending from SCIn</p> <p>1_B A transaction ID error is pending from SCIn</p> <p>Writing a zero to the bit leaves the content unchanged.</p> <p>Writing a one to the bit clears it.</p> <p><i>Note: In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</i></p>
IDSCID	15	rwh	<p>Transaction ID Error from Default Slave Status</p> <p>0_B No transaction ID error is pending from default slave</p> <p>1_B A transaction ID error is pending from default slave</p> <p>Writing a zero to the bit leaves the content unchanged.</p> <p>Writing a one to the bit clears it.</p> <p>In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</p>
IDMCIO	16	rwh	<p>Transaction ID Error from MCIO Status</p> <p>0_B No transaction ID error is pending from MCIn</p> <p>1_B A transaction ID error is pending from MCIn</p> <p>Writing a zero to the bit leaves the content unchanged.</p> <p>Writing a one to the bit clears it.</p> <p><i>Note: In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</i></p>
IDMCIn (n = 4-5)	n+16	rwh	<p>Transaction ID Error from MCIn Status</p> <p>0_B No transaction ID error is pending from MCIn</p> <p>1_B A transaction ID error is pending from MCIn</p> <p>Writing a zero to the bit leaves the content unchanged.</p> <p>Writing a one to the bit clears it.</p> <p><i>Note: In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</i></p>

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
IDMCIn (n = 8-9)	n+16	rwh	Transaction ID Error from MCIn Status 0 _B No transaction ID error is pending from MCIn 1 _B A transaction ID error is pending from MCIn Writing a zero to the bit leaves the content unchanged. Writing a one to the bit clears it. <i>Note: In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</i>
IDMCIn (n = 12-13)	n+16	rwh	Transaction ID Error from MCIn Status 0 _B No transaction ID error is pending from MCIn 1 _B A transaction ID error is pending from MCIn Writing a zero to the bit leaves the content unchanged. Writing a one to the bit clears it. <i>Note: In case of a parallel clearing via software and an error from the hardware the bit remains set and is not cleared.</i>
0	[31:30], [27:26], [23:22], [19:17], [14:9], 5, [3:2]	r	Reserved Read as 0; should be written with 0.

Note: Only the bits assigned to configured SCIs are implemented. Not implemented bits treated as reserved bits, read as '0', should be written with '0'.

On-Chip System Buses and Bus Bridges

XBAR_IDINTEN
Transaction ID Interrupt Enable Register(418_H)
Reset Value: 3331 81D3_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	ENM CI13	ENM CI12	0	0	ENM CI9	ENM CI8	0	0	ENM CI5	ENM CI4	0	0	0	0	ENM CI0
r	rw	rw	r	r	rw	rw	r	r	rw	rw	r	r	r	r	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENS CID	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
rw	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
ENSCIn (n = 0-1)	n	rw	Enable ID Error from SCIn 0 _B No transaction ID error from SCIn are sampled 1 _B A transaction ID error from SCIn are sampled
ENSCI4	4	rw	Enable ID Error from SCI4 0 _B No transaction ID error from SCI4 are sampled 1 _B A transaction ID error from SCI4 are sampled
ENSCIn (n = 6-8)	n	rw	Enable ID Error from SCIn 0 _B No transaction ID error from SCIn are sampled 1 _B A transaction ID error from SCIn are sampled
ENSCID	15	rw	Enable ID Error from Default Slave 0 _B No transaction ID error from the default slave is sampled 1 _B A transaction ID error from the default slave is sampled
ENMCIO	16	rw	Enable ID Error from MCIO 0 _B No transaction ID error from MCIn are sampled 1 _B A transaction ID error from MCIn are sampled
ENMCIn (n = 4-5)	n+16	rw	Enable ID Error from MCIn 0 _B No transaction ID error from MCIn are sampled 1 _B A transaction ID error from MCIn are sampled

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
ENMCIn (n = 8-9)	n+16	rw	Enable ID Error from MCIn 0 _B No transaction ID error from MCIn are sampled 1 _B A transaction ID error from MCIn are sampled
ENMCIn (n = 12-13)	n+16	rw	Enable ID Error from MCIn 0 _B No transaction ID error from MCIn are sampled 1 _B A transaction ID error from MCIn are sampled
0	[31:30], [27:26], [23:22], [19:17], [14:9], 5, [3:2]	r	Reserved Read as 0; should be written with 0.

Note: Only the bits assigned to configured SCIs are implemented. Not implemented bits treated as reserved bits, read as '0', should be written with '0'.

Note: Reset values for bits/bit fields coupled to masters or slaves that are not configured or enabled are zero.

XBAR_EXTCOND
External Control Register D
(000_H)
Reset Value: 0000 0200_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												MAX_WS			
rw												rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAX_WS			0		NOR MW	NOD ELT R	0		FRE QDIS F	0		WF WD	0		
rw			rw		rw	rw	rw		rw	rw		rw	rw		

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
WFWD	3	rw	Wait for FPI Write Data For FPI-Bus block write transfers the transaction request can be delayed until all write data arrived from the FPI-Bus in the SFI. As on the FPI-Bus side very slow masters can resident SRI-Bus slaves can be blocked for many SRI-Bus cycles if the write transaction is started with the first write data 0 _B Write transactions on the SRI-Bus are requested with the first received write data from the FPI-Bus (default) 1 _B Write transactions on the SRI-Bus are requested with the last received write data from the FPI-Bus
FREQDISF	6	rw	Disable Fast Request Feature for FPI to SRI Transactions 0 _B Fast request feature is enabled (default) 1 _B Fast request feature is disabled
NODELTR	9	rw	Control Signal for deferred transactions 0 _B Deferred Transactions are generated 1 _B Deferred Transactions are not generated (default)
NORMW	10	rw	Control Signal for deferred transactions 0 _B Deferred Transactions are generated for RMW (default) 1 _B Deferred Transactions are not generated for RMW
MAX_WS	[19:13]	rw	FPI-Bus Wait State Retry Ratio SFI slave interface retries a read transaction to the FPI bus after the programmed value of wait states to change the read transaction in a delayed read transaction.
0	[31:20], [12:11], [8:7], [5:4], [2:0]	rw	Reserved Read as 0; shall be written with 0.

On-Chip System Buses and Bus Bridges

Note: Only the bits assigned to configured SCIs are implemented. Not implemented bits treated as reserved bits, read as '0', should be written with '0'.

Note: Reset values for bits/bit fields coupled to masters or slaves that are not configured or enabled are zero.

XBAR_ARBCONx (x = 0-1)

Arbiter Control Register x (044_H+x*40_H) **Reset Value: FFF0 0003_H**

XBAR_ARBCON4

Arbiter Control Register 4 (144_H) **Reset Value: FFF0 0003_H**

XBAR_ARBCONx (x = 6-8)

Arbiter Control Register x (044_H+x*40_H) **Reset Value: FFF0 0003_H**

XBAR_ARBCOND

Arbiter Control Register D (004_H) **Reset Value: FFF0 0003_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SPC												0			
rw												r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											INTA CK	SET SCIN T	SET PRIN T	SCE RRE N	PRE RRE N
r											rwh	rwh	rwh	rw	rw

Field	Bits	Type	Description
PRERREN	0	rw	SRI Protocol Error Enable 0 _B Protocol errors are not recognized and no information is captured. 1 _B Protocol errors are recognized and information is captured.
SCERREN	1	rw	SRI Starvation Error Enable 0 _B Starvation based errors are not recognized and no information is captured. 1 _B Starvation based errors are recognized and information is captured.

On-Chip System Buses and Bus Bridges

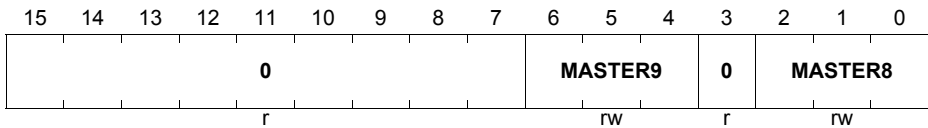
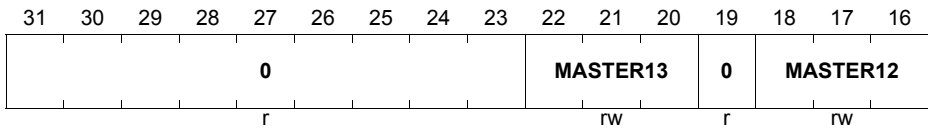
Field	Bits	Type	Description
SETPRINT	2	rwh	Set SRI Protocol Interrupt 0 _B No protocol interrupt is generated 1 _B A protocol interrupt is generated After the interrupt is generated by set the bit it's automatically cleared by the hardware
SETSCINT	3	rwh	Set SRI Starvation Interrupt 0 _B No starvation interrupt is generated 1 _B A starvation interrupt is generated After the interrupt is generated by set the bit it's automatically cleared by the hardware
INTACK	4	rwh	Interrupt Acknowledge 0 _B Default value 1 _B An Error for this arbiter is pending. The ERRADDR and ERR registers are not updated for new errors. Writing a one to this bit field while it's set have the following results: The error lock of registers ERRADDR and ERR are released and the register could be updated with the next interrupt request detected (see Chapter 2.2.7.5). In the cycle after the write action the hardware automatically clears the bit.
SPC	[31:20]	rw	Starvation Protection Counter Reload Value The reload value defines the period for the starvation protection.
0	[19:5]	r	Reserved Read as 0; should be written with 0.

Note: Only the bits assigned to configured SCIs are implemented. Bits for non configured SCIs are treated as reserved bits.

Note: The 'D' at ARBCOND stands for Default Slave.

On-Chip System Buses and Bus Bridges

XBAR_PRIOHx (x = 0-1)
Arbiter Priority Register x (048_H+x*40_H) Reset Value: 0055 0055_H
XBAR_PRIOH4
Arbiter Priority Register 4 (148_H) Reset Value: 0055 0055_H
XBAR_PRIOHx (x = 6-8)
Arbiter Priority Register x (048_H+x*40_H) Reset Value: 0055 0055_H
XBAR_PRIOHD
Arbiter Priority Register D (008_H) Reset Value: 0055 0055_H



Field	Bits	Type	Description
MASTER8	[2:0]	rw	Master 8 Priority (Priority of CPU1.DMI access) This bit field contains the master priority for the arbitration used by the arbiter of slave x. For each master a unique number for this slave has to be used. A lower number has a higher priority in the arbitration round than a higher one.
MASTER9	[6:4]	rw	Master 9 Priority (Priority of CPU1.PMI access) This bit field contains the master priority for the arbitration used by the arbiter of slave x. For each master a unique number for this slave has to be used. A lower number has a higher priority in the arbitration round than a higher one.

On-Chip System Buses and Bus Bridges

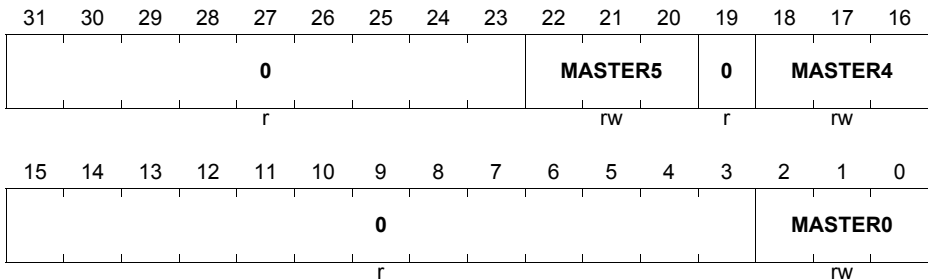
Field	Bits	Type	Description
MASTER12	[18:16]	rw	Master 12 Priority (Priority of CPU0.DMI access) This bit field contains the master priority for the arbitration used by the arbiter of slave x. For each master a unique number for this slave has to be used. A lower number has a higher priority in the arbitration round than a higher one.
MASTER13	[22:20]	rw	Master 13 Priority (Priority of CPU0.PMI access) This bit field contains the master priority for the arbitration used by the arbiter of slave x. For each master a unique number for this slave has to be used. A lower number has a higher priority in the arbitration round than a higher one.
0	[31:23], 19, [15:7], 3	r	Reserved Read as 0; should be written with 0.

Note: Only the bits assigned to configured MCIs are implemented. Bits for non configured SCIs are treated as reserved bits.

Note: The 'D' at XBAR_PRIOLD stands for Default Slave.

Note: Reset values for bits/bit fields coupled to masters or slaves that are not configured or enabled are zero.

On-Chip System Buses and Bus Bridges

XBAR_PRIOLD
Arbiter Priority Register D (00C_H) Reset Value: 0022 0002_H
XBAR_PRIOLx (x = 0-1)
Arbiter Priority Register x (04C_H+x*40_H) Reset Value: 0022 0002_H
XBAR_PRIOL4
Arbiter Priority Register 4 (14C_H) Reset Value: 0022 0002_H
XBAR_PRIOLx (x = 6-8)
Arbiter Priority Register x (04C_H+x*40_H) Reset Value: 0022 0002_H


Field	Bits	Type	Description
MASTER0	[2:0]	rw	Master 0 Priority (Priority of DMA access) This bit field contains the master priority for the arbitration used by the arbiter of slave x. For each master a unique number for this slave has to be used. A lower number has a higher priority in the arbitration round than a higher one.
MASTER4	[18:16]	rw	Master 4 Priority (Priority of HSSL access) This bit field contains the master priority for the arbitration used by the arbiter of slave x. For each master a unique number for this slave has to be used. A lower number has a higher priority in the arbitration round than a higher one.

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
MASTER5	[22:20]	rw	Master 5 Priority (Priority of SFI access) This bit field contains the master priority for the arbitration used by the arbiter of slave x. For each master a unique number for this slave has to be used. A lower number has a higher priority in the arbitration round than a higher one.
0	[31:23], [15:3], 19	r	Reserved Read as 0; should be written with 0.

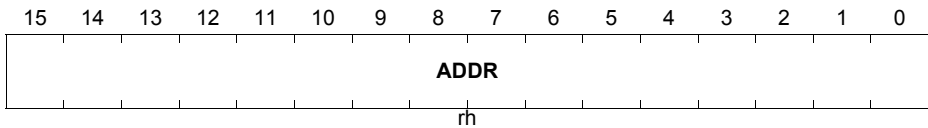
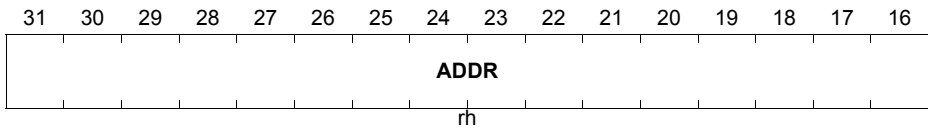
Note: Only the bits assigned to configured MCIs are implemented. Bits for non configured SCIs are treated as reserved bits.

Note: The 'D' at XBAR_PRIOLD stands for Default Slave.

Note: Reset values for bits/bit fields coupled to masters or slaves that are not configured or enabled are zero.

On-Chip System Buses and Bus Bridges

XBAR_ERRADDRD	Error/Debug Address Capture Register D(010 _H)	Reset Value: 7000 0000 _H
XBAR_ERRADDR0	Error/Debug Address Capture Register 0(050 _H)	Reset Value: 6000 0000 _H
XBAR_ERRADDR1	Error/Debug Address Capture Register 1(090 _H)	Reset Value: 5000 0000 _H
XBAR_ERRADDR4	Error/Debug Address Capture Register 4(150 _H)	Reset Value: 8000 0000 _H
XBAR_ERRADDR6	Error/Debug Address Capture Register 6(1D0 _H)	Reset Value: 8800 0000 _H
XBAR_ERRADDR7	Error/Debug Address Capture Register 7(210 _H)	Reset Value: 8020 0000 _H
XBAR_ERRADDR8	Error/Debug Address Capture Register 8(250 _H)	Reset Value: 8040 0000 _H



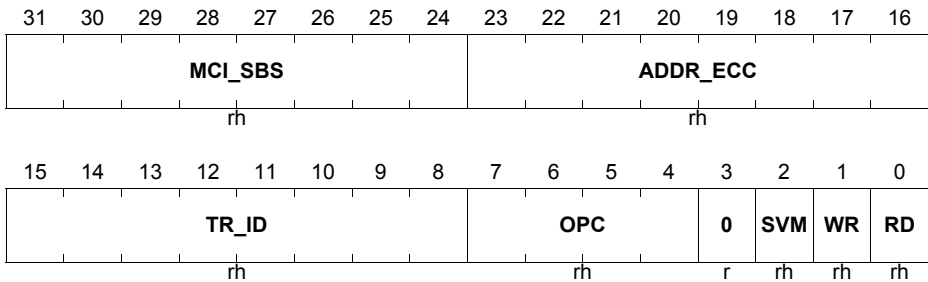
Field	Bits	Type	Description
ADDR	[31:0]	rh	Transaction Address This bifield contains the address of the erroneous transaction from the address phase

Note: The default value can differ from the one shown here because a constant can be used to reduce the number of compared bits in the arbitration if a slave occupies only a limited address area. For more details see the design specification of the XBar_SRI.

Note: The 'D' at XBAR_ERRADDRD stands for Default Slave.

On-Chip System Buses and Bus Bridges

XBAR_ERRx (x = 0-1)		
Error/Debug Capture Register x	(054_H+x*40_H)	Reset Value: 0000 0000_H
XBAR_ERR4		
Error/Debug Capture Register 4	(154_H)	Reset Value: 0000 0000_H
XBAR_ERRx (x = 6-8)		
Error/Debug Capture Register x	(054_H+x*40_H)	Reset Value: 0000 0000_H
XBAR_ERRD		
Error/Debug Capture Register D	(014_H)	Reset Value: 0000 0000_H



Field	Bits	Type	Description
RD	0	rh	Read Indication Status 0 _B The read indication SRI-Bus signal line was asserted (read or start of RMW transaction) 1 _B The read indication SRI-Bus signal line was deasserted (no read transaction)
WR	1	rh	Write Indication Status 0 _B The write indication SRI-Bus signal line was asserted (write or start of RMW transaction) 1 _B The write indication SRI-Bus signal line was deasserted (no write transaction)
SVM	2	rh	Supervisor Mode Indication Status 0 _B The supervisor mode indication SRI-Bus signal line was deasserted 1 _B The supervisor mode indication SRI-Bus signal line was asserted
OPC	[7:4]	rh	Operation Code This bit field contains the op-code of the erroneous transaction.

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
TR_ID	[15:8]	rh	Transaction ID This bit field contains the transaction ID of the erroneous transaction from the address phase. The Transaction ID is build out of an 6 bit unique TAG ID TR_ID[5:0] and a 2 bit running number TR_ID[7:6] (see also Chapter 2.6).
ADDR_ECC	[23:16]	rh	SRI Address Phase ECC This bit field contains the Address Phase Error ECC of the erroneous transaction <i>Note: In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.</i>
MCI_SBS	[31:24]	rh	MCI Sideband Signals [7:0] This bit field contains the MCI Sideband Signals [7:0] that are related to the address phase informations captured by the ERRD/ERRADDR registers. In the TC26x B-Step the sideband signals are used by the DMA SRI master interface to provide information about the DMA requestor of a DMA transaction (for the encoding see Table 2-5).
0	3	r	Reserved Read as 0; should be written with 0

Note: The 'D' at XBAR_ERRD stands for Default Slave.

On-Chip System Buses and Bus Bridges

XBAR_DBCONx (x = 0-1)
Debug Control Register x (058_H+x*40_H) **Reset Value: 0000 0000_H**
XBAR_DBCON4
Debug Control Register 4 (158_H) **Reset Value: 0000 0000_H**
XBAR_DBCONx (x = 6-8)
Debug Control Register x (058_H+x*40_H) **Reset Value: 0000 0000_H**
XBAR_DBCOND
Debug Control Register D (018_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0		MASTER						MAS EN	0	ERR EN	ADD EN	SVM EN	WRE N	RDE N		
r		rw						rw	r	rw	rw	rw	rw	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0												SET DBE VT	REA RM	DBS AT	DBE N	
r												w	w	rh	r	

Field	Bits	Type	Description
DBEN	0	r	Status of OCDS Enable Signal Displays the value of the OCDS enable signal from Cerberus.
DBSAT	1	rh	Debug (OCDS) Trigger Status 0 _B The debug (OCDS) trigger was used and has to be rearmed 1 _B The debug (OCDS) trigger is armed

On-Chip System Buses and Bus Bridges

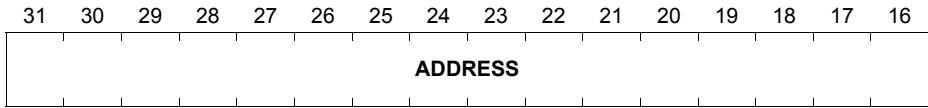
Field	Bits	Type	Description
REARM	2	w	<p>Rearm Debug (OCDS) Trigger</p> <p>0_B Read back value</p> <p>1_B Writing a one to this bit arms sets bit DBCON.DBSAT.</p> <p>Writing a one to this bit field while it's set have the following results: The debug lock of registers ERRADDR and ERR are released and the register could be updated with the next debug event request detected (see Chapter 2.2.7.5).</p> <p>In the cycle after the write action the hardware automatically clears the bit.</p> <p><i>Note: This bit is automatically reset by the hardware after DBCON.DBSAT was set.</i></p>
SETDBEVT	3	w	<p>Set Debug Event</p> <p>0_B Default value</p> <p>1_B A debug trigger event will be generated by this arbiter if the debug feature is enabled (DBCON.ENST is set). The registers ERR and ERRADD capture the status if not locked already.</p> <p><i>Note: This bit is automatically reset by the hardware.</i></p>
RDEN	16	rw	<p>Read Trigger Enable</p> <p>0_B Read Transaction are not used to trigger the debug trigger event (OCDS)</p> <p>1_B Read Transaction are used to trigger the debug trigger event (OCDS)</p>
WREN	17	rw	<p>Write Trigger Enable</p> <p>0_B Write Transaction are not used to trigger the debug trigger event (OCDS)</p> <p>1_B Write Transaction are used to trigger the debug trigger event (OCDS)</p>
SVMEN	18	rw	<p>SVM Trigger Enable</p> <p>0_B SVM Transaction are not used to trigger the debug trigger event (OCDS)</p> <p>1_B SVM Transaction are used to trigger the debug trigger event (OCDS)</p>

On-Chip System Buses and Bus Bridges

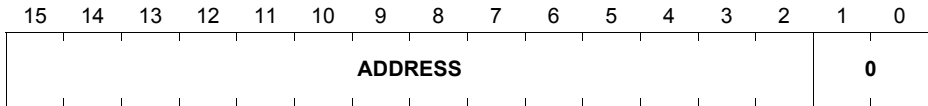
Field	Bits	Type	Description
ADDEN	19	rw	Address Trigger Enable 0 _B Transaction addresses are not used to trigger the debug trigger event (OCDS) 1 _B Transaction address defined by the registers DBADD and DBMADD are used to trigger the debug trigger event (OCDS)
ERREN	20	rw	Error Trigger Enable 0 _B Errored Transactions are not used to trigger the debug trigger event (OCDS) 1 _B Errored Transactions are used to trigger the debug trigger event (OCDS) Reading this bit return always zero. <i>Note: Protocol errors, starvation errors and transaction ID errors can be used where, but have to enabled before as usual in registers ARBCON or IDINTEN depending on the error type.</i>
MASEN	23	rw	Master Trigger Enable 0 _B The Master TAG ID as defined in the bit field MASTER is not used to trigger the debug trigger event (OCDS) 1 _B The Master TAG ID as defined in the bit field MASTER is used to trigger the debug trigger event (OCDS)
MASTER	[29:24]	rw	Master TAG ID Trigger Selector The value of this bit field define the Master TAG ID within the address phase of an SRI transaction to the related SRI slave module that triggers the debug trigger event (OCDS). The Master TAG IDs are defined here: Table 2-15 .
0	[31:30], [22:21], [15:4]	r	Reserved Read as 0; should be written with 0.

Note: This register is not reset with the normal system reset as all other registers in the XBar_SRI. This register is only reset with the special debug reset.

Note: The 'D' at XBAR_DBCOND stands for Default Slave.

On-Chip System Buses and Bus Bridges
XBAR_DBADDD
Debug Address Register D
(01C_H)
Reset Value: 0000 0000_H


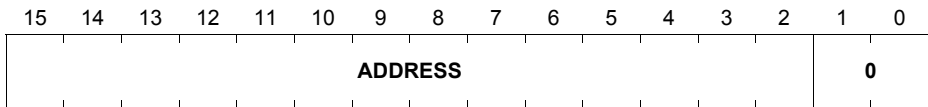
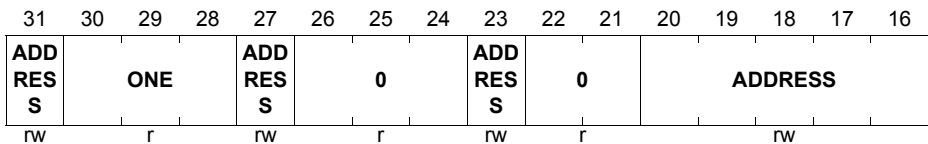
rw



rw

r

Field	Bits	Type	Description
ADDRESS	[31:2]	rw	Debug Address Boundary
0	[1:0]	r	Reserved Read as 0; should be written with 0.

Note: This register is reset with the debug reset (Class 1 reset).
Note: The last 'D' at XBAR_DBADDD stands for Default Slave.
XBAR_DBADD0
Debug Address Register 0
(05C_H)
Reset Value: 7000 0000_H


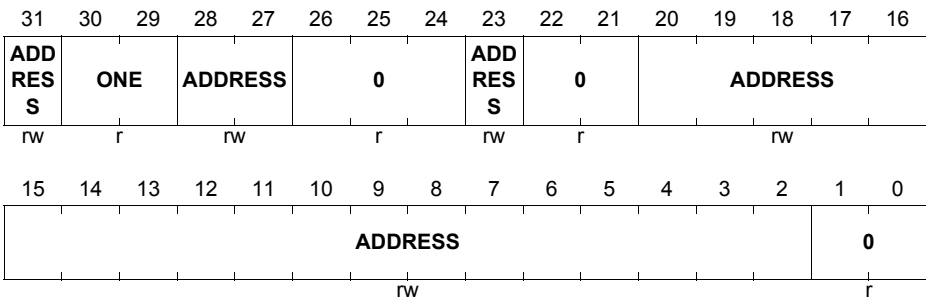
rw

r

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
ADDRESS, ADDRESS, ADDRESS, ADDRESS	27, 23, [20:2], 31	rw	Debug Address Boundary
ONE	[30:28]	r	Reserved Read as 1; should be written with 0.
0	[26:24], [22:21], [1:0]	r	Reserved Read as 0; should be written with 0.

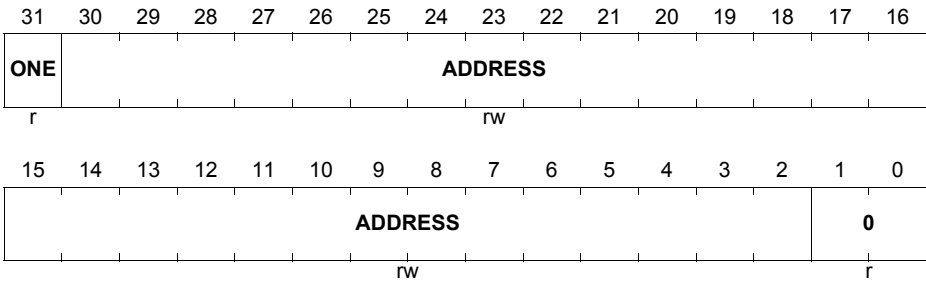
Note: This register is reset with the debug reset (Class 1 reset).

XBAR_DBADD1
Debug Address Register 1
(09C_H)
Reset Value: 6000 0000_H


Field	Bits	Type	Description
ADDRESS, ADDRESS, ADDRESS, ADDRESS	31, [28:27], 23, [20:2]	rw	Debug Address Boundary
ONE	[30:29]	r	Reserved Read as 1; should be written with 0.
0	[26:24], [22:21], [1:0]	r	Reserved Read as 0; should be written with 0.

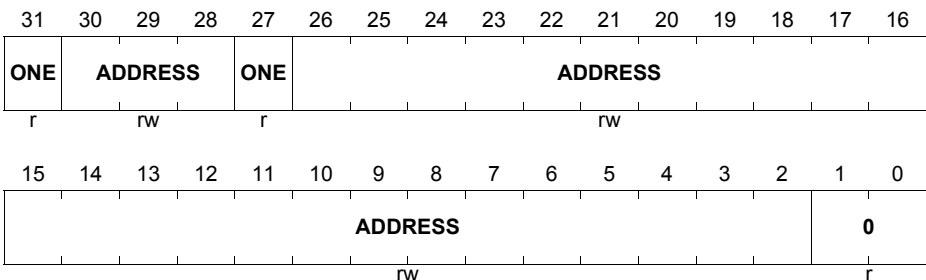
On-Chip System Buses and Bus Bridges

Note: This register is reset with the debug reset (Class 1 reset).

XBAR_DBADD4
Debug Address Register 4
(15C_H)
Reset Value: 8000 0000_H


Field	Bits	Type	Description
ADDRESS	[30:2]	rw	Debug Address Boundary
ONE	31	r	Reserved Read as 1; should be written with 0.
0	[1:0]	r	Reserved Read as 0; should be written with 0.

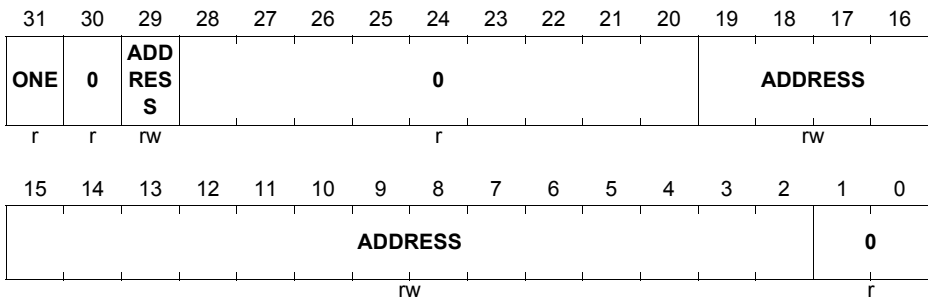
Note: This register is reset with the debug reset (Class 1 reset).

XBAR_DBADD6
Debug Address Register 6
(1DC_H)
Reset Value: 8800 0000_H


On-Chip System Buses and Bus Bridges

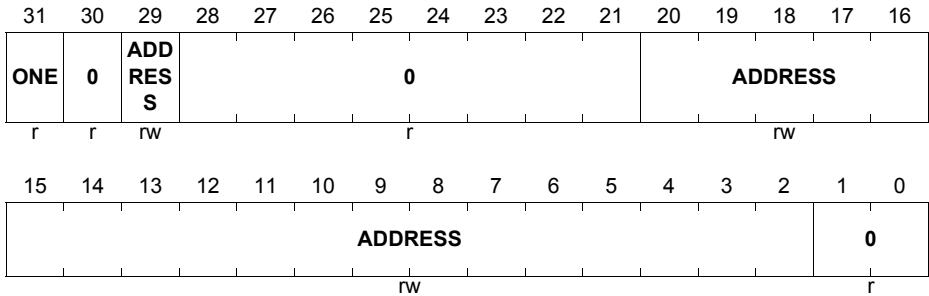
Field	Bits	Type	Description
ADDRESS, ADDRESS	[30:28], [26:2]	rw	Debug Address Boundary
ONE, ONE	31, 27	r	Reserved Read as 1; should be written with 0.
0	[1:0]	r	Reserved Read as 0; should be written with 0.

Note: This register is reset with the debug reset (Class 1 reset).

XBAR_DBADD7
Debug Address Register 7
(21C_H)
Reset Value: 8000 0000_H


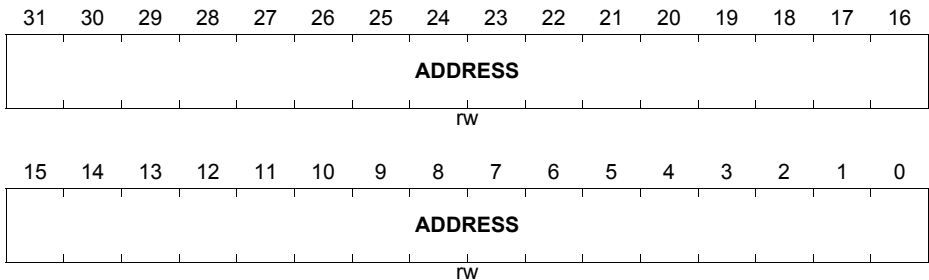
Field	Bits	Type	Description
ADDRESS, ADDRESS	29, [19:2]	rw	Debug Address Boundary
ONE	31	r	Reserved Read as 1; should be written with 0.
0	30, [28:20], [1:0]	r	Reserved Read as 0; should be written with 0.

Note: This register is reset with the debug reset (Class 1 reset).

On-Chip System Buses and Bus Bridges
XBAR_DBADD8
Debug Address Register 8
(25C_H)
Reset Value: 8000 0000_H


Field	Bits	Type	Description
ADDRESS, ADDRESS	29, [20:2]	rw	Debug Address Boundary
ONE	31	r	Reserved Read as 1; should be written with 0.
0	30, [28:21], [1:0]	r	Reserved Read as 0; should be written with 0.

Note: This register is reset with the debug reset (Class 1 reset).

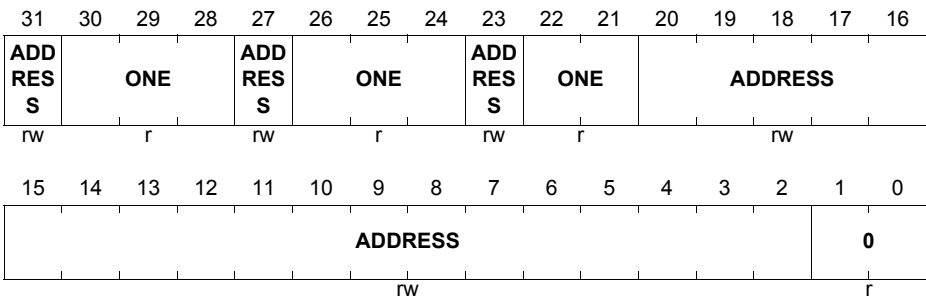
XBAR_DBMADD
Debug Mask Address Register D
(020_H)
Reset Value: 0000 0000_H


On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
ADDRESS	[31:0]	rw	Debug Address Boundary

Note: This register is reset with the debug reset (Class 1 reset).

Note: The last 'D' at XBAR_DBMADDD stands for Default Slave.

XBAR_DBMADDD
Debug Mask Address Register 0
(060_H)
Reset Value: 7760 0000_H


Field	Bits	Type	Description
ADDRESS, ADDRESS, ADDRESS, ADDRESS	[20:2], 27, 23, 31	rw	Debug Address Boundary
ONE, ONE, ONE	[30:28], [26:24], [22:21]	r	Reserved Read as 1; should be written with 0.
0	[1:0]	r	Reserved Read as 0; should be written with 0.

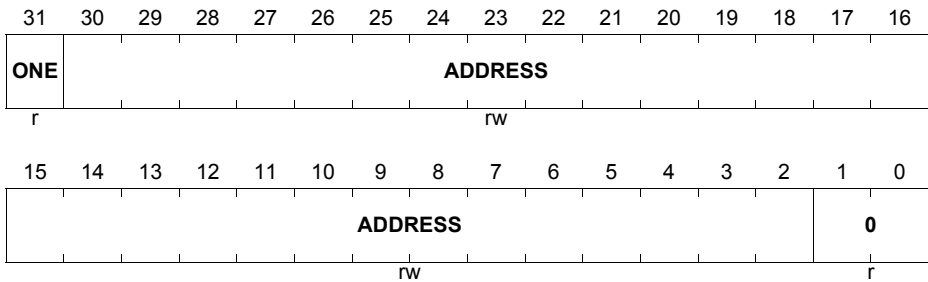
Note: This register is reset with the debug reset (Class 1 reset).

On-Chip System Buses and Bus Bridges
XBAR_DBMADD1
Debug Mask Address Register 1
(0A0_H)
Reset Value: 6760 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADD RES S	ONE		ADDRESS			ONE		ADD RES S	ONE		ADDRESS				
rw	r		rw			r		rw	r		rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRESS														0	
rw														r	

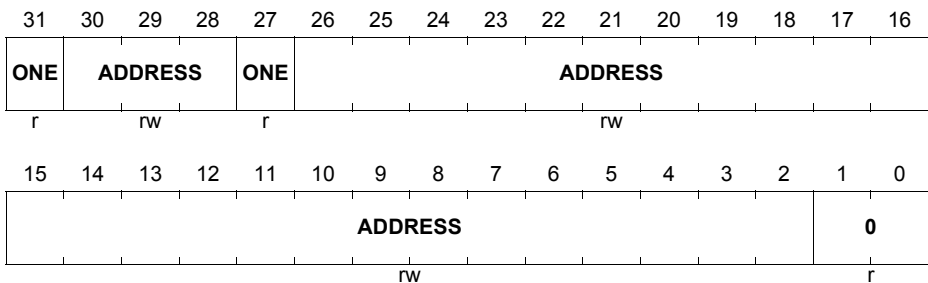
Field	Bits	Type	Description
ADDRESS, ADDRESS, ADDRESS, ADDRESS	31, [28:27], 23, [20:2]	rw	Debug Address Boundary
ONE, ONE, ONE	[30:29], [26:24], [22:21]	r	Reserved Read as 1; should be written with 0.
0	[1:0]	r	Reserved Read as 0; should be written with 0.

Note: This register is reset with the debug reset (Class 1 reset).

On-Chip System Buses and Bus Bridges
XBAR_DBMADD4
Debug Mask Address Register 4
(160_H)
Reset Value: 8000 0000_H


Field	Bits	Type	Description
ADDRESS	[30:2]	rw	Debug Address Boundary
ONE	31	r	Reserved Read as 1; should be written with 0.
0	[1:0]	r	Reserved Read as 0; should be written with 0.

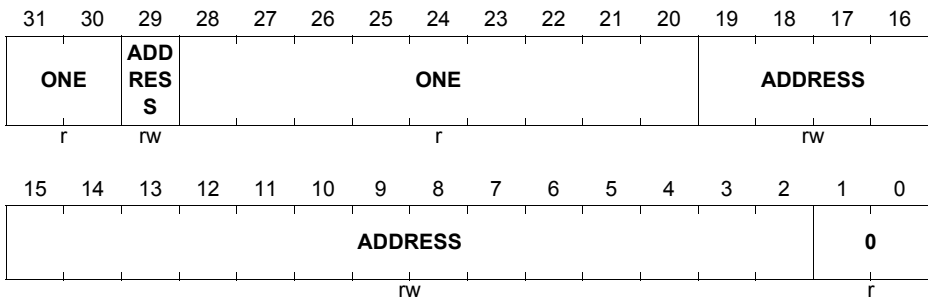
Note: This register is reset with the debug reset (Class 1 reset).

XBAR_DBMADD6
Debug Mask Address Register 6
(1E0_H)
Reset Value: 8800 0000_H


On-Chip System Buses and Bus Bridges

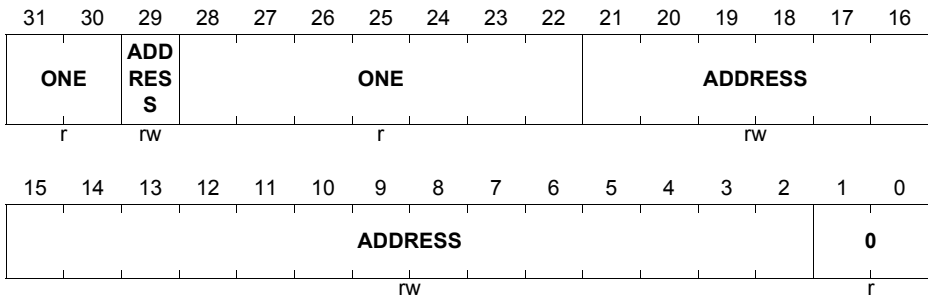
Field	Bits	Type	Description
ADDRESS, ADDRESS	[30:28], [26:2]	rw	Debug Address Boundary
ONE, ONE	31, 27	r	Reserved Read as 1; should be written with 0.
0	[1:0]	r	Reserved Read as 0; should be written with 0.

Note: This register is reset with the debug reset (Class 1 reset).

XBAR_DBMADD7
Debug Mask Address Register 7
(220_H)
Reset Value: DFF0 0000_H


Field	Bits	Type	Description
ADDRESS, ADDRESS	29, [19:2]	rw	Debug Address Boundary
ONE, ONE	[31:30], [28:20]	r	Reserved Read as 1; should be written with 0.
0	[1:0]	r	Reserved Read as 0; should be written with 0.

Note: This register is reset with the debug reset (Class 1 reset).

On-Chip System Buses and Bus Bridges
XBAR_DBMADD8
Debug Mask Address Register 8
(260_H)
Reset Value: DFC0 0000_H


Field	Bits	Type	Description
ADDRESS, ADDRESS	29, [21:2]	rw	Debug Address Boundary
ONE, ONE	[31:30], [28:22]	r	Reserved Read as 1; should be written with 0.
0	[1:0]	r	Reserved Read as 0; should be written with 0.

Note: This register is reset with the debug reset (Class 1 reset).

Access Enable Register 0 (ACCEN0)

The Access Enable Register 0 controls write access for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ... ,EN31 -> TAG ID 011111_B.

On-Chip System Buses and Bus Bridges
XBAR_ACCEN0
Access Enable Register 0
(4FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN3	EN3	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN1	EN1	EN1	EN1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN1	EN1	EN1	EN1	EN1	EN1	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
5	4	3	2	1	0										
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register 1 (ACCEN1)

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000_B, EN1 -> TAG ID 100001_B, ... ,EN31 -> TAG ID 111111_B.

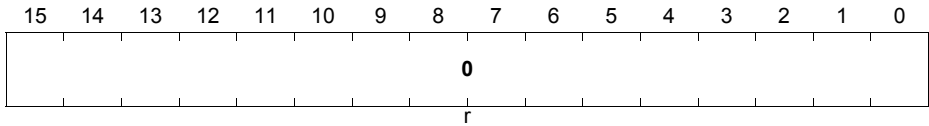
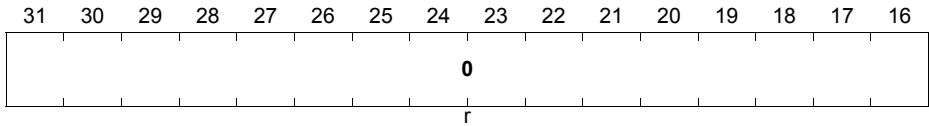
On-Chip System Buses and Bus Bridges

XBAR_ACCEN1

Access Enable Register 1

(4F8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

On-Chip System Buses and Bus Bridges

2.3 Shared Resource Interconnect to FPI Bus Interface (SFI Bridge)

This section describes the basic functionality of the SFI Bridge.

2.3.1 Functional Overview

The SFI Bridge is implemented as an uni-directional bus bridge that forwards transactions from the System Peripheral Bus (SPB) to the SRI Interconnect. The bridge supports all transactions types of both the SRI Bus and FPI Bus.

The bridge is transparent, this means that the address of a transaction and the master TAG of a bus master is forwarded to the other side of the bridge. Addresses are only changed by the bridge where it is required by the transaction conversion from the 64 bit SRI Interconnect to the 32 bit SPB.

Bus Errors at Writes via the SFI Bridge

Write transactions are handled as posted writes. The SFI is able to buffer multiple posted writes. This means that a write operation from the SPB through the SFI Bridge to the SRI Interconnect can be finished on SPB and then generated on the SRI autonomously by the SFI. If this write operation results in a bus error on the SRI the Error information is not passed back to the SPB bus. This is also valid for transactions from SRI to SPB via SFI bridge. The bus error is detected by the on chip bus control logic of the target on chip bus system (BCU_FPI on the SPB, XBar_SRI on the SRI) which can generate an interrupt.

Note that this behavior occurs only at write operations via the SFI Bridge. It can also be triggered by an erroneous write cycle of a read-modify-write bus transaction.

2.4 System Peripheral Bus

The TC26x B-Step has one on-chip FPI Bus:

- System Peripheral Bus (SPB)
 - System bus for on-chip peripherals

This section gives an overview of the on-chip FPI Bus. It describes its bus control units, the bus characteristics, bus arbitration, scheduling, prioritizing, error conditions, and debugging support.

2.4.1 Overview

The FPI Bus interconnects the on-chip peripheral functional units with the TC26x B-Step processor subsystem.

The FPI Bus is designed to be quick to be acquired by on-chip functional units, and quick to transfer data. The low setup overhead of the FPI Bus access protocol guarantees fast FPI Bus acquisition, which is required for time-critical applications.

The FPI Bus is designed to sustain high transfer rates. For example, a peak transfer rate of up to 320 Mbyte/s can be achieved with the 32-bit data bus at 80 MHz bus clock. Multiple data transfers per bus arbitration cycle allow the FPI Bus to operate close to its peak bandwidth.

Additional features of the FPI Bus include:

- Optimized for high speed and high performance
- Support of multiple bus masters and pipelined transactions
- 32-bit wide address and data buses
- 8-, 16-, and 32-bit data transfers
- 64-, 128-, and 256-bit block transfers
- Central simple per-cycle arbitration
- Slave-controlled wait state insertion
- Support of atomic operations LDMST, ST.T and SWAP.W
- Starvation prevention mechanism that can take care that even low priority requests will be granted after a configurable number of arbitration cycles, permanently enabled
- Default slave that takes over transactions no other slave responds
- Timeout detection and handling
- Capturing of transaction information in case of a bus error that can be released by again by SW incl. transaction address, control incl. op-code, data
- Address Phase includes Supervisor Mode information
- All SPB (FPI) slave modules implemented with a TAG ID based access protection that provides a generic write protection for the control registers

The functional units of the TC26x B-Step are connected to the FPI Bus via FPI Bus interfaces. An FPI Bus interfaces acts as bus agents, requesting bus transactions on behalf of their functional unit, or responding to bus transaction requests.

On-Chip System Buses and Bus Bridges

There are two types of bus agents:

- FPI Bus master agents can initiate FPI Bus transactions and can also act as slaves.
- Slave agents can only react and respond to FPI Bus transaction requests in order to read or write internal registers of slave modules as for example memories.

When an FPI Bus master attempts to initiate a transfer on the FPI Bus, it first signals a request for bus ownership to the bus control unit (SBCU). When bus ownership is granted by the SBCU, an FPI Bus read or write transaction is initiated. The unit targeted by the transaction becomes the FPI Bus slave, and responds with the requested action.

Some functional units operate only as slaves, while others can operate as either masters or slaves on the FPI Bus.

FPI Bus arbitration is performed by the Bus Control Unit (SBCU) of the FPI Bus. In case of bus errors, the SBCU generates an interrupt request to the CPU and provides debugging information about the actual bus error to the CPU.

2.4.2 Bus Transaction Types

This section describes the SPB transaction types.

Single Transfers

Single transfers are byte, half-word, and word transactions that target any slave connected to SPB. Note that the SFI Bridge operates as an SPB master.

Block Transfers

Block transfers operate in principle in the same way as single transfers do, but one address phase is followed by multiple data phases. Block transfers can be composed of 2 word, 4 word, or 8 word transfers.

Note: In general, block transfers (2 word, 4 word, or 8 word) cannot be executed in the TC26x B-Step with peripheral units that operate as FPI Bus slaves during an FPI Bus transaction.

Block transfers are initiated by the following CPU instructions: LD.D, LD.DA, MOV.D, ST.D and ST.DA. Additionally there are communication peripherals that are able to generate block transfers (e.g. Ethernet).

Atomic Transfers

Atomic transfers are generated by LDMST, ST.T and SWAP.W instructions that require two single transfers. The read and write transfer of an atomic transfer are always locked and cannot be interrupted by another bus masters. Atomic transfers are also referenced as read-modify-write transfers.

Note: See also [Table 2-11](#) for available FPI Bus transfer types.

2.4.3 Reaction of a Busy Slave

If an FPI Bus slave is busy at an incoming FPI Bus transaction request, it can delay the execution of the FPI Bus transaction. The requesting FPI Bus master releases the FPI Bus for one cycle after the FPI Bus transaction request, in order to allow the FPI Bus slave to indicate if it is ready to handle the requested FPI Bus transaction. This sequence is repeated as long as the slave indicates that it is busy.

Note: For the FPI Bus default master, the one cycle gap does not result in a performance loss because it is granted the FPI Bus in this cycle as default master if no other master requests the FPI Bus for some other reasons.

On-Chip System Buses and Bus Bridges

2.4.4 Address Alignment Rules

FPI Bus address generation is compliant with the following rules:

- Half-word transactions must have a half-word aligned address ($A_0 = 0$). Half-word accesses on byte lanes 1 and 2 addresses are illegal.
- Word transactions must always have word-aligned addresses ($A[1:0] = 00_B$).
- Block transactions must always have block-type aligned addresses.

2.4.5 FPI Bus Basic Operations

This section describes some basic transactions on the FPI Bus.

The example in **Figure 2-9** shows the three cycles of an FPI Bus operation:

1. **Request/Grant Cycle:** The FPI Bus master attempts to perform a read or write transfer and requests for the FPI Bus. If the FPI Bus is available, it is granted in the same cycle by the FPI Bus controller.
2. **Address Cycle:** After the request/grant cycle, the master puts the address on the FPI Bus, and all FPI Bus slave devices check whether they are addressed for the following data cycle.
3. **Data Cycle:** In the data cycle, either the master puts write data on the FPI Bus which is read by the FPI Bus slave (write cycle) or vice versa (read cycle).

Transfers 2 and 3 show the conflict when two master try to use the FPI Bus and how the conflict is resolved. In the example, the FPI Bus master of transfer 2 has a higher priority than the FPI Bus master of transfer 3.

Bus Cycle	1	2	3	4	5
Transfer 1	Request/Grant	Address Cycle	Data Cycle		
Transfer 2		Request/Grant	Address Cycle	Data Cycle	
Transfer 3		Request/Grant		Address Cycle	Data Cycle

MCA06109

Figure 2-9 Basic FPI Bus Transactions

At a block transfer, the address cycle of a second transfer is extended until the data cycles of the block transfer are finished. In the example of **Figure 2-10**, transfer 1 is a block transfer, while transfer 2 is a single transfer.

On-Chip System Buses and Bus Bridges

Bus Cycle	1	2	3	4	5	6	7
Transfer 1	Request/ Grant	Address Cycle	Data Cycle	Data Cycle	Data Cycle	Data Cycle	
Transfer 2		Request/ Grant	Address Cycle				Data Cycle

MCA06110

Figure 2-10 FPI Bus Block Transactions

2.5 FPI Bus Control Unit (SBCU)

The TC26x B-Step incorporates one FPI Bus control Unit (BCU) for the FPI based System Peripheral Bus (SPB), called System Bus Control Unit (SBCU).

2.5.1 FPI Bus Arbitration

The arbitration unit of the BCU determines whether it is necessary to arbitrate for FPI Bus ownership, and, if so, which available bus requestor gets the FPI Bus for the next data transfer. During arbitration, the bus is granted to the requesting agent with the highest priority. If no request is pending, the bus is granted to a default master. If no bus master takes the bus, the BCU itself will set the FPI Bus into an idle state.

2.5.1.1 Arbitration on the System Peripheral Bus

The TC26x B-Step SPB has multiple bus agents that can become SPB master:

- Each agent is assigned to 4-bit priority bit field in the PRIOH or in the PRIOL register
- The value in a priority bit field defines the related SPB master agent priority
- A lower priority number has a higher priority, '0' is the highest priority
- Each priority bit field in the PRIOH and PRIOL has a unique default value that can be changed during ramp up

Note: The DMA controller as an bus agent supports three priorities as this module covers two DMA Move Engines but also the Cerberus (JTAG/L1 Debug Interface) which might be used with the highest or lowest priority. Additionally the DMA Move Engine channels can be assigned to one of the three priorities.

2.5.1.2 Default Master

If no request is active, the FPI-Bus will be granted to a Default Master. The FPI master that was most recently granted will be granted to the Default Master. After reset, the master with the priority 0 will be the Default Master.

2.5.1.3 Arbitration Algorithms

The arbitration algorithm implemented in the BCU is based on:

- Priority driven arbitration of master agents with a pending request
- Starvation Prevention mechanism can increase master agent priorities during Starvation Prevention process

The default priority of each connected FPI master agent can be changed during runtime via the arbiter priority registers (PRIOH, PRIOL, see 'Changing Master Priorities'). There is a 4-bit priority bit field for each master agent in the PRIOH/PRIOL registers that defines the priority of this for this arbiter. After reset, each priority register has a default value, which means that each FPI master has unique default priority.

On-Chip System Buses and Bus Bridges

The default priority settings should be optimal for most applications. Where required, the arbitration setting can be adapted to the specific application needs (see 'Changing Master Priorities').

It must be ensured that two FPI master agents don't have the same priority.

Priority Driven Arbitration

The general arbitration algorithm is priority driven where priority 0 is the highest priority and 15 the lowest one. If multiple masters are requesting for one slave, the master with the highest priority will win the next arbitration round (see also 'starvation prevention').

Changing Master Priorities

Master priorities must be configured during ramp up as long only one FPI master agent is active (CPU0). Master priorities must not be changed while multiple FPI master agents are active / enabled.

2.5.1.4 Starvation Prevention

Starvation prevention is a feature of the SBCU that can take care that even requesting low priority master agents will be granted after a period, where the period length can be controlled by SBCU control registers. Because the priority assignment of the SPB agents is fixed, it is possible that a lower-priority bus requestor may never be granted the bus if a higher-priority bus requestor continuously asks for, and receives, bus ownership. To protect against bus starvation of lower-priority masters, the starvation prevention mechanism of the SBCU will detect such cases and momentarily raise the priority of the lower-priority requestor to the highest priority (above all other priorities), thereby guaranteeing it access.

Starvation protection employs a counter that is decremented each time an arbitration is performed on the connected FPI bus. The counter is re-loaded with the starvation period value in the SBCU_CON.SPC bit field as long it is enabled SBCU_CON.SPE. When this counter is counted down to zero, for each active bus request a request flag is stored in the BCU. This flag is cleared automatically when a master is granted the bus.

When the next period is finished, an active request of a master from which the request flag was set, a starvation event happened. This master will now be set to the highest priority and will be granted service. If there are several masters to which this starvation condition applies, they are served in the order of their configured priority ranking.

If a master that is processing its transaction under starvation condition is retried, its corresponding request flag is automatically again.

Starvation protection is permanently enabled. The sample period of the counter is programmed through bit field SBCU_CON.SPC. SPC should be set to a value at least greater than or equal to the number of masters. Its reset value is FF_H .

2.5.2 FPI Bus Error Handling

When an error occurs on an FPI Bus, its BCU captures and stores data about the erroneous condition and generates a service request if enabled to do so. The error conditions that force an error-capture are:

- Error Acknowledge: An FPI Bus slave responds with an error to a transaction.
- Un-implemented Address: No FPI Bus slave responds to a transaction request.
- Time-out: A slave does not respond to a transaction request within a certain time window. The number of bus clock cycles that can elapse until a bus time-out is generated is defined by bit field SBCU_CON.TOUT.

When a transaction causes an error, the address and data phase signals of the transaction causing the error are captured and stored in registers.

- The Error Address Capture Register (SBCU_EADD) stores the 32-bit FPI Bus address that has been captured during the erroneous FPI Bus transaction.
- The Error Data Capture Registers (SBCU_EDAT) stores the 32-bit FPI Bus data bus information that has been captured during the erroneous FPI Bus transaction.

On-Chip System Buses and Bus Bridges

- The Error Control Capture Register (SBCU_ECON) stores status information of the bus error event.

If more than one FPI Bus transaction generates a bus error, only the first bus error is captured. After a bus error has been captured, the capture mechanism must be released again by software. The lock is removed by reading the register SBCU_ECON which clears the SBCU_ECON.ERRCNT bit field.

Note: It is recommended to read in a debug session register ECON last as this removes the lock and a new error can already modify the content of the other two registers EDAT and EADD.

If a write transaction from TriCore causes an error on the SPB, the originating master is not informed about this error as it is not an SPB master agent. With each bus error-capture event, a service request is generated, and an interrupt can be generated if enabled and configured in the corresponding service request register.

Interpreting the BCU Control Register Error Information

Although the address and data values captured in registers SBCU_EADD and SBCU_EDAT, respectively, are self-explanatory, the captured FPI Bus control information needs some more explanation.

Register SBCU_ECON captures the state of the read (RDN), write (WRN), Supervisor Mode (SVM), acknowledge (ACK), ready (RDY), abort (ABT), time-out (TOUT), bus master identification lines (TAG) and transaction operation code (OPC) lines of the FPI Bus.

The SVM signal is set to 1 for an access in Supervisor Mode and set to 0 for an access in User Mode. The time-out signal indicates if there was no response on the bus to an access, and the programmed time (via SBCU_TOUT) has elapsed. TOUT is set to 1 in this case. An acknowledge code has to be driven by the selected slave during each data cycle of an access. These codes are listed in [Table 2-10](#).

Table 2-10 FPI Bus Acknowledge Codes

Code (ACK)	Description
00 _B	NSC: No Special Condition.
01 _B	SPT: Split Transaction (not used in the TC26x B-Step).
10 _B	RTY: Retry. Slave can currently not respond to the access. Master needs to repeat the access later.
11 _B	ERR: Bus Error, last data cycle is aborted.

Transactions on the FPI Bus are classified via a 4-bit operation code (see [Table 2-11](#)). Note that split transactions (OPC = 1000_B to 1110_B) are not used in the TC26x B-Step.

Table 2-11 FPI Bus Operation Codes (OPC)

OPC	Description
0000 _B	Single Byte Transfer (8-bit)
0001 _B	Single Half-Word Transfer (16-bit)
0010 _B	Single Word Transfer (32-bit)
0100 _B	2-Word Block Transfer
0101 _B	4-Word Block Transfer
0110 _B	8-Word Block Transfer
1111	No operation
0011 _B , 0111 _B , 1000 _B - 1110 _B	Reserved

2.5.3 System Registers

Figure 2-11 shows these system registers which include registers for:

The SBCU module includes the following TC26x B-Step standard system registers

- Register access protection

The following TC26x B-Step standard system registers are not included:

- Module Clock Control
- Module Kernel Reset
- OCDS Control and Status Register

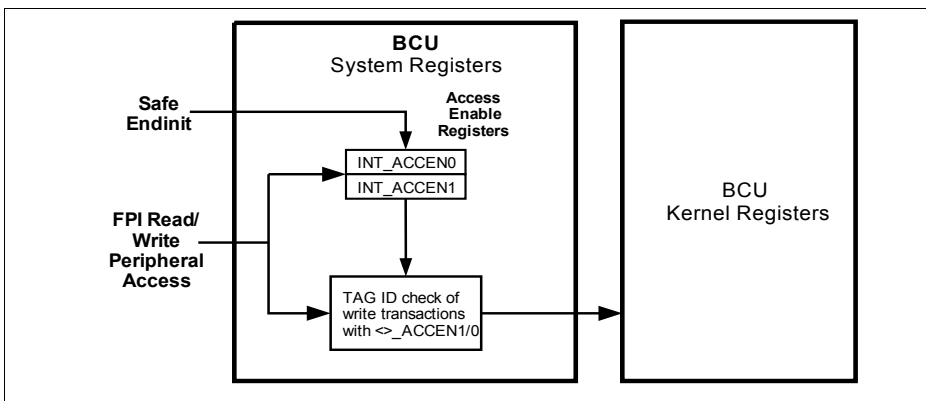


Figure 2-11 TC26x OCB System System Registers

2.5.3.1 Register Access Protection (ACCEN1/0)

The TC26x OCB System module provides a master TAG ID based write access protection as part of the AURIX safety concept. Each on chip resource with direct or indirect bus master capability has a unique master TAG ID that can be used to identify the master of an on chip bus transaction (see also chapter On Chip Bus Systems).

The SRC register is write protected via an On Chip Bus Master TAG-ID protection (see [Chapter 2.5.3.1](#)). This protection is controlled via the Interrupt Router control registers ACCEN10 and ACCEN00.

TAG ID based protection means that the support of write transactions to the TC26x OCB System control registers can be enabled / disabled for each master TAG ID individually. For a disabled master TAG ID, write access will be disconnected with error acknowledge, read access will be processed (see also [Figure 2-11](#)).

The register access protection is controlled via the registers INT_ACCEN1 and INT_ACCEN0 where each bit is related to one encoding of the 6 bit On Chip Master TAG ID.

The INT_ACCEN1/0 registers are controlling the write access to all TC26x OCB System control and system registers with the exception of the INT_ACCEN1/0 registers themselves. INT_ACCEN1/0 are Safety Endinit protected.

After reset, all access enable bits and access control bits are enabled, access protection mechanism has to be configured and checked to bring the system in a safe state.

2.5.3.2 Kernel Reset Registers (KRST1/0, KRSTCLR)

The TC26x OCB System module does not include the kernel reset registers (KRST1, KRST0, KRSTCLR).

Note: The TC26x OCB System module does not support a module kernel reset.

2.5.3.3 Clock Control Register (CLC)

The TC26x OCB System module does not include the module clock control (CLC).

Note: The TC26x OCB System module does not support the Clock Control register functionality which means that the TC26x OCB System module clock can not be disabled by the CLC register.

2.5.3.4 OCDS Control and Status Register (OCS)

The TC26x OCB System module does not include OCDS Control and Status (OCS) register.

Note: The TC26x OCB System module does not support the OCS register functionality.

2.5.4 BCU Debug Support

For debugging purposes, the BCU has the capability for breakpoint generation support. This OCDS debug capability is controlled by the Cerberus module and must be enabled by it (indicated by bit SBCU_DBCNTL.EO).

When BCU debug support has been enabled (EO = 1), any breakpoint request generated by the BCU to the Cerberus disarms the BCU breakpoint logic for further breakpoint requests. In order to rearm the BCU breakpoint logic again for the next breakpoint request generation, bit SBCU_DBCNTL.RA must be set. The status of the BCU breakpoint logic (armed or disarmed) is indicated by bit SBCU_DBCNTL.OA.

There are three types of trigger events:

- Address triggers
- Signal triggers
- Grant triggers

2.5.4.1 Address Triggers

The address debug trigger event conditions are defined by the contents of the SBCU_DBADR1, SBCU_DBADR2, and SBCU_DBCNTL registers. A wide range of possibilities arise for the creation of debug trigger events based on addresses. The following debug trigger events can be selected:

- Match on one signal address
- Match on one of two signal addresses
- Match on one address area
- Mismatch on one address area

Each pair of DBADR_x registers and DBCNTL.ONA_x bits determine one possible debug trigger event. The combination of these two possible debug trigger events defined by DBCNTL.CONCOM1 determine the address debug trigger event condition.

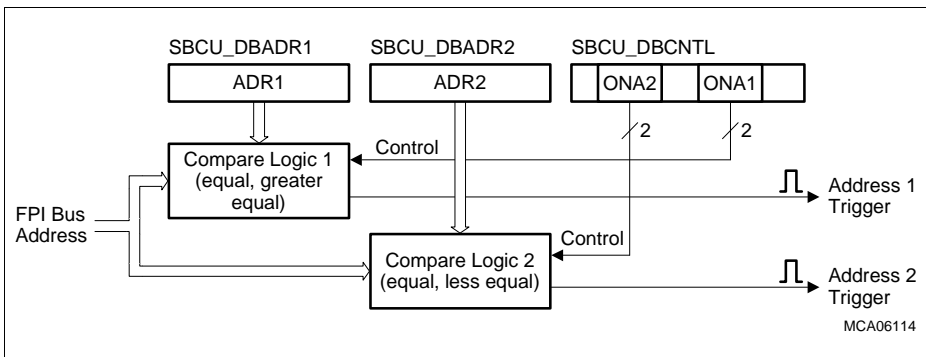


Figure 2-12 Address Trigger Generation

2.5.4.2 Signal Status Triggers

The signal status debug trigger event conditions are defined by the contents of the SBCU_DBBOS and SBCU_DBCNTL registers. Depending on the selected configuration a wide range of possibilities arise for the creation of a debug trigger event based on FPI Bus status signals. Possible combinations are:

- Match on a single signal status
- Match on a multiple signal status

With the multiple signal match conditions, all single signal match conditions are combined with a logical **AND** to the signal status debug trigger event signal. The selection whether or not a single match condition is selected can be enabled/disabled selectively for each condition via the SBCU_DBCNTL.ONBOSx bits.

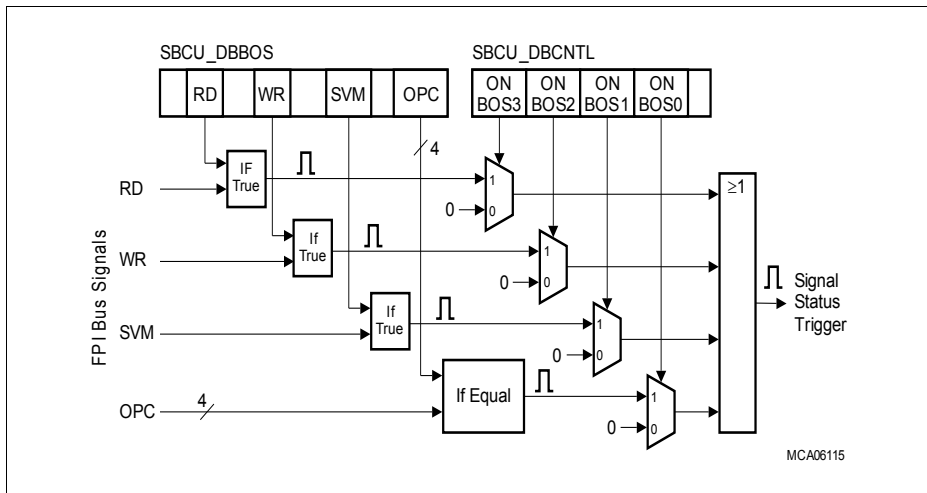


Figure 2-13 Signal Status Trigger Generation

2.5.4.3 Grant Triggers

The signal status debug trigger event conditions are defined via the registers SBCU_DBGRNT and SBCU_DBCNTL. Depending on the configuration of these registers, any combination of FPI Bus master trigger events can be configured. Only the enabled masters in the SBCU_DBGRNT register are of interest for the grant debug trigger event condition. The grant debug trigger event condition can be enabled/disabled via bit SBCU_DBCNTL.ONG (see [Figure 2-14](#)).

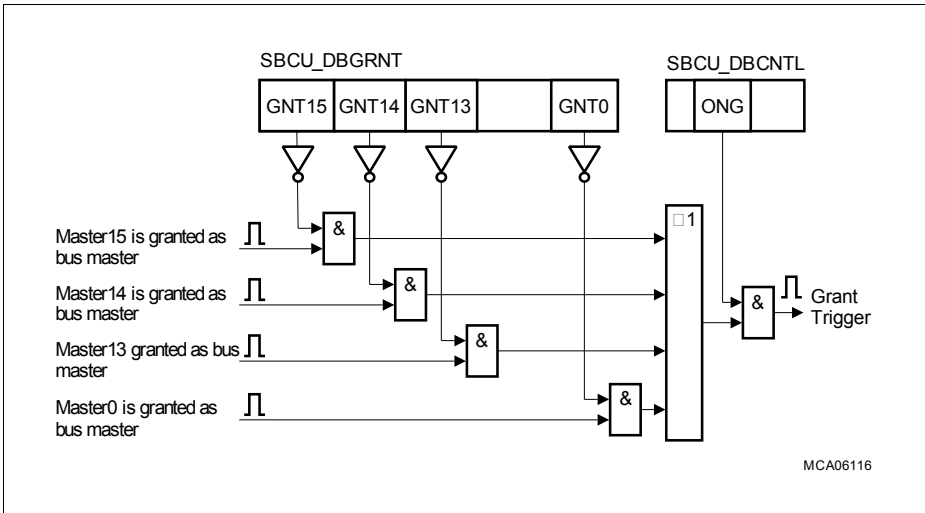


Figure 2-14 Grant Trigger Generation

2.5.4.4 Combination of Triggers

The combination of the four debug trigger signals to the single BCU breakpoint trigger event is defined via the bits CONCOM[2:0] of register SBCU_DBCNTL (see [Figure 2-15](#)). The two address triggers are combined to one address trigger that is further combined with signal status and grant trigger signals. A logical AND or OR combination can be selected for the BCU breakpoint trigger generation.

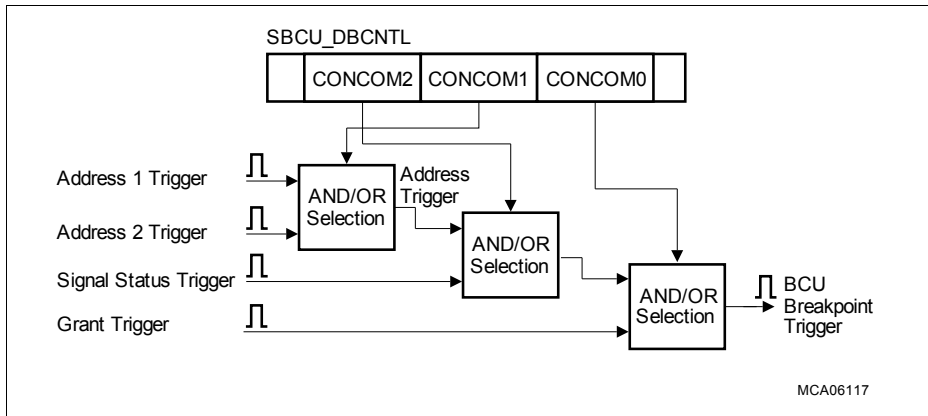


Figure 2-15 BCU Breakpoint Trigger Combination Logic

2.5.4.5 BCU Breakpoint Generation Examples

This section gives three examples of how BCU debug trigger events are programmed.

OCDS Debug Example 1

- Task: Generation of a BCU debug trigger event on any SPB write access to address 00002004_H or 000020A0_H by an SPB master.

For this task, the following programming settings for the BCU breakpoint logic must be executed:

1. Writing SBCU_DBADR1 = 0000 2004_H
2. Writing SBCU_DBADR2 = 0000 20A0_H
3. Writing SBCU_DBCNTL = C1115010_H:
 - a) ONBOS[3:0] = 1100_B means that no signal status trigger is generated (disabled) for a read signal match AND write signal match condition according to the settings of bits RD and WR in register SBCU_DBBOS. Debug trigger event generation for Supervisor Mode signal match and op-code signal match condition is disabled.
 - b) ONA2 = 01_B means that the equal match condition for debug address 2 register is selected.

On-Chip System Buses and Bus Bridges

- c) $ONA1 = 01_B$ means that the equal match condition for debug address 1 register is selected.
 - d) $ONG = 1$ means that the grant debug trigger is enabled.
 - e) $CONCOM[2:0] = 101_B$ means that the address trigger is created by address trigger 1 OR address trigger 2 ($CONCOM1 = 0$), and that the grant trigger is ANDed with the address trigger ($CONCOM0 = 1$), and that the signal status trigger is ANDed with the address trigger ($CONCOM2 = 1$).
 - f) $RA = 1$ means that the BCU breakpoint logic is rearmed.
4. Writing $SBCU_DBGRNT = FFFFFFFD7_H$:
means that the grant trigger for the SPB master is enabled.
 5. Writing $SBCU_DBBOS = 00001000_H$:
means that the signal status trigger is generated on a write transfer and not on a read transfer.

OCDS Debug Example 2

- Task: generation of a BCU debug trigger event on any half-word access in User Mode to address area $01FFFFFF_H$ to $02FFFFFF_H$ by any master.

For this task, the following programming settings for the BCU breakpoint logic must be executed:

1. Writing $SBCU_DBADR1 = 01FFFFFF_H$
2. Writing $SBCU_DBADR2 = 02FFFFFF_H$
3. Writing $SBCU_DBCNTL = 32206010_H$:
 - a) $ONBOS[3:0] = 0011_B$ means that the signal status trigger is disabled for a read or for write signal status match but enabled for Supervisor Mode match AND op-code match conditions according to the settings of bit SVM and bit field OPC in register $SBCU_DBBOS$.
 - b) $ONA2 = 10_B$ means that the address 2 trigger is generated if the FPI Bus address is less or equal to $SBCU_DBADR2$.
 - c) $ONA1 = 10_B$ means that the address 1 trigger is generated if the FPI Bus address is greater or equal to $SBCU_DBADR1$.
 - d) $ONG = 0$ means that the grant debug trigger is disabled.
 - e) $CONCOM[2:0] = 110_B$ means that the address trigger is created by address trigger 1 AND address trigger 2 ($CONCOM1 = 1$), and that the grant trigger is OR-ed with the address trigger ($CONCOM0 = 0$), and that the signal status trigger is AND-ed with the address trigger ($CONCOM2 = 1$).
 - f) $RA = 1$ means that the BCU breakpoint logic is rearmed.
4. Writing $SBCU_DBGRNT = FFFFFFFF_H$:
means that no grant trigger for SPB masters is selected ("don't care" because also disabled by $ONG = 0$).
5. Writing $SBCU_DBBOS = 00000001_H$:
means that the signal status trigger is generated for read ($RD = 0$) and write ($WR = 0$) half-word transfers ($OPC = 0001_B$) in User Mode ($SVM = 0$).

On-Chip System Buses and Bus Bridges

2.5.5 System Bus Control Unit Registers

Figure 2-16 and Table 2-13 are showing the address maps with all registers of the System Bus Control Unit (SBCU) module.

SBCU Control Registers Overview

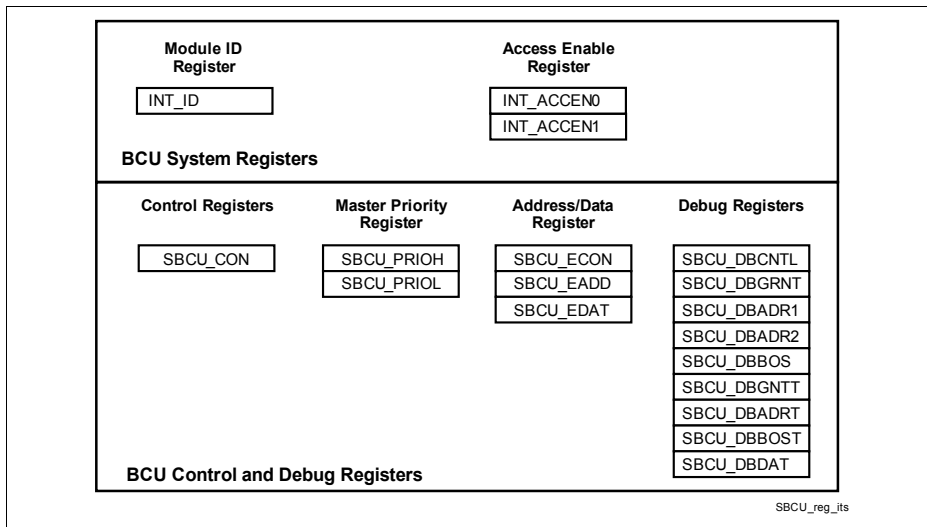


Figure 2-16 SBCU Registers

Table 2-12 Registers Address Space - SBCU Address Space

Module	Base Address	End Address	Note
SBCU	F003 0000 _H	F003 00FF _H	

Table 2-13 Registers Overview - SBCU Control Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
–	Reserved	000 _H - 004 _H	BE	BE	-	-
SBCU_ID	SBCU Module Identification Register	008 _H	U, SV	BE	-	Page 2-91
–	Reserved	00C _H	BE	BE	-	-

On-Chip System Buses and Bus Bridges
Table 2-13 Registers Overview - SBCU Control Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
SBCU_CON	SBCU Control Register	010 _H	U, SV	SV, P	3	Page 2-94
SBCU_PRIOH	Arbiter Priority Register High	014 _H	U, SV	SV, E, P	3	Page 2-95
SBCU_PRIOL	Arbiter Priority Register Low	018 _H	U, SV	SV, E, P	3	Page 2-96
–	Reserved	01C _H	BE	BE	3	-
SBCU_ECON	SBCU Error Control Capture Register	020 _H	U, SV	SV, P	3	Page 2-98
SBCU_EADD	SBCU Error Address Capture Register	024 _H	U, SV	SV, P	3	Page 2-100
SBCU_EDAT	SBCU Error Data Capture Register	028 _H	U, SV	SV, P	3	Page 2-100
–	Reserved	02C _H	BE	BE	-	-
SBCU_DBCNTL	SBCU Debug Control Register	030 _H	U, SV	SV, P	1	Page 2-102
SBCU_DBGRNT	SBCU Debug Grant Mask Register	034 _H	U, SV	SV, P	1	Page 2-105
SBCU_DBADR1	SBCU Debug Address Register 1	038 _H	U, SV	SV, P	1	Page 2-107
SBCU_DBADR2	SBCU Debug Address Register 2	03C _H	U, SV	SV, P	1	Page 2-107
SBCU_DBBOS	SBCU Debug Bus Operation Signals Register	040 _H	U, SV	SV, P	1	Page 2-108
SBCU_DBGNTT	SBCU Debug Trapped Master Register	044 _H	U, SV	BE	1	Page 2-109
SBCU_DBADR1	SBCU Debug Trapped Address Register	048 _H	U, SV	BE	1	Page 2-111
SBCU_DBBOST	SBCU Debug Trapped Bus Operation Signals Register	04C _H	U, SV	BE	1	Page 2-112

On-Chip System Buses and Bus Bridges
Table 2-13 Registers Overview - SBCU Control Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
SBCU_DBDAT	SBCU Debug Data Status Register	050 _H	U, SV	BE	1	Page 2-115
–	Reserved	054 _H - 0F4 _H	BE	BE	-	-
SBCU_ACCEN1	Access Enable Register 1 (Access Mode, Write: P1)	0F8 _H	U, SV	SV, SE	3	Page 2-92
SBCU_ACCEN0	Access Enable Register 0 (Access Mode, Write: P1)	0FC _H	U, SV	SV, SE	3	Page 2-93

1) The absolute register address is calculated as follows:
Module Base Address ([Table 2-12](#)) + Offset Address (shown in this column)

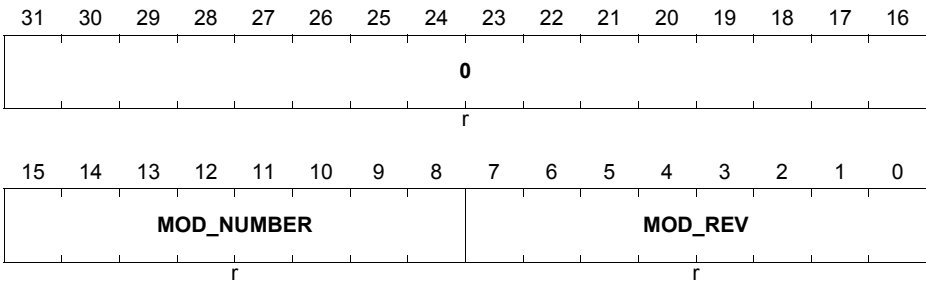
2.5.5.1 SBCU System Registers

Module Identification Register (ID)

The identification register allows the programmer version-tracking of the module. The table below shows the identification register which is implemented in the SBCU module.

SBCU_ID

Module Identification Register (008_H) **Reset Value: 0000 6AXX_H**



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MOD_NUMBER	[15:8]	r	Module Number Value This bit field defines a module identification number. The value for the LBCU module is 006AH.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

Access Enable Register 0 (ACCEN0)

The Access Enable Register 0 controls write access for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The registers ACCEN00 / ACCEN01 are providing one enable bit for each 6-bit On Chip Bus Master TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ..., EN31 -> TAG ID 011111_B.

On-Chip System Buses and Bus Bridges
SBCU_ACCEN0
Access Enable Register 0
(0FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN3	EN3	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN1	EN1	EN1	EN1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN1	EN1	EN1	EN1	EN1	EN1	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
5	4	3	2	1	0										
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register 1 (ACCEN1)

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). ACCEN11/01 are not implemented with register bits as the related On Chip Bus Master TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000_B, EN1 -> TAG ID 100001_B, ... ,EN31 -> TAG ID 111111_B.

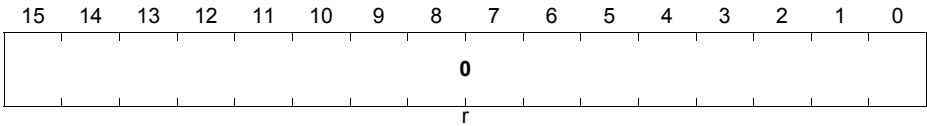
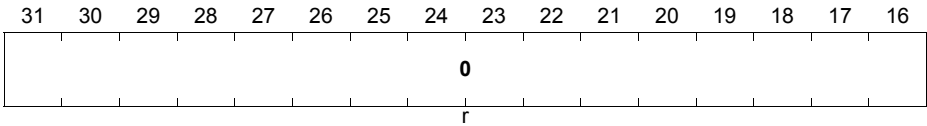
On-Chip System Buses and Bus Bridges

SBCU_ACCEN1

Access Enable Register 1

(0F8_H)

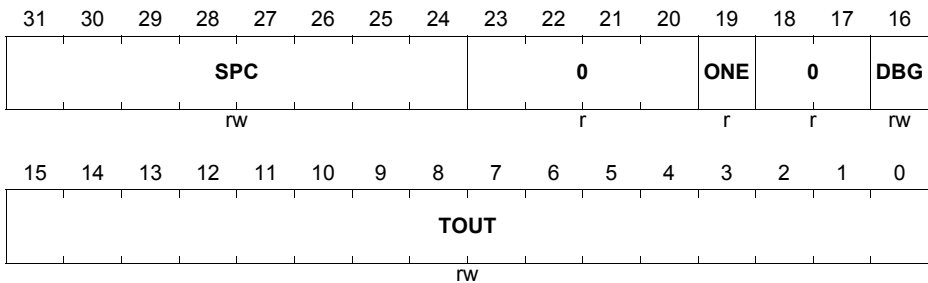
Reset Value: 0000 0000_H



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

On-Chip System Buses and Bus Bridges
2.5.5.2 SBCU Control Registers Descriptions

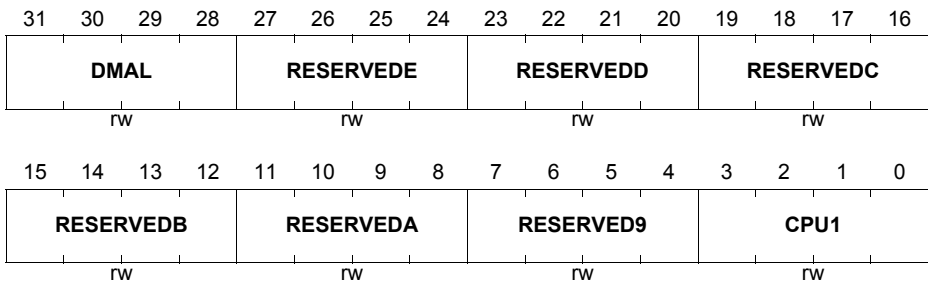
The SBCU Control Register controls the overall operation of the SBCU, including setting the starvation sample period, the bus time-out period, enabling starvation-protection mode, and error handling.

SBCU_CON
SBCU Control Register
(010_H)
Reset Value: FF09 FFFF_H


Field	Bits	Type	Description
TOUT	[15:0]	rw	SBCU Bus Time-Out Value The bit field determines the number of System Peripheral Bus time-out cycles. Default after reset is FFFF _H (= 65536 bus cycles). Please Note: TOUT value must be >= 5.
DBG	16	rw	SBCU Debug Trace Enable The bit enables/disables the error capture mechanism for the registers BCU_ECON, BCU_EADD, BCU_EDAT 0 _B SBCU debug trace disabled 1 _B SBCU debug trace enabled (default after reset) <i>Note: The bit does not affect the SMU alarm or the BCU interrupt that are send in case of an error condition.</i>
SPC	[31:24]	rw	Starvation Period Control Determines the sample period for the starvation counter. Must be larger than the number of masters. The reset value is FF _H .
ONE	19	r	Reserved Read as 1; should be written with 0.

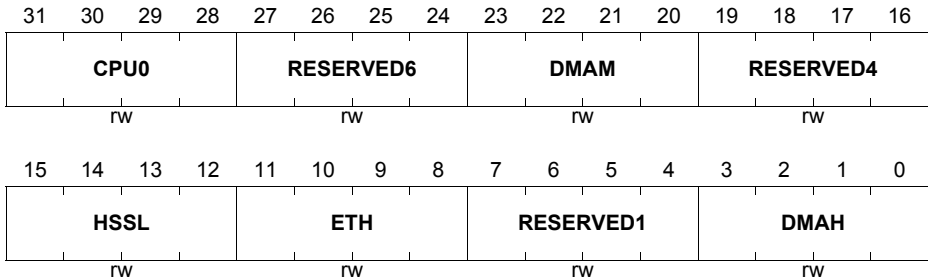
On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
0	[23:20], [18:17]	r	Reserved Read as 0; should be written with 0.

SBCU_PRIOH
Arbiter Priority Register High
(014_H)
Reset Value: FCDE BA98_H


Field	Bits	Type	Description
CPU1	[3:0]	rw	Master 8 Priority This bit field contains the master priority for master connected to BCU request input 8 A lower number has a higher priority in the arbitration round than a higher one.
DMAL	[31:28]	rw	Master 15 Priority¹⁾ This bit field contains the master priority for master connected to BCU request input 15 A lower number has a higher priority in the arbitration round than a higher one.
RESERVEDE, RESERVEDD, RESERVEDC, RESERVEDB, RESERVEDA, RESERVED9	[27:24], [23:20], [19:16], [15:12], [11:8], [7:4]	rw	Reserved Read as reset value or last written value; should be written with 0.

1) Including DMA transactions from DMA channels with low priority. Including Cerberus transactions with IOCONF.FPI_PRI0=0'.

On-Chip System Buses and Bus Bridges
SBCU_PRIOL
Arbiter Priority Register Low
(018_H)
Reset Value: 7654 3210_H


Field	Bits	Type	Description
DMAH	[3:0]	rw	Master 0 Priority¹⁾ This bit field contains the master priority for master connected to BCU request input 0 A lower number has a higher priority in the arbitration round than a higher one.
ETH	[11:8]	rw	Master 2 Priority This bit field contains the master priority for master connected to BCU request input 2 A lower number has a higher priority in the arbitration round than a higher one.
HSSL	[15:12]	rw	Master 3 Priority This bit field contains the master priority for master connected to BCU request input 3 A lower number has a higher priority in the arbitration round than a higher one.
DMAM	[23:20]	rw	Master 5 Priority²⁾ This bit field contains the master priority for master connected to BCU request input 5 A lower number has a higher priority in the arbitration round than a higher one.

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
CPU0	[31:28]	rw	Master 7 Priority This bit field contains the master priority for master connected to BCU request input 8 A lower number has a higher priority in the arbitration round than a higher one.
RESERVED6, RESERVED4, RESERVED1	[27:24], [19:16], [7:4]	rw	Reserved Read as reset value or last written value; should be written with 0.

1) Including DMA transactions from DMA channels with high priority. Including Cerberus transactions with IOCONF.FPI_PRI0='1'.

2) Including DMA transactions from DMA channels with medium priority.

2.5.5.3 SBCU Error Registers Descriptions

The capture of bus error conditions is enabled by setting SBCU_CON.DBG to 1. In case of a bus error, information about the condition will then be stored in the SBCU error capture registers. The SBCU error capture registers can then be examined by software to determine the cause of the FPI Bus error.

If enabled and an FPI Bus error occurs, the SBCU_ECON register holds the captured FPI Bus control information and an error count of the number of bus errors. The SBCU_EADD register stores the captured FPI Bus address. The SBCU_EDAT register stores the captured FPI Bus data.

If the capture of FPI Bus error conditions is disabled (SBCU_CON.DBG = 0), the SBCU error capture registers remain untouched.

Note: The SBCU error capture registers store only the parameters of the first error. In case of multiple bus errors, an error counter SBCU_ECON.ERRCNT shows the number of bus errors since the first error occurred. An application reset clears this bit field to zero, but the counter can be set to any value through software. This counter is prevented from overflowing, so a value of $2^{16} - 1$ indicates that at least this many errors have occurred, but there may have been more. After SBCU_ECON has been read, the SBCU_ECON, SBCU_EADD and SBCU_EDAT registers are re-enabled to trace FPI Bus error conditions.

On-Chip System Buses and Bus Bridges

SBCU_ECON
SBCU Error Control Capture Register(020_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OPC			TAG						RDN	WRN	SVM	ACK		ABT	
rwh			rwh						rwh	rwh	rwh	rwh		rwh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDY	TOU T	ERRCNT													
rwh	rwh	rwh													

Field	Bits	Type	Description
ERRCNT	[13:0]	rwh	FPI Bus Error Counter ERRCNT is incremented on every occurrence of an FPI Bus error. ERRCNT is reset to 0 after the SBCU_ECON register is read. ¹⁾
TOU	14	rwh	State of FPI Bus Time-Out Signal This bit indicates the state of the time-out signal at an FPI Bus error. 0 _B No time-out occurred 1 _B Time-out has occurred
RDY	15	rwh	State of FPI Bus Ready Signal This bit indicates the state of the ready signal at an FPI Bus error. 0 _B Wait state(s) have been inserted. Ready signal was active 1 _B Ready signal was inactive
ABT	16	rwh	State of FPI Bus Abort Signal This bit indicates the state of the abort signal at an FPI Bus error. 0 _B Master has aborted an FPI Bus transfer. Abort signal was active 1 _B Abort signal was inactive
ACK	[18:17]	rwh	State of FPI Bus Acknowledge Signals This bit field indicates the acknowledge code that has been output by the selected slave at an FPI Bus error. Coding see Table 2-10 .

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
SVM	19	rwh	State of FPI Bus Supervisor Mode Signal This bit indicates whether the FPI Bus error occurred in Supervisor Mode or in User Mode. 0 _B Transfer was initiated in User Mode 1 _B Transfer was initiated in Supervisor Mode
WRN	20	rwh	State of FPI Bus Write Signal This bit indicates whether the FPI Bus error occurred at a write cycle (see Table 2-14).
RDN	21	rwh	State of FPI Bus Read Signal This bit indicates whether the FPI Bus error occurred at a read cycle (see Table 2-14).
TAG	[27:22]	rwh	FPI Bus Master Tag Number Signals This bit field indicates the FPI Bus master TAG number (definitions see Table 2-15).
OPC	[31:28]	rwh	FPI Bus Operation Code Signals The FPI Bus operation codes are defined in Table 2-11 .

1) In the TC26x B-Step, aborted accesses to a 0 wait state SPB slave may also increment ERRCNT when the slave generates an error acknowledge.

Table 2-14 FPI Bus Read/Write Error Indication

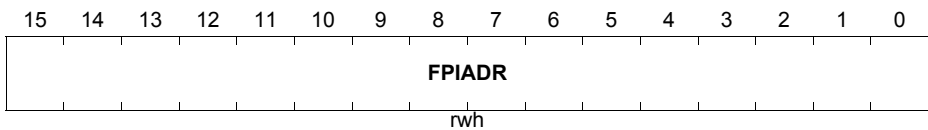
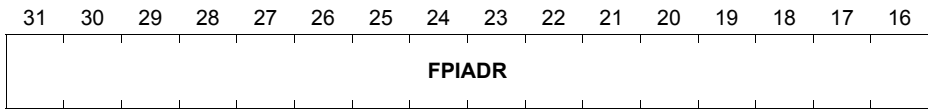
RD	WR	FPI Bus Cycle
0	0	FPI Bus error occurred at the read transfer of a read-modify-write transfer.
0	1	FPI Bus error occurred at a read cycle of a single transfer.
1	0	FPI Bus error occurred at a write cycle of a single transfer or at the write cycle of a read-modify-write transfer.
1	1	Does not occur.

On-Chip System Buses and Bus Bridges

SBCU_EADD

SBCU Error Address Capture Register (024_H)

Reset Value: 0000 0000_H

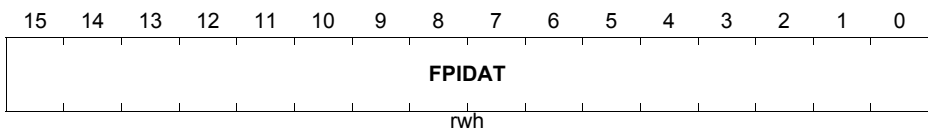
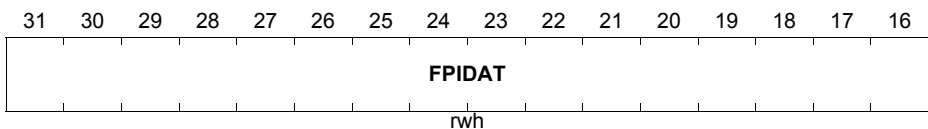


Field	Bits	Type	Description
FPIADR	[31:0]	rwh	Captured FPI Bus Address This bit field holds the 32-bit FPI Bus address that has been captured at an FPI Bus error. Note that if multiple bus errors occurred, only the address of the first bus error is captured.

SBCU_EDAT

SBCU Error Data Capture Register (028_H)

Reset Value: 0000 0000_H



On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
FPIDAT	[31:0]	rwh	Captured FPI Bus Data This bit field holds the 32-bit FPI Bus data that has been captured at an FPI Bus error. Note that if multiple bus errors occurred, only the data of the first bus error is captured.

On-Chip System Buses and Bus Bridges

2.5.5.4 SBCU OCDS Registers Descriptions

SBCU_DBCNTL

 SBCU Debug Control Register (030_H) Reset Value: 0000 7003_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ONB OS3	ONB OS2	ONB OS1	ONB OS0	0		ONA2		0		ONA1		0			ONG
r/w	r/w	r/w	r/w	r		r/w		r		r/w		r			r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CON COM 2	CON COM 1	CON COM 0				0				RA		0	OA	EO
r	r/w	r/w	r/w				r				w		r	r	r

Field	Bits	Type	Description
EO	0	r	Status of SBCU Debug Support Enable This bit is controlled by the Cerberus and enables the SBCU debug support. 0 _B SBCU debug support is disabled 1 _B SBCU debug support is enabled (default after reset)
OA	1	r	Status of SBCU Breakpoint Logic 0 _B The SBCU breakpoint logic is disarmed. Any further breakpoint activation is discarded 1 _B The SBCU breakpoint logic is armed The OA bit is set by writing a 1 to bit RA. When OA is set, registers SBCU_DBGNTT, SBCU_DBADRT, and SBCU_DBDAT are reset. Also SBCU_DBBOST is reset with the exception of the bit field FPIRST.
RA	4	w	Rearm SBCU Breakpoint Logic Writing a 1 to this bit rearms SBCU breakpoint logic and sets bit OA = 1. RA is always reads as 0.

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
CONCOM0	12	rw	<p>Grant and Address Trigger Relation</p> <p>0_B The grant phase trigger condition and the address trigger condition (see CONCOM1) are combined with a logical OR for further control</p> <p>1_B The grant phase trigger condition and the address trigger condition (see CONCOM1) are combined with a logical AND for further control (see Figure 2-15)</p>
CONCOM1	13	rw	<p>Address 1 and Address 2 Trigger Relation</p> <p>0_B Address 1 trigger condition and address 2 trigger condition are combined with a logical OR to the address trigger condition for further control</p> <p>1_B Address 1 trigger condition and address 2 trigger condition are combined with a logical AND to the address trigger condition for further control (see Figure 2-15)</p>
CONCOM2	14	rw	<p>Address and Signal Trigger Relation</p> <p>0_B Address trigger condition (see CONCOM1) and signal status trigger conditions are combined with a logical OR for further control</p> <p>1_B Address phase trigger condition (see CONCOM1) and the signal status trigger conditions are combined with a logical AND for further control (see Figure 2-15)</p>
ONG	16	rw	<p>Grant Trigger Enable</p> <p>0_B No grant debug event trigger is generated</p> <p>1_B The grant debug event trigger is enabled and generated according the settings of register SBCU_DBGRNT (see Figure 2-15)</p>
ONA1	[21:20]	rw	<p>Address 1 Trigger Control</p> <p>00_B No address 1 trigger is generated</p> <p>01_B An address 1 trigger event is generated if the FPI Bus address is equal to SBCU_DBADR1</p> <p>10_B An address 1 trigger event is generated if FPI Bus address is greater or equal to SBCU_DBADR1</p> <p>11_B same as 00_B (See also Figure 2-12).</p>

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
ONA2	[25:24]	rw	Address 2 Trigger Control 00 _B No address 2 trigger is generated 01 _B An address 2 trigger event is generated if the FPI Bus address is equal to SBCU_DBADR2 10 _B An address 2 trigger event is generated if FPI Bus address is less or equal to SBCU_DBADR2 11 _B same as 00 _B See also Figure 2-12 .
ONBOS0	28	rw	Opcode Signal Status Trigger Condition 0 _B A signal status trigger is generated for all FPI Bus op-codes except a “no operation” op-code 1 _B A signal status trigger is generated if the FPI Bus op-code matches the op-code as defined in DBBOS.OPC (see Figure 2-13)
ONBOS1	29	rw	Supervisor Mode Signal Trigger Condition 0 _B The signal status trigger generation for the FPI Bus Supervisor Mode signal is disabled 1 _B A signal status trigger is generated if the FPI Bus Supervisor Mode signal state is equal to the value of DBBOS.SVM (see Figure 2-13)
ONBOS2	30	rw	Write Signal Trigger Condition 0 _B The signal status trigger generation for the FPI Bus write signal is disabled 1 _B A signal status trigger is generated if the FPI Bus write signal state is equal to the value of DBBOS.WR (see Figure 2-13)
ONBOS3	31	rw	Read Signal Trigger Condition 0 _B The signal status trigger generation for the FPI Bus read signal is disabled 1 _B A signal status trigger is generated if the FPI Bus read signal state is equal to the value of DBBOS.RD (see Figure 2-13)

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
0	[3:2], [11:5], 15, [19:17], [23:22], [27:26]	r	Reserved Read as 0; should be written with 0.

SBCU_DBGRT
SBCU Debug Grant Mask Register (034_H) **Reset Value: 0000 FFFF_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA	ONE			ONE3			CPU	CPU	ONE	DMA	ONE	HSS	ETH	ONE	DMA
L	4						1	0	2	M	1	L		0	H
rw	rw			rw			rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
DMAH	0	rw	DMA High Priority Trigger Enable¹⁾ 0 _B FPI Bus transactions with DMA high priority request are enabled for grant trigger event generation 1 _B FPI Bus transactions with DMA / Cerberus high-priority request are disabled for grant trigger event generation
ETH	2	rw	Ethernet Trigger Enable 0 _B FPI Bus transactions requested by the ETH bus master are enabled for grant trigger event generation 1 _B FPI Bus transactions requested by the ETH bus master are disabled for grant trigger event generation

On-Chip System Buses and Bus Bridges

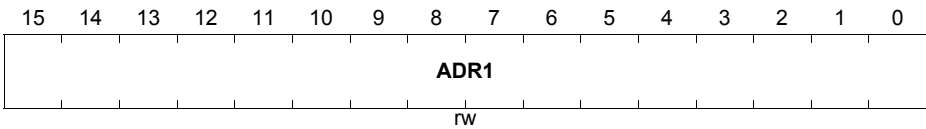
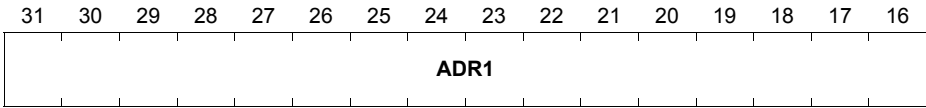
Field	Bits	Type	Description
HSSL	3	rw	HSSL Trigger Enable 0 _B FPI Bus transactions with HSSL as bus master are enabled for grant trigger event generation 1 _B FPI Bus transactions with HSSL as bus master are disabled for grant trigger event generation
DMAM	5	rw	DMA Grant Trigger Enable, Medium Priority²⁾ 0 _B FPI Bus transactions with medium-priority DMA channels as bus master are enabled for grant trigger event generation 1 _B FPI Bus transactions with medium-priority DMA channels as bus master are disabled for grant trigger event generation
CPU0	7	rw	CPU0 Grant Trigger Enable 0 _B FPI Bus transactions with CPU0 as bus master are enabled for grant trigger event generation 1 _B FPI Bus transactions with CPU as bus master are disabled for grant trigger event generation
CPU1	8	rw	CPU1 Grant Trigger Enable 0 _B FPI Bus transactions with CPU1 as bus master are enabled for grant trigger event generation 1 _B FPI Bus transactions with CPU1 as bus master are disabled for grant trigger event generation
DMAL	15	rw	DMA Grant Trigger Enable, Low Priority³⁾ 0 _B FPI Bus transactions with low-priority DMA channels as bus master are enabled for grant trigger event generation 1 _B FPI Bus transactions with low-priority DMA channels as bus master are disabled for grant trigger event generation
ONE4, ONE3, ONE2, ONE1, ONE0	14, [13:9], 6, 4, 1	rw	Reserved Read as 1 after reset; reading these bits will return the value last written.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

1) Including DMA transactions from DMA channels with high priority and from Cerberus with high priority.

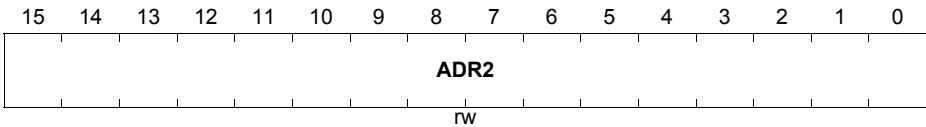
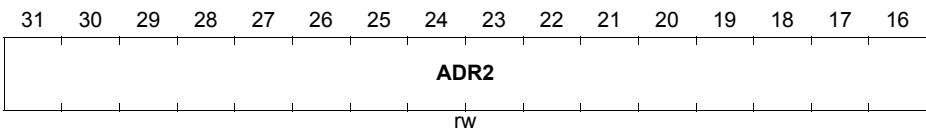
2) Including DMA transactions from DMA channels with medium priority.

3) Including DMA transactions from DMA channels with low priority and from Cerberus with low priority.

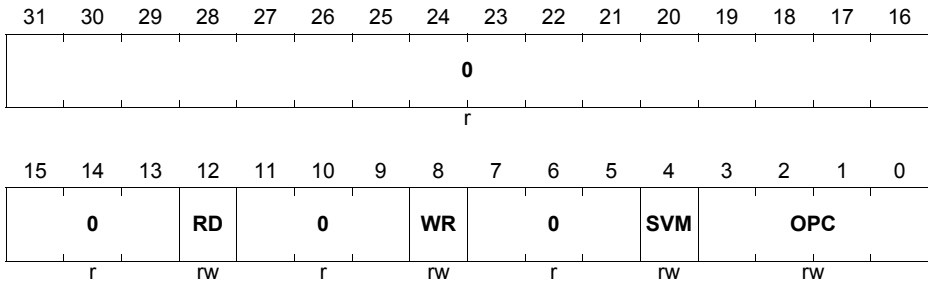
On-Chip System Buses and Bus Bridges

SBCU_DBADR1
SBCU Debug Address 1 Register
(038_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
ADR1	[31:0]	rw	Debug Trigger Address 1 This register contains the address for the address 1 trigger event generation.

SBCU_DBADR2
SBCU Debug Address 2 Register
(03C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
ADR2	[31:0]	rw	Debug Trigger Address 2 This register contains the address for the address 2 trigger event generation.

On-Chip System Buses and Bus Bridges
SBCU_DBBOS
SBCU Debug Bus Operation Signals Register
(040_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
OPC	[3:0]	rw	Opcode for Signal Status Debug Trigger This bit field determines the type (opcode) of an FPI Bus transaction for which a signal status debug trigger event is generated (if enabled by DBCNTL.ONBOS0 = 1). 0000 _B Trigger on single byte transfer selected 0001 _B Trigger on single half-word transfer selected 0010 _B Trigger on single word transfer selected 0100 _B Trigger on 2-word block transfer selected 0101 _B Trigger on 4-word block transfer selected 0110 _B Trigger on 8-word block transfer selected 1111 _B Trigger on no operation selected Other bit combinations are reserved.
SVM	4	rw	SVM Signal for Status Debug Trigger This bit determines the mode of an FPI Bus transaction for which a signal status debug trigger event is generated (if enabled by DBCNTL.ONBOS1 = 1). 0 _B Trigger on User Mode selected 1 _B Trigger on Supervisor Mode selected

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
WR	8	rw	Write Signal for Status Debug Trigger This bit determines the state of the WR signal of an FPI Bus transaction for which a signal status debug trigger event is generated (if enabled by DBCNTL.ONBOS2 = 1). 0_B Trigger on a single write transfer or write cycle of an atomic transfer selected 1_B No operation or read transaction selected
RD	12	rw	Write Signal for Status Debug Trigger This bit determines the state of the RD signal of an FPI Bus transaction for which a signal status debug trigger event is generated (if enabled by DBCNTL.ONBOS3 = 1). 0_B Trigger on a single read transfer or read cycle of an atomic transfer selected 1_B No operation or write transfer selected
0	[7:5], [11:9], [31:13]	r	Reserved Read as 0; should be written with 0.

SBCU_DBGNTT
SBCU Debug Trapped Master Register
(044_H)
Reset Value: 0000 FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ONE4								DMACHNR							
rw								rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA	ONE3						CPU	CPU	ONE	DMA	ONE	HSS	ETH	ONE	DMA
L							1	0	2	M	1	L	ETH	0	H
rw	rw						rw	rw	rw	rw	rw	rw	rw	rw	rw

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
DMAH	0	rw	High-Priority DMA FPI Bus Master Status¹⁾ This bit indicates whether the DMA with a medium priority request was FPI Bus master when the break trigger event occurred. 0 _B The medium-priority DMA was the FPI bus master. 1 _B The medium-priority DMA was not the FPI Bus master.
ETH	2	rw	Ethernet FPI Bus Master Status This bit indicates whether the Ethernet was FPI Bus master when the break trigger event occurred. 0 _B The Ethernet was the FPI bus master. 1 _B The Ethernet was not the FPI Bus master.
HSSL	3	rw	HSSL FPI Bus Master Status This bit indicates whether the HSSL was FPI Bus master when the break trigger event occurred. 0 _B The Ethernet was the FPI bus master. 1 _B The Ethernet was not the FPI Bus master.
DMAM	5	rw	Medium-Priority DMA FPI Bus Master Status²⁾ This bit indicates whether the DMA with a medium priority request was FPI Bus master when the break trigger event occurred. 0 _B The medium-priority DMA was the FPI bus master. 1 _B The medium-priority DMA was not the FPI Bus master.
CPU0	7	rw	CPU0 FPI Bus Master Status This bit indicates whether the CPU0 was FPI Bus master when the break trigger event occurred. 0 _B The CPU0 was the FPI Bus master. 1 _B The CPU0 was not the FPI Bus master.
CPU1	8	rw	CPU1 FPI Bus Master Status This bit indicates whether the CPU1 was FPI Bus master when the break trigger event occurred. 0 _B The CPU1 was the FPI Bus master. 1 _B The CPU1 was not the FPI Bus master.

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
DMAL	15	rw	Low-Priority DMA FPI Bus Master Status³⁾ This bit indicates whether the DMA with a low-priority request was the FPI Bus master when the break trigger event occurred. 0 _B The low-priority DMA was the FPI Bus master. 1 _B The low-priority DMA was not the FPI Bus master.
DMACHNR	[23:16]	rw	DMA-FPI Channel Grant Status The encoding is directly dedicated to the dma channel number
ONE4, ONE3, ONE2, ONE1, ONE0	[31:24], [14:9], 6, 4, 1	rw	Reserved Read as 1; shall be written with 0.

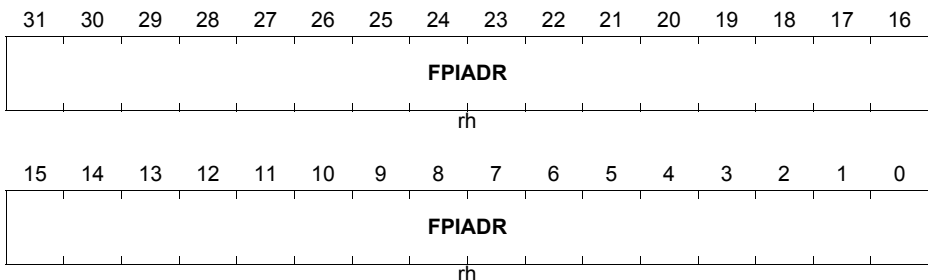
- 1) Including DMA transactions from DMA channels with high priority and from Cerberus with high priority.
- 2) Including DMA transactions from DMA channels with medium priority.
- 3) Including DMA transactions from DMA channels with low priority, MLI and from Cerberus with low priority.

SBCU_DBADRT

SBCU Debug Trapped Address Register

(048_H)

Reset Value: 0000 0000_H



On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
FPIADR	[31:0]	rh	FPI Bus Address Status This register contains the FPI Bus address that was captured when the OCDS break trigger event occurred.

SBCU_DBBOST
SBCU Debug Trapped Bus Operation Signals Register
(04C_H)
Reset Value: 0000 3180_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0										FPITAG						
rh										rh						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
ENDI NIT	FPIT OUT	FPIA BOR T	FPIR D	FPIO PS	FPIRST	FPI WR	FPIR DY	FPIACK	FPIS VM	FPIOPC						
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh					

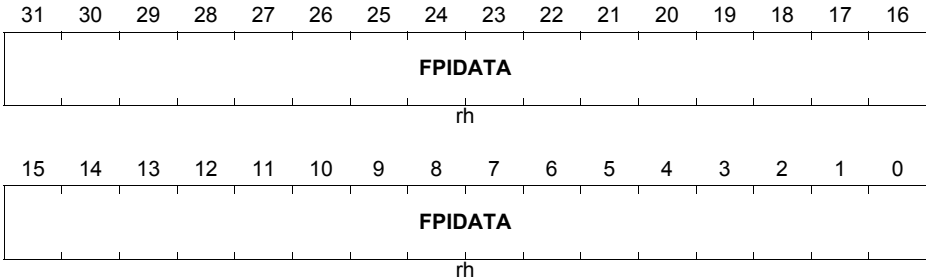
Field	Bits	Type	Description
FPIOPC	[3:0]	rh	FPI Bus Opcode Status This bit field indicates the type (opcode) of the FPI Bus transaction captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0000 _B Single byte transfer 0001 _B Single half-word transfer 0010 _B Single word transfer 0100 _B 2-word block transfer 0101 _B 4-word block transfer 0110 _B 8-word block transfer 1111 _B No operation Other bit combinations are reserved.

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
FPI SVM	4	rh	FPI Bus Supervisor Mode Status This bit indicates the state of the Supervisor Mode signal captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 _B User mode 1 _B Supervisor mode
FPI ACK	[6:5]	rh	FPI Bus Acknowledge Status This bit field indicates the acknowledge signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 00 _B No special case 01 _B Error 10 _B Reserved 11 _B Retry, slave did not respond
FPI RDY	7	rh	FPI Bus Ready Status This bit indicates the ready signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 _B Last cycle of transfer 1 _B Not last cycle of transfer
FPI WR	8	rh	FPI Bus Write Indication Status This bit indicates the write signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 _B Single write transfer or write cycle of an atomic transfer 1 _B No operation or read transfer
FPI RST	[10:9]	rh	FPI Bus Reset Status This bit field indicates the reset signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 00 _B Reset of all FPI Bus components 11 _B No reset Others Reserved

On-Chip System Buses and Bus Bridges

Field	Bits	Type	Description
FPIOPS	11	rh	FPI Bus OCDS Suspend Status This bit indicates the OCDS suspend signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 _B No OCDS suspend request is pending 1 _B An OCDS suspend request is pending
FPIRD	12	rh	FPI Bus Read Indication Status This bit indicates the read signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 _B Single read transfer or read cycle of an atomic transfer 1 _B No operation or write transfer
FPIABORT	13	rh	FPI Bus Abort Status This bit indicates the abort signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 _B A transfer that has already started was aborted 1 _B Normal operation
FPIOUT	14	rh	FPI Bus Time-out Status This bit indicates the time-out signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 _B Normal operation 1 _B A time-out event was generated
ENDINIT	15	rh	FPI Bus Endinit Status This bit indicates the ENDINIT signal status captured from the FPI Bus signal lines when the BCU break trigger event occurred. 0 _B Normal operation 1 _B System was in ENDINIT state
FPITAG	[21:16]	rh	FPI Bus Master TAG Status This bit field indicates the master TAG captured from the FPI Bus signal lines when the BCU break trigger event occurred (see Table 2-15). The master TAG identifies the master of the transfer which generated BCU break trigger event.
0	[31:22]	rh	Reserved Read as 0; should be written with 0.

On-Chip System Buses and Bus Bridges
SBCU_DBDAT
SBCU Debug Data Status Register
(050_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
FPIDATA	[31:0]	rh	FPI Bus Data Status This register contains the FPI Bus data that was captured when the OCDS break trigger event occurred.

2.6 On Chip Bus Master TAG Assignments

Each master interface on the System Peripheral Bus and on the SRI Bus is assigned to a 6-bit identification number, the master TAG number (see [Table 2-15](#)). This makes it possible for software debug and MCDS purposes to distinguish which master has performed the current transaction (see [“XBAR_ERRx \(x = 0-1\)” on Page 2-54^{1\)}](#) and [“SBCU_DBADRT” on Page 2-111](#)).

Table 2-15 On Chip Bus Master TAG Assignments

TAG-Number	Module	Location	Description
000000 _B	CPU0	SRI/SPB	DMI.NonSafe TAG ID
000001 _B	CPU0	SRI/SPB	DMI.Safe TAG ID
000010 _B	CPU1	SRI/SPB	DMI.NonSafe TAG ID
000011 _B	CPU1	SRI/SPB	DMI.Safe TAG ID
000100 _B	-	-	Reserved

1) Pls. note that the Transaction ID bit field in the register [“XBAR_ERRx \(x = 0-1\)” on Page 2-54](#) includes the TAG ID on the 6 LSB (TRID[5:0]).

On-Chip System Buses and Bus Bridges
Table 2-15 On Chip Bus Master TAG Assignments

TAG-Number	Module	Location	Description
000101 _B	-	-	Reserved
000110 _B	DMA	SRI/SPB	DMA Resource Partition 0
000111 _B	DMA	SRI/SPB	DMA Resource Partition 1
001000 _B	DMA	SRI/SPB	DMA Resource Partition 2
001001 _B	DMA	SRI/SPB	DMA Resource Partition 3
001010 _B	DMA	SRI/SPB	Cerberus
001011 _B	HSSL	SRI/SPB	High Speed Serial Link
001100 _B	ETH	SPB	Ethernet
001101 _B	-	-	Reserved
001110 _B	-	-	Reserved
001111 _B	-	-	Reserved
010000 _B	CPU0	SRI	PMI
010001 _B	CPU1	SRI	PMI
011100 _B	IOC32	BBB	Cerberus on Back Bone Bus (ED)
011101 _B	EMEM	BBB	EMEM Master on Back Bone Bus (ED)
011110 _B	CIF	BBB	CIF Master on Back Bone Bus (ED)
Others	-	-	Reserved

2.7 On Chip Bus Access Times

The table describes the CPU access times in CPU clock cycles for the TC26x B-Step. The access times are described as maximum CPU stall cycles where e.g. an data access to the local DSPR results in zero stall cycles. Pls. note that the CPU does not always immediately stall after the start of a data read from another SPR due to instruction pipelining effects. This means that the average number will be below the here shown numbers[RF00105].

Note: All values are preliminary

Table 2-16 CPU access latency in CPU clock cycles for TC26x B-Step

CPU Access Mode	CPU clock cycles
Data read access to own DSPR	0
Data write access to own DSPR	0
Data read access to own or other PSPR	5

On-Chip System Buses and Bus Bridges
Table 2-16 CPU access latency in CPU clock cycles for TC26x B-Step (cont'd)

CPU Access Mode	CPU clock cycles
Data write access to own or other PSPR	0
Data read access to other DSPR	5
Data write access to other DSPR	0
Instruction fetch from own PSPR	0
Instruction fetch from other PSPR (critical word)	5
Instruction fetch from other PSPR (any remaining words)	0
Instruction fetch from other DSPR (critical word)	5
Instruction fetch from other DSPR (any remaining words)	0
Initial Pflash Access (critical word)	5 + configured PFlash Wait States ¹⁾
Initial Pflash Access (remaining words)	0
PMU PFlash Buffer Hit (critical word)	4
PMU PFlash Buffer Hit (remaining words)	0
Initial Dflash Access	5 + configured DFlash Wait States ²⁾
TC1.6E Data read from System Peripheral Bus (SPB)	4 ($f_{\text{CPU}}=f_{\text{SPB}}$) 7 ($f_{\text{CPU}}=2*f_{\text{SPB}}$)
TC1.E Data write to System Peripheral Bus (SPB)	0

1) FCON.WSPFLASH + FCON.WSECPF (see PMU chapter for the detailed description of these parameters).

2) FCON.WSDFLASH + FCON.WSECDF (see PMU chapter for the detailed description of these parameters).

3 Memory Maps

This chapter gives an overview of the TC26x B-Step memory map, and describes the address locations and access possibilities for the units, memories, and reserved areas as “seen” from the two different on-chip buses’ point of view.

The TC26x B-Step has the following CPU related memories:

- Program Memory Unit (PMU0) with
 - 2.5 Mbyte of Program Flash Memory (PFLASH)
 - Data Flash Memory (DF_EEPROM)
 - User Configuration Blocks (DF_UCB)
 - 32 Kbyte of Boot ROM (BROM)
- CPU0:
 - 16 Kbyte of Program Scratch-Pad SRAM (PSPR)¹⁾
 - 72 Kbyte of Data Scratch-Pad SRAM (DSPR)¹⁾
 - 8 Kbyte of Program Cache (PCache)
- CPU1:
 - 32 Kbyte of Program Scratch-Pad SRAM (PSPR)¹⁾
 - 120 Kbyte of Data Scratch-Pad SRAM (DSPR)¹⁾
 - 16 Kbyte of Program Cache (PCache)
 - 8 Kbyte of Data Cache (DCACHE)
- LMU:
 - 0 Kbyte SRAM (LMURAM)¹⁾

Furthermore, the TC26x B-Step has two on-chip buses:

- System Peripheral Bus (SPB)
- Shared Resource Interconnect (SRI)

1) Before used by the application, the memory has to be initialized (see chapter ‘Memory Test Unit’, MTU)

3.1 How to Read the Address Maps

The bus-specific address maps describe how the different bus master devices react on accesses to on-chip memories and modules, and which address ranges are valid or invalid for the corresponding buses.

The detailed address mapping of e.g. control registers, sram blocks or flash banks/sectors within a module is described in the related module chapter.

The SPB Bus address map shows the system addresses from the point of view of the SPB master agents¹⁾.

The SRI address map shows the system addresses from the point of view of the SRI master agents¹⁾.

The SFI is an uni-directional bridge for access from SRI to SPB and therefore not mentioned here as SRI master in the Address Map. The SFI is fully transparent and does not include an address translation mechanism.

Note: In addition to the here described system address map, each TriCore has a TriCore IP internal access to its PSPR via C000_0000 and an internal access to its DSPR via D000_0000. This additional/private view to the local scratch pad srams is described in the CPU chapter.

1) Available SRI / SPB master agents are described in the On Chip Bus System chapter

Memory Maps

Table 3-1 defines the acronyms and other terms that are used in the address maps (**Table 3-2** and **Table 3-3**).

Table 3-1 Definition of Acronyms and Terms

Term	Description
...BE	Means "Bus error" generation.
SPBBE	A bus access is terminated with a bus error on the SPB.
SRIBE	A bus access is terminated with a bus error on the SRI.
access	A bus access is allowed and is executed.

3.2 Contents of the Segments

This section summarizes the contents of the segments.

The CPU attributes (cached / non-cached) of these segments can be configured for each CPU's data and program side individually (see CPU chapter: Physical Memory Attribute Registers, PMAx).

Segments 0-5

These memory segments are reserved in the TC26x B-Step.

Segments 6-7

These memory segments allow access to the CPU's Program and Data Scratch Pad SRAM (PSPR, DSPR), Program and Data Cache SRAMs (PCache, DCache) as well as TAG SRAMs related to Program and Data Cache (PTAG SRAM¹) and DTAG SRAM¹). Where DCache is supported, DCache and DTAG SRAM¹) can be only accessed if the Data Cache is disabled.

PCache and PTAG SRAMs¹) can be only accessed if the related Program Cache is disabled²).

CPUx default attributes for these segments: non-cached.

Segment 8

This memory segment allows access to PFlash and BROM.

CPUx default attributes for this segment: cached.

Segment 9

This memory segment allows access to LMU SRAM (if implemented) and to the EMEM (Emulation Device only).

CPUx default attributes for this segment: cached.

Segment 10

This memory segment allows access to PFlash, DFlash and BROM.

CPUx default attributes for this segment: non-cached.

1) TAG SRAMs are not meant to be used as general SRAMs and can be accessed only with single data access and only with 64 bit aligned address.

2) Mapping of Cache and TAG SRAMs is controlled via the MTU register MTU_MEMMAP.

Segment 11

This memory segment allows access to LMU SRAM (if implemented) and to the EMEM (Emulation Device only).

CPUx default attributes for this segment: non-cached.

Segment 12

This memory segment is reserved in the TC26x B-Step.

Segment 13

This memory segment is reserved in the TC26x B-Step.

Segment 14

This memory segment is reserved in the TC26x B-Step.

Segment 15

This memory segment allows accesses to all SFRs, CSFRs.

Access from DMA Move Engine, Cerberus to the lower 128 MB of this segment are processed by the DMA FPI master interface on the SPB Bus, to the upper 128 MB of this segment by the DMA SRI master interface on the SRI Bus.

Data access from TC1.6 CPUs to the lower 128 MB of this segment are processed by the CPU.DMI FPI master interface on the SPB Bus, to the upper 128 MB of this segment by the CPU.DMI SRI master interface on the SRI Bus.

3.3 Address Map of the On Chip Bus System

This chapter describes the system address map as it is seen from the SRI and SPB bus masters (bus master agents are described in the chapter On Chip Bus System).

All bus master agents can address identical peripherals and memories at identical address. The system address map is visible and valid for all CPUs which means that all peripherals and resources are accessible from all TriCore CPUs and other on chip bus master agents.

Parallel access by more than one bus master agents to one slave agent are executed sequentially. Additionally the SRI and SPB bus do support atomic Read Modify Write sequences from the CPUs. Hardware semaphores for temporary exclusive bus access from one master to a slave are not implemented

3.3.1 Segments 0 to 14

Table 3-2 shows the address map of segments 0 to 14.

Table 3-2 On Chip Bus Address Map of Segment 0 to 14

Segment	Address Range	Size	Description	Access Type	
				Read ¹⁾	Write ²⁾
0-5	0000 0000 _H - 0000 0007 _H	8 byte	Reserved (virtual address space)	SRIBE	SRIBE
	0000 0008 _H - 5FFF FFFF _H	-		SRIBE	SRIBE
6	6000 0000 _H - 6001 DFFF _H	120 Kbyte	CPU1 Data Scratch-Pad SRAM (CPU1.DSPR)	access	access
	6001 E000 _H - 6001 FFFF _H	8 Kbyte	CPU1.Data Cache SRAM (CPU1.DCACHE)	access ³⁾ / SRIBE	access ³⁾ / SRIBE
	6002 0000 _H - 600B FFFF _H	-	Reserved	SRIBE	SRIBE
	600C 0000 _H - 600C 0BFF _H	-	CPU1 Data Cache TAG SRAM ⁴⁾ (CPU1.DTAG)	access ³⁾ / SRIBE	access ³⁾ / SRIBE
	600C 0C00 _H - 600F FFFF _H	-	Reserved	SRIBE	SRIBE
	6010 0000 _H - 6010 7FFF _H	32 Kbyte	CPU1 Program Scratch-Pad SRAM (CPU1.PSPR)	access	access
	6010 8000 _H - 6010 BFFF _H	16 Kbyte	CPU1.Program Cache SRAM (CPU1.PCache)	access ³⁾ / SRIBE	access ³⁾ / SRIBE

Memory Maps
Table 3-2 On Chip Bus Address Map of Segment 0 to 14 (cont'd)

Segment	Address Range	Size	Description	Access Type	
				Read ¹⁾	Write ²⁾
	6010 C000 _H - 601B FFFF _H	-	Reserved	SRIBE	SRIBE
	601C 0000 _H - 601C 17FF _H	-	CPU1 Program Cache TAG SRAM ⁴⁾ (CPU1.PTAG)	access ³⁾ / SRIBE	access ³⁾ / SRIBE
	601C 1800 _H - 6FFF FFFF _H	-	Reserved	SRIBE	SRIBE
7	7000 0000 _H - 7001 1FFF _H	72 Kbyte	CPU0 Data Scratch-Pad SRAM (CPU0.DSPR)	access	access
	7001 2000 _H - 700F FFFF _H	-	Reserved	SRIBE	SRIBE
	7010 0000 _H - 7010 3FFF _H	16 Kbyte	CPU0 Program Scratch- Pad SRAM (CPU0.PSPR)	access	access
	7010 4000 _H - 7010 5FFF _H	8 Kbyte	CPU0.Program Cache SRAM (CPU0.PCache)	access ³⁾ / SRIBE	access ³⁾ / SRIBE
	7010 6000 _H - 701B FFFF _H	-	Reserved	SRIBE	SRIBE
	701C 0000 _H - 701C 0BFF _H	-	CPU0 Program Cache TAG SRAM ⁴⁾ (CPU0.PTAG)	access ³⁾ / SRIBE	access ³⁾ / SRIBE
	701C 0C00 _H - 7FFF FFFF _H	-	Reserved	SRIBE	SRIBE
8	8000 0000 _H - 800F FFFF _H	1 Mbyte	Program Flash 0 (PF0)	access	access ²⁾
	8010 0000 _H - 8027 FFFF _H	1.5 Mbyte	Program Flash 1 (PF1)	access	access ²⁾
	8028 0000 _H - 8FE6 FFFF _H	-	Reserved	SRIBE	SRIBE
	8FE7 0000 _H - 8FE7 7FFF _H	32 Kbyte	Online Data Acquisition (OLDA)	SRIBE	access ⁵⁾ / SRIBE
	8FE7 8000 _H - 8FFF 7FFF _H	-	Reserved	SRIBE	SRIBE

Memory Maps
Table 3-2 On Chip Bus Address Map of Segment 0 to 14 (cont'd)

Segment	Address Range	Size	Description	Access Type	
				Read ¹⁾	Write ²⁾
	8FFF 8000 _H - 8FFF FFFF _H	32 Kbyte	Boot ROM (BROM)	access	SRIBE
9	9000 0000 _H - 9EFF FFFF _H	-	Reserved	SRIBE	SRIBE
	9F00 0000 _H - 9F07 FFFF _H	512 Kbyte	Reserved for TC26x B- Step Emulation Device Memory (EMEM)	SRIBE	SRIBE ⁶⁾
	9F08 0000 _H - 9F0F FFFF _H	-	Reserved	SRIBE	SRIBE
	9F10 0000 _H - 9F10 3FFF _H	16 Kbyte	Reserved for TC26x B- Step Emulation Device Memory (EMEM)	SRIBE	SRIBE ⁶⁾
	9F10 4000 _H - 9FFF FFFF _H	-	Reserved	SRIBE	SRIBE
10	A000 0000 _H - A00F FFFF _H	1 Mbyte	Program Flash 0 (PF0)	access	access ²⁾
	A010 0000 _H - A027 FFFF	1.5 Mbyte	Program Flash 1 (PF1)	access	access ²⁾
	A028 0000 _H - AEFF FFFF _H	-	Reserved	SRIBE	SRIBE
	AF00 0000 _H - AF0F FFFF _H	1 Mbyte	Data Flash 0 (DF0 Address Range)	access	access ²⁾⁷⁾
	AF10 0000 _H - AF10 3FFF _H	16 Kbyte	Data Flash 0 (DF0 Address Range)	access	access ²⁾⁷⁾
	AF10 4000 _H - AFE6 FFFF _H	-	Reserved	SRIBE	SRIBE
	AFE7 0000 _H - AFE7 7FFF _H	32 Kbyte	Online Data Acquisition (OLDA)	SRIBE	access ⁵⁾ / SRIBE
	AFE7 8000 _H - AFF7 FFFF _H	-	Reserved	SRIBE	SRIBE
	AFFF 8000 _H - AFFF FFFF _H	32 Kbyte	Boot ROM (BROM)	access	SRIBE
11	B000 0000 _H - BDFF FFFF _H	-	Reserved	SRIBE	SRIBE

Memory Maps
Table 3-2 On Chip Bus Address Map of Segment 0 to 14 (cont'd)

Segment	Address Range	Size	Description	Access Type	
				Read ¹⁾	Write ²⁾
	BE00 0000 _H - BE07 FFFF _H	512 Kbyte	Reserved for FFT Accelerator (ADAS Device only)	SRIBE	SRIBE ⁶⁾
	BE08 0000 _H - BE0F FFFF _H	-	Reserved	SRIBE	SRIBE
	BE10 0000 _H - BE17 FFFF _H	512 Kbyte	Reserved for FFT Accelerator (ADAS Device only)	SRIBE	SRIBE ⁶⁾
	BE18 0000 _H - BEFF FFFF _H	-	Reserved	SRIBE	SRIBE
	BF00 0000 _H - BF07 FFFF _H	512 Kbyte	Reserved for TC26x B-Step Emulation Device Memory (EMEM)	SRIBE	SRIBE ⁶⁾
	BF08 0000 _H - BF0F FFFF _H	-	Reserved	SRIBE	SRIBE
	BF10 0000 _H - BF10 3FFF _H	16 Kbyte	Reserved for TC26x B-Step Emulation Device Memory (EMEM)	SRIBE	SRIBE ⁶⁾
	BF10 4000 _H - BFFF FFFF _H	-	Reserved	SRIBE	SRIBE
12	C000 0000 _H - CFFF FFFF _H	-	Reserved ⁸⁾	SRIBE	SRIBE
13	D000 0000 _H - DFFF FFFF _H	-	Reserved ⁸⁾	SRIBE	SRIBE
14	E000 0000 _H - EFFF FFFF _H	-	Reserved	SRIBE	SRIBE
15	F000 0000 _H - FFFF FFFF _H	256 Mbyte	see Table 3-3		

1) A read transaction through the SRI to FPI bridge that is terminated with Bus Error will result in Bus Errors on SRI and FPI (valid for transactions from FPI to SRI and SRI to FPI).

2) Write access to Flash resources are handled by the PMU module (Flash command sequence, see PMU chapter for details).

3) PCache/DCache SRAMs (and the corresponding TAG SRAMs) can be only accessed when mapped into the address space (PCache / DCache disabled. See MTU chapter, register MTU_MEMMAP for details).

Memory Maps

- 4) TAG SRAMs are not meant to be used as general SRAMs and can be accessed only with single data access and only with 64 bit aligned address. Mapping of TAG SRAMs in the address map can be used as additional option for memory testing.
- 5) Online Data Acquisition address space can be disabled/enabled via LMU control register bit LMU_MEMCON.OLDAEN. TC1.6P access to OLDA address space via segment 8 (cached) results in SRIBE independent of the LMU_MEMCON.OLDAEN bit setting.
- 6) This address range is mapped to the LMU module. A read/write from/to this address range will result in an SRI Bus Error initiated by the LMU module,
- 7) DF0 is the address where all Data Flash blocks are mapped to. The details about the Data Flash block sizes, segmentation and exact mapping are described in the chapter PMU.
- 8) See also chapter 'CPU, 'Local and Global Addressing' for CPU local views to segment 'C' and segment 'D'.

3.3.2 Segment 15

Table 3-3 shows the address map of segment 'F' as seen from the SRI and SPB bus masters (bus master agents are described in the chapter On Chip Bus System).

Table 3-1 gives an overview about the address mapping of the module address ranges:

- which modules are mapped into the first 16 KB of segment 'F' and can be accessed by the TC1.6E/P with absolute addressing modes (left side)
- examples of covering modules with relative addressing mode (base address +/- 32 KB, in the middle)

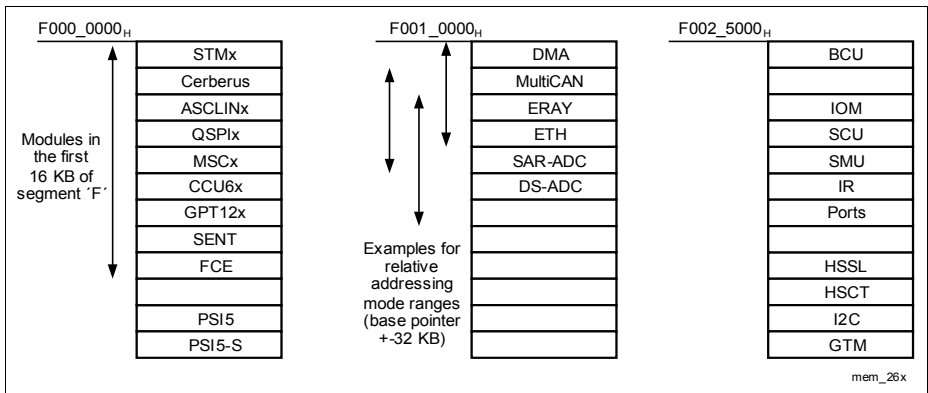


Figure 3-1 Segment F Structure

Please note that **Table 3-3** describes the mapping of modules to segment F. The details of the module address ranges can be found in the module chapters register overview.

Table 3-3 On Chip Bus Address Map of Segment 15

Unit	Address Range	Size	Access Type	
			Read	Write
System Timer 0 (STM0)	F000 0000 _H - F000 00FF _H	256 byte	access	access
System Timer 1 (STM1)	F000 0100 _H - F000 01FF _H	256 byte	access	access
Reserved	F000 0200 _H - F000 03FF _H	–	SPBBE	SPBBE
On-Chip Debug Support (Cerberus)	F000 0400 _H - F000 05FF _H	2x256 byte	access	access

Table 3-3 On Chip Bus Address Map of Segment 15 (cont'd)

Unit	Address Range	Size	Access Type	
			Read	Write
ASCLIN0 (ASCLIN0)	F000 0600 _H - F000 06FF _H	256 byte	access	access
ASCLIN1 (ASCLIN1)	F000 0700 _H - F000 07FF _H	256 byte	access	access
ASCLIN2 (ASCLIN2)	F000 0800 _H - F000 08FF _H	256 byte	access	access
ASCLIN3 (ASCLIN3)	F000 0900 _H - F000 09FF _H	256 byte	access	access
Reserved	F000 0A00 _H - F000 1BFF _H	–	SPBBE	SPBBE
QUEUED SPI 0 (QSPI0)	F000 1C00 _H - F000 1CFF _H	256 byte	access	access
QUEUED SPI 1 (QSPI1)	F000 1D00 _H - F000 1DFF _H	256 byte	access	access
QUEUED SPI 2 (QSPI2)	F000 1E00 _H - F000 1EFF _H	256 byte	access	access
QUEUED SPI 3 (QSPI3)	F000 1F00 _H - F000 1FFF _H	256 byte	access	access
Reserved	F000 2000 _H - F000 25FF _H	–	SPBBE	SPBBE
MicroSecond Bus Controller 0 (MSC0)	F000 2600 _H - F000 26FF _H	256 byte	access	access
MicroSecond Bus Controller 1 (MSC1)	F000 2700 _H - F000 27FF _H	256 byte	access	access
Reserved	F000 2800 _H - F000 29FF _H	–	SPBBE	SPBBE
Capture/Compare Unit 6 0 (CCU60)	F000 2A00 _H - F000 2AFF _H	256 byte	access	access
Capture/Compare Unit 6 1 (CCU61)	F000 2B00 _H - F000 2BFF _H	256 byte	access	access
Reserved	F000 2C00 _H - F000 2DFF _H	–	SPBBE	SPBBE

Table 3-3 On Chip Bus Address Map of Segment 15 (cont'd)

Unit	Address Range	Size	Access Type	
			Read	Write
General Purpose Timer 12 0 (GPT120)	F000 2E00 _H - F000 2EFF _H	256 byte	access	access
Reserved	F000 2F00 _H - F000 2FFF _H	–	SPBBE	SPBBE
SENT Module (SENT)	F000 3000 _H - F000 3AFF _H	11x256 byte	access	access
Reserved	F000 3B00 _H - F000 3EFF _H	–	SPBBE	SPBBE
Flexible CRC Engine (FCE)	F000 3F00 _H - F000 3FFF _H	256 byte	access	access
Reserved	F000 4000 _H - F000 4FFF _H	–	SPBBE	SPBBE
PSI5 (PSI5)	F000 5000 _H - F000 6FFF _H	8 KByte	access	access
PSI5-S (PSI5-S)	F000 7000 _H - F000 7FFF _H	4 KByte	access	access
Reserved	F000 8000 _H - F000 FFFF _H	–	SPBBE	SPBBE
Direct Memory Access Controller (DMA)	F001 0000 _H - F001 3FFF _H	16 KByte	access	access
Reserved	F001 4000 _H - F001 7FFF _H	–	SPBBE	SPBBE
MultiCAN Controller (CAN)	F001 8000 _H - F001 BFFF _H	16 Kbyte	access	access
FlexRay™ Protocol Controller (E-Ray)	F001 C000 _H - F001 CFFF _H	4 Kbyte	access	access
Ethernet Controller System Control Register (ETH)	F001 D000 _H - F001 D0FF _H	256 byte	access	access
Reserved	F001 D100 _H - F001 DFFF _H	–	SPBBE	SPBBE
Ethernet Controller (ETH)	F001 E000 _H - F001 FFFF _H	8 KByte	access	access

Table 3-3 On Chip Bus Address Map of Segment 15 (cont'd)

Unit	Address Range	Size	Access Type	
			Read	Write
Analog-to-Digital Converter (VADC)	F002 0000 _H - F002 3FFF _H	16 KByte	access	access
Delta Sigma Digital Analog-to-Digital Converter (DSADC)	F002 4000 _H - F002 4FFF _H	4 Kbyte	access	access
Reserved	F002 5000 _H - F002 FFFF _H	–	SPBBE	SPBBE
System Peripheral Bus Control Unit (BCU)	F003 0000 _H - F003 00FF _H	256 byte	access	access
Reserved	F003 0100 _H - F003 4FFF _H	–	SPBBE	SPBBE
I/O Monitor (IOM)	F003 5000 _H - F003 51FF _H	2x256 byte	access	access
Reserved	F003 5200 _H - F003 5FFF _H	–	SPBBE	SPBBE
System Control Unit (SCU)	F003 6000 _H - F003 63FF _H	1 Kbyte	access	access
Reserved	F003 6400 _H - F003 67FF _H	–	SPBBE	SPBBE
Safety Management Unit (SMU)	F003 6800 _H - F003 6FFF _H	2 Kbyte	access	access
Interrupt Router (IR)	F003 7000 _H - F003 7FFF _H	4 Kbyte	access	access
Interrupt Router (IR) SRC Registers	F003 8000 _H - F003 9FFF _H	8 Kbyte	access	access
Port 00	F003 A000 _H - F003 A0FF _H	256 byte	access	access
Reserved	F003 A100 _H - F003 A1FF _H	256 byte	SPBBE	SPBBE
Port 02	F003 A200 _H - F003 A2FF _H	256 byte	access	access
Reserved	F003 A300 _H - F003 AFFF _H	–	SPBBE	SPBBE

Table 3-3 On Chip Bus Address Map of Segment 15 (cont'd)

Unit	Address Range	Size	Access Type	
			Read	Write
Port 10	F003 B000 _H - F003 B0FF _H	256 byte	access	access
Port 11	F003 B100 _H - F003 B1FF _H	256 byte	access	access
Reserved	F003 B200 _H - F003 B2FF _H	–	SPBBE	SPBBE
Port 13	F003 B300 _H - F003 B3FF _H	256 byte	access	access
Port 14	F003 B400 _H - F003 B4FF _H	256 byte	access	access
Port 15	F003 B500 _H - F003 B5FF _H	256 byte	access	access
Reserved	F003 B600 _H - F003 BFFF _H	–	SPBBE	SPBBE
Port 20	F003 C000 _H - F003 C0FF _H	256 byte	access	access
Port 21	F003 C100 _H - F003 C1FF _H	256 byte	access	access
Port 22	F003 C200 _H - F003 C2FF _H	256 byte	access	access
Port 23	F003 C300 _H - F003 C3FF _H	256 byte	access	access
Reserved	F003 C400 _H - F003 D1FF _H	–	SPBBE	SPBBE
Port 32	F003 D200 _H - F003 D2FF _H	256 byte	access	access
Port 33	F003 D300 _H - F003 D3FF _H	256 byte	access	access
Reserved	F003 D400 _H - F003 DFFF _H	–	SPBBE	SPBBE
Port 40	F003 E000 _H - F003 E0FF _H	256 byte	access	access

Memory Maps
Table 3-3 On Chip Bus Address Map of Segment 15 (cont'd)

Unit	Address Range	Size	Access Type	
			Read	Write
Reserved	F003 E100 _H - F005 FFFF _H	–	SPBBE	SPBBE
Memory Test Unit (MTU)	F006 0000 _H - F006 FFFF _H	64 Kbyte	access	access
Reserved	F007 0000 _H - F007 FFFF _H	–	SPBBE	SPBBE
High Speed Serial Link (HSSL)	F008 0000 _H - F008 03FF _H	4x256 byte	access	access
Reserved	F008 0400 _H - F008 FFFF _H	–	SPBBE	SPBBE
High Speed Communication Tunnel (HSCT)	F009 0000 _H - F009 FFFF _H	64 Kbyte	access	access
Reserved	F00A 0000 _H - F00B FFFF _H	–	SPBBE	SPBBE
I2C 0 (I2C0)	F00C 0000 _H - F00C FFFF _H	64 Kbyte	access	access
I2C 0 System Control Register (I2C0)	F00D 0000 _H - F00D 00FF _H	256 byte	access	access
Reserved	F00D 0100 _H - F00F FFFF _H	–	SPBBE	SPBBE
Global Timer Module (GTM)	F010 0000 _H - F019 FFFF _H	640 Kbyte	access	access
Reserved	F01A 0000 _H - F7FF FFFF _H	–	SPBBE	SPBBE
Reserved	F800 0000 _H - F800 04FF _H	–	SRIBE	SRIBE
Program Memory Unit 0 (PMU0)	F800 0500 _H - F800 05FF _H	256 byte	access	access
Reserved	F800 0600 _H - F800 0FFF _H	–	SRIBE	SRIBE
Flash Register (PMU0)	F800 1000 _H - F800 23FF _H	5 Kbyte	access	access

Table 3-3 On Chip Bus Address Map of Segment 15 (cont'd)

Unit	Address Range	Size	Access Type	
			Read	Write
Reserved	F800 2400 _H - F86F FFFF _H	–	SRIBE	SRIBE
SRI Crossbar (XBar_SRI)	F870 0000 _H - F870 04FF _H	5x256 byte	access	access
Reserved	F870 0500 _H - F870 07FF _H	–	SRIBE	SRIBE
Local Memory Unit (LMU)	F870 0800 _H - F870 08FF _H	256 byte	access	access
Reserved	F870 0900 _H - F870 0BFF _H	–	SRIBE	SRIBE
FFT Accelerator (FFT, Emulation Device only)	F870 0C00 _H - F870 0CFF _H	256 byte	access	access
Reserved	F870 0D00 _H - F87F FFFF _H	–	SRIBE	SRIBE
CPU0 SFR	F880 0000 _H - F880 FFFF _H	64 KByte	access	access
CPU0 CSFR	F881 0000 _H - F881 FFFF _H	64 KByte	access	access
CPU1 SFR	F882 0000 _H - F882 FFFF _H	64 KByte	access	access
CPU1 CSFR	F883 0000 _H - F883 FFFF _H	64 KByte	access	access
Reserved	F884 0000 _H - F8FF FFFF _H	–	SRIBE	SRIBE
Reserved for TC26x B-Step Emulation Device Registers (ED Reg)	F900 0000 _H - F90F FFFF _H	1 MB	access	access
Reserved	F910 0000 _H - FFFF FFFF _H	–	SRIBE	SRIBE

3.4 Memory Module Access Restrictions

Table 3-4 describes which type of accesses are possible to the different memories in the TC26x B-Step.

Table 3-4 Possible Memory Accesses¹⁾

Memory		Bit	Byte		Half-word		Word		Double-word	
		rmw	r	w	r	w	r	w	r	w
PMI ²⁾	PSPR	y	y	y	y	y	y	y	y	y
	PTAG ³⁾	-	-	-	-	-	y	y	-	-
	PCACHE	y	y	y	y	y	y	y	y	y
DMI ²⁾	DSPR	y	y	y	y	y	y	y	y	y
	DTAG ³⁾	-	-	-	-	-	y	y	-	-
	DCACHE	y	y	y	y	y	y	y	y	y
LMU ²⁾	LMURAM	y	y	y	y	y	y	y	y	y
PMU	BROM	-	y	-	y	-	y	-	y	-
	PFLASH	-	y	-	y	-	y	y	y	y
	DFLASH	-	y	-	y	-	y	y	y	y

1) 'y' means: supported. '-' means: not supported

2) The module also supports SRI 2-Word and 4-Word Block read and write accesses.

3) TAG SRAMs are not meant to be used as general SRAMs and can be accessed only with 32-bit data access and only with 64 bit aligned address. Mapping of TAG SRAMs in the address map can be used as additional option for memory testing. The TAG SRAM size is below 24 bit, so the 8 MSB of a 32 bit write or read are dont care.

3.5 Side Effects from Modules to CPU0 Data Scratch Pad SRAM (DSPR0)

Pls. note that a part of the CPU0 DSPR will be overwritten by the start-up (BootROM) procedure after cold power-on as well as the information in this area should be used and in case modified by the (user) SW procedure when preparing the device to enter Stand-By mode. The details are described in the Firmware chapter, sub-chapter 'RAMs Handling'.

4 TC26x B-Step BootROM Content

The BOOT_TC26X consists basically of three parts:

- Startup Software (short name SSW);
- Software modules implementing additional functions (Bootstrap Loaders);
- Test Firmware.

4.1 Startup Software

The Startup Software is the first software executed after a chip reset.

TC26x SSW is executed on CPU0 - all other CPUs are kept in Halt-state during boot, to be started by user software - and SSW has:

- one only entry point - SSW start address in BootROM is the reset value in Program Counter register of the CPU0. From this location an instruction is fetched and this is the first instruction executed after any device start-up.
 - immediately after this entry point the firmware checks for testmode, and in case testmode is selected - jump to test firmware is executed
- one only exit point - the last SSW instruction performs a jump to the first user code instruction. This first user instruction can be fetched from different locations depending on the start-up configuration as selected by the user.

The Startup Software contains procedures to initialize the device depending on one or more from the following:

- information previously stored into dedicated Flash locations
- the current state of special bits/fields in dedicated register/memory locations
- the type of event which has triggered the SSW-execution (the last reset event)
- values applied to external (configuration-) pins (optional)

The SSW also calls - in case - other firmware modules.

4.1.1 Events triggering SSW execution

SSW execution can be triggered by different events. SSW recognizes the triggering (reset) event and takes (partially) different execution flows.

4.1.1.1 Power-on

This is the initial powering-up of the device after the supply has been switched off, or in other words - the only way to generate this reset event is by applying power to a previously un-powered device.

The conditions at which the SSW execution starts upon this event include:

- all the registers are in their initial (reset) state
- Flash is under reset - meaning not active, and not ready to perform any (read/erase/program) operation

- RAMs' content is undefined
- clock system is in its initial state

Due to the overall “fully initial” state of the device upon power-on, the SSW flow is respectively longer in this case and covers the biggest amount of activities compared to other reset events.

4.1.1.2 System reset

Attention: *From SSW point of view, the handling of system reset is the same as of “warm power-on”.*

Therefore further in this Chapter when system reset is referred - the same SSW handling applies upon warm power-on, exceptions will be in case specially notified.

This reset event can be requested by different sources:

- device internal hardware - from modules like watchdog timer and security/memory control logic
- external hardware - when active signal is applied to defined device pin(s)
- software - when defined control bit(s) are respectively installed during user code execution

For most of the sources, generating system reset is a feature configurable by software.

The exception is PORST pin, applying active low level to which generates system reset only if the supply voltage is above defined level permanently in the surrounding time window - e.g. the device is continuously powered before and during system reset activation. Otherwise - upon a supply drop below some level - power-on is immediately triggered. All this functionality is supported by the EVR module.

The conditions at which SSW execution starts upon system reset include:

- all the registers are in their initial (reset) state
- Flash is under reset - meaning not active, and not ready to perform any (read/erase/program) operation
- RAMs' content is the same as just before the system reset has been triggered
- clock system is in its initial state

As seen from the above list, the only difference in the initial system status after power-on and system reset is in that the content of RAMs is not changed by the reset event.

4.1.1.3 Application reset

Similarly to system reset, an application reset can be requested by different sources: internal/external hardware as well as software. For all the sources, generating system reset is a feature configurable by software.

The conditions at which SSW execution starts upon application reset include:

- registers connected to this reset type are in their initial (reset) state

TC26x B-Step BootROM Content

- Flash is in read mode
- all the rest - RAMs and surround logic, clock system, registers under system reset - is not affected by this event.

Following the limited changes within the device status upon application reset, the SSW flow is shortest in this case.

4.1.2 Clock system during start-up

The state of clock system during device start-up depends on the reset event type:

- upon power-on and system reset - clock system is in its initial state, namely:
 - $f_{SRI} = f_{CPU0} = f_{FSI} = f_{BACK} = 100\text{MHz}$ nominal
 - $f_{SPB} = f_{STM} = f_{BACK}/2 = 50\text{MHz}$ nominal
 - in Emulation Device (ED) $f_{BBB} = f_{BACK}/2 = 50\text{MHz}$ nominal
 - PLL and VCO are in power-down mode
- upon application reset - clock system does not change its state, therefore the device runs at same frequency and clock source as before the reset event

With the same clock system state the first user code is started, with exception of the Bootstrap Loader mode upon power-on - refer to [Chapter 4.2](#).

4.1.3 RAM overwrite during start-up

Start-up procedure upon power-on and system reset can overwrite up to 8kByte at the beginning of CPU0 DSPR. Therefore this area should not be used by application software to save data, which must be preserved through warm power-on or system reset.

4.1.4 Boot Options Summary

This chapter summarizes the TC26x startup configurations in user mode.

Internal Start

In this basic startup mode, the first user instruction is fetched from address A000 0020_H in Internal Program Flash of the device.

Bootloader Modes

Different Bootstrap Loader routines are used in these modes to download code/data into the Instruction Scratchpad Memory SPRAM (PSPR), as follows:

- ASC bootloader
- CAN bootloader
- Generic bootloader mode - the bootloading procedure itself here is in fact one of the two above possibilities - ASC/CAN. The selection either to use ASC- or CAN-

TC26x B-Step BootROM Content

communication protocol is done by the SSW upon the first byte received at the couple of pins, which are then respectively configured to ASC or to MultiCAN module.

Alternate Boot Modes

In these modes, program code is started from user-defined address but only if all defined check-conditions are satisfied. Otherwise Generic Bootstrap Loader mode can be entered to download the code into device, this code is afterwards started by the SSW.

All the information needed for the SSW to handle ABM startup mode is collected into the so-called Headers. The checks are performed according to [Start-up mode selection](#) flow as defined in [Chapter 4.1.5](#).

4.1.5 Start-up mode selection

TC26x B-Step start-up flow in regard to mode selection is shown at [Figure 4-1](#).

There one way only to select start-up configuration in TC26x B-Step:

- [Configuration by Boot Mode Index \(BMI\)](#) - according to values taken from dedicated locations in Flash (new feature for Aurix products)

Additionally, as an option of the BMI-configuration flow the well known proceeding from previous devices is supported:

- [Hardware configuration](#) - according to the values at configuration pins

The flow to evaluate start-up configuration in TC26x B-Step is as follows:

1. if error happens during Flash ramp-up or Flash Config sector is corrupted (unrecoverable data error) - SSW terminates immediately
2. SSW evaluates sequentially up to four Boot Mode Headers BMHD0...BMHD3 - for header locations refer to [Table 4-2](#), evaluation procedure - see below and [Figure 4-2](#)
 - a) if a valid header is found - start-up mode is taken accordingly
 - b) if no valid header is found - continue with step 3.
3. check if the last BMI has enabled pin configuration and the pins have selected ABM but the code check failed:
 - a) if not - continue with step 4.
 - b) if yes - execute Generic or ASC Bootstrap Loader mode according to STSTAT.HWCFG[5:4] (refer to [Table 4-3](#))
4. check if the condition (OTP protected mode disabled) is true:
 - a) if yes - initialize all security-relevant RAMs, then check either debugger is connected and halt-after-reset is requested (OSTATE.HARR=1)
 - if yes - execute Internal start mode - CPU0 will be halted before the first user instruction
 - if no - execute Generic Bootstrap Loader mode - default mode if no valid BMI
 - b) if not - check and in case enable debug access using the general evaluation sequence, then enter endless loop - an external debugger (if access granted) are able to handle further the device start-up

TC26x B-Step BootROM Content

Note: The initialization of all security-relevant RAMs here is to prevent using BSL mode as back door for reading out user information previously stored into some RAM(s).

Table 4-1 Boot Mode Header (BMHD) structure

Offset Addr.	Size Byte	Field Name	Description
00 _H	4	STADABM	User Code Start Address
04 _H	2	BMI ¹⁾	Boot Mode Index (BMI)
06 _H	2	BMHDID ¹⁾	Boot Mode Header ID (Confirmation code) = B359 _H
08 _H	4	ChkStart	Memory Range to be checked - Start Address
0C _H	4	ChkEnd	Memory Range to be checked - End Address
10 _H	4	CRCRange	Check Result for the Memory Range
14 _H	4	<u>CRCRange</u>	Inverted Check Result for the Memory Range
18 _H	4	CRChead	Check Result for the ABM Header (offset 00 _H ..17 _H)
1C _H	4	<u>CRChead</u>	Inverted Check Result for the ABM Header

1) These fields are different from the ABM Headers in previous devices.

Table 4-2 Boot Mode Headers locations

Header	Start address	End address
BMHD0	A000 0000 _H	A000 001F _H
BMHD1	A002 0000 _H	A002 001F _H
BMHD2	A000 FFE0 _H	A000 FFFF _H
BMHD3	A001 FFE0 _H	A001 FFFF _H

Table 4-1 shows the Boot Mode Header (BMHD) structure, the procedure to evaluate one BMHD includes following steps (refer to **Figure 4-2**):

1. check either Boot Mode Header ID is correct
 - a) if BMHDID = B359_H - continue with 2.
 - b) if not - continue with 7.
2. calculate the CRC of the first 24 Bytes from the ABM Header (refer to **Table 4-1**) - process the fields STADABM...CRCRange at offsets 00_H..17_H
 - a) compare the result with the CRChead value (offset 18_H)
 - if OK - continue with 2.b)
 - if not - continue with 7.

TC26x B-Step BootROM Content

- b) inverse the result value and compare with $\overline{\text{CRChead}}$ (offset 1C_H)
 - if OK - continue with 3.
 - if not - continue with 7.
 - 3. check either BMI[15:10, 7:0] value is valid - refer to **Chapter 4.1.5.2**
 - a) if yes - continue with 4.
 - b) if not - continue with 7.
- Note: Bits BMI[9:8] are excluded from this check because they control Lockstep Logic and can have arbitrary values.*
- 4. check the conditions BMI[3]=0 (pin-configuration enabled) AND HWCFG[3] pin value=0 (pin-configuration selected) AND PROCONOTP[30:29]=00_B (boot flow with standard evaluation of headers):
 - a) if all the conditions are true - **Hardware configuration** is taken (refer to **Chapter 4.1.5.1**) meaning in the further handling STSTAT.HWCFG[6] will be ignored and the mode selection will be done according to **Table 4-3**, continue with 5.
 - b) if not - **Configuration by Boot Mode Index (BMI)** is taken (refer to **Chapter 4.1.5.2**) and respective value installed in STSTAT.HWCFG, then continue with 5.
 - 5. check either STSTAT.HWCFG value selects Alternate Boot Mode (ABM)
 - a) if yes - continue with 6.
 - b) if not - exit this procedure, BMI is OK, mode is selected in STSTAT.HWCFG
 - 6. calculate the CRC over the memory address range ChkStart...ChkEnd (start- and end- addresses taken from offsets 08_H and 0C_H respectively)
 - a) compare the result with the CRCrange value (offset 10_H)
 - if OK - continue with 6.b)
 - if not - continue with 7.
 - b) inverse the result value and compare with $\overline{\text{CRCrange}}$ (offset 14_H)
 - if OK - exit this procedure, BMI and code are OK, ABM is selected in STSTAT.HWCFG
 - if not - continue with 7.

Attention: If (some part of) the memory range selected by ChkStart...ChkEnd is into illegal address area (refer to Memory Map Chapter), attempting to read from there will cause a trap and SSW will terminate on error!

- 7. check again so called Magic number in Flash Config sector
 - a) if OK - exit this procedure, header and BMI check failed
 - b) if wrong - terminate SSW immediately on error

Note: The last step above is to reduce the probability that a hacker attack disturbs BMI/ABM checks targeting to cause unintentional (from the original user code) entry into Bootstrap Loader mode.

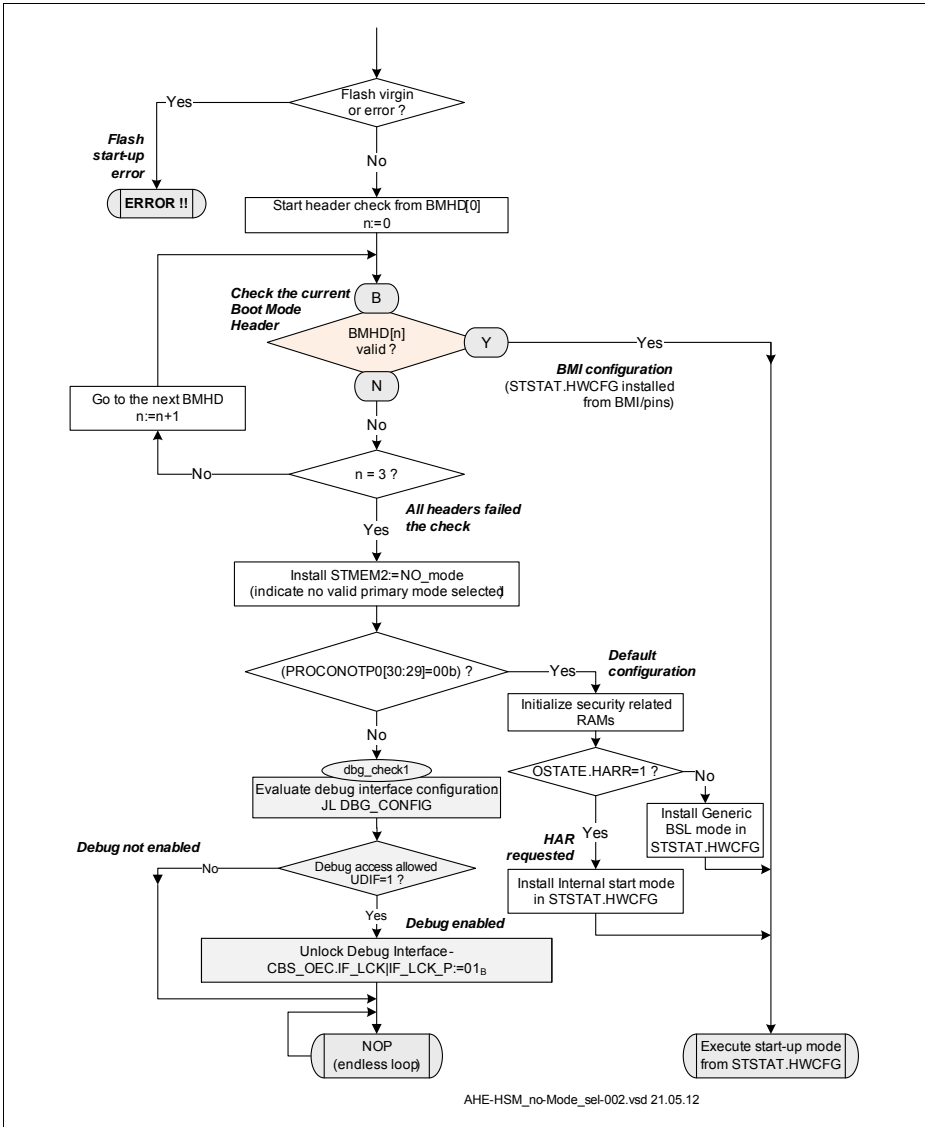
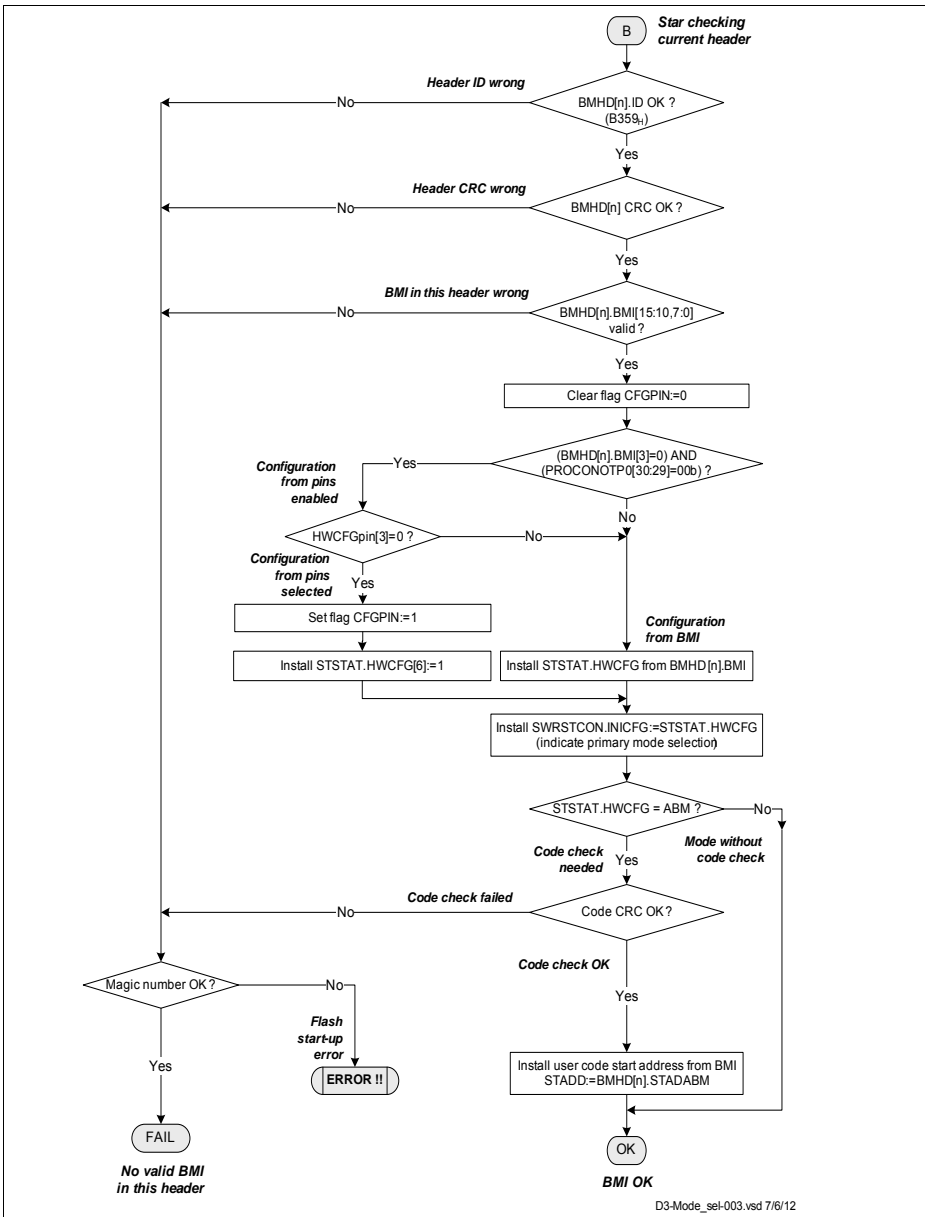


Figure 4-1 Start-up mode selection flow in TC26x B-Step

TC26x B-Step BootROM Content



D3-Mode_sel-003.vsd 7/6/12

Figure 4-2 Mode selection in TC26x B-Step: check of one Boot Mode Header

4.1.5.1 Hardware configuration

The start-up mode selection by configuration pins HWCFG[5:4] in TC26x B-Step is shown in [Table 4-3](#). The values at these pins are latched by hardware upon any reset (deactivation - rising edge) into register which is read and evaluated by SSW during start-up processing.

Note: HWCFG[2:0] pins are used for EVR control, HWCFG[6] - for pull-up/tristate pin configuration, HWCFG[7] is not available for TC26x B-Step.

Table 4-3 Start-up mode selection by pins in TC26x B-Step

HWCFG pins		Start-up Mode
[5]	[4]	
1	1	Internal start from Flash
1	0	ABM, Generic Bootstrap Loader on fail
0	1	ABM, ASC Bootstrap Loader on fail
0	0	Generic Bootstrap Loader

4.1.5.2 Configuration by Boot Mode Index (BMI)

The Boot Mode Index (BMI) is 2 Byte value holding information about start-up mode of the device and Lockstep mode enable/disable.

The data structure of BMI is described below, followed by the definition of FLASH0_PROCOND bits which control RAM initialization, oscillator circuit (OSC) configuration and ESR0 pin handling by SSW.

NOTE: The start-up mode selections defined as "Reserved" in BMI.HWCFG are treated as invalid by SSW

BMI structure

BMI

Boot Mode Index

Boot Mode Header offset 04_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						LCL1 LSEN	0	0	HWCFG			PIN DIS	0		
r						r	r	r	r			r	r		

TC26x B-Step BootROM Content

Field	Bits	Typ	Description
0	[15:10], 8,7,[2:0]	r	Reserved
PINDIS	3	r	Mode selection by configuration pins: 0 Mode selection by HWCFG pins is enabled 1 Mode selection by HWCFG pins is disabled
HWCFG	[6:4]	r	Start-up mode selection: 111 _B Internal start from Flash 110 _B Alternate Boot Mode (ABM) 101 _B Reserved 100 _B Generic Bootstrap Loader 011 _B ASC Bootstrap Loader else Reserved
LCL1LSEN	9	r	Lockstep Comparator Logic control for CPU1: 0 Lockstep is disabled for CPU1 1 Lockstep is enabled for CPU1

FLASH0_PROCOND
DFlash Protection Configuration (F800 1030_H)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RPRO	0	0	ESR0CNT												
rh	rh	rh	rh												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP 3 EN	CAP 2 EN	CAP 1 EN	CAP 0 EN	APR EN	MODE	OSC CFG	RAMINSEL				RAMIN		NSAF ECC	L	
rh				rh				rh		rh	rh				

Field	Bits	Type	Description
NSAF ECC, L, RPRO	0,1,31	rh	Flash related control bits

TC26x B-Step BootROM Content

Field	Bits	Type	Description
RAMIN	[3:2]	rh	RAM initialization by SSW control: x0 _B Initialize RAMs as selected by RAMINSEL upon cold power-on 0x _B Initialize RAMs as selected by RAMINSEL upon warm power-on/system reset
RAMINSEL	[7:4]	rh	RAMs selected for initialization by SSW: xxx0 _B RAMs in CPU0 are selected for initialization xx0x _B RAMs in CPU1 are selected for initialization x0xx _B Reserved - not used in TC26x B-Step 0xxx _B Reserved - not used in TC26x B-Step
OSCCFG	8	rh	OSC configuration by SSW: 0 OSC is not configured by SSW - bits [15:9] are ignored 1 OSC is configured by SSW - bits [15:9] are installed into respective SCU_OSCCON bits/fields
MODE	[10:9]	rh	OSC mode - installed by SSW into SCU_OSCCON.MODE if OSCCFG=1; otherwise ignored
APREN	11	rh	Amplitude Regulation Enable - installed by SSW into SCU_OSCCON.APREN if OSCCFG=1; otherwise ignored
CAPxEN (x=0..3)	[15:12]	rh	Capacitance x Enable (x=0..3) - installed by SSW into SCU_OSCCON.CAPxEN if OSCCFG=1; otherwise ignored
ESR0CNT	[27:16]	rh	ESR0 prolongation counter FFF _H Application Reset Indicator/ESR0 pin is not handled by SSW 000 _H Application Reset Indicator - ESR0 pin when respectively configured - is released by SSW as soon as possible (zero prolongation) after device reset else Application Reset Indicator - ESR0 pin at low level, when respectively configured - is hold by SSW active for (ESR0CNT)*10μsec after device reset release

Field	Bits	Type	Description
0	[30:28]	rh	Reserved

4.1.6 Startup Software Main Flow

Below the SSW functionality is outlined during different execution-steps (refer to [Figure 4-3](#)) and upon various configuration settings.

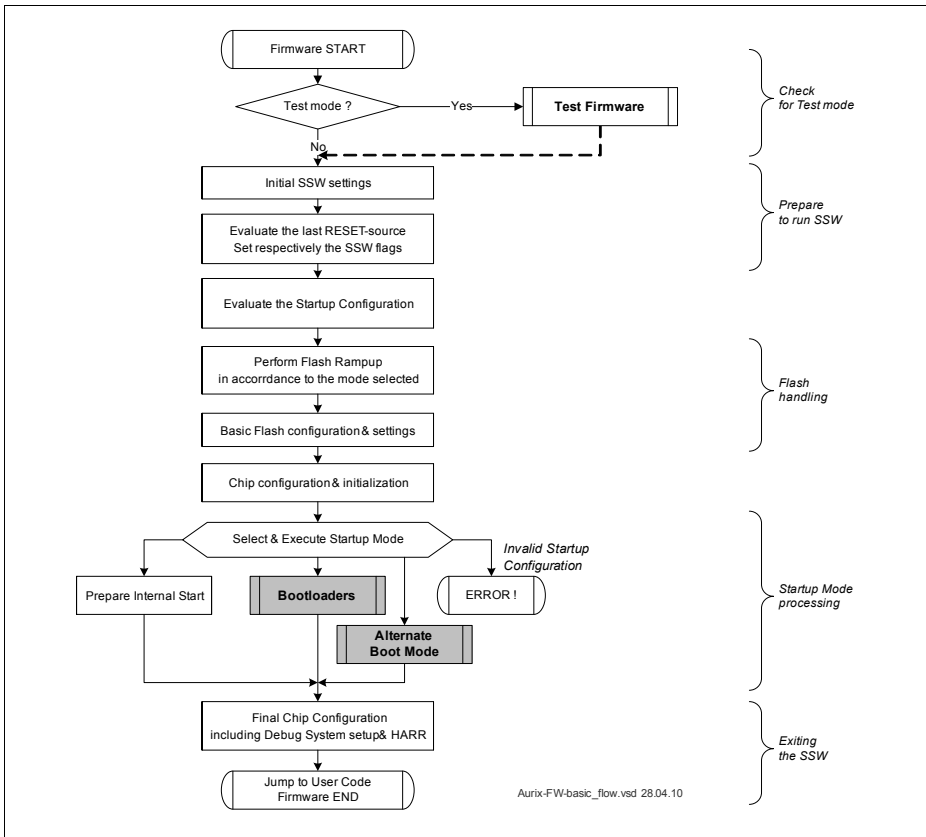


Figure 4-3 TC26x Firmware: Firmware main flow

4.1.6.1 Basic Device Settings

The target of this functional module is to initialize several TC26x-registers with the values, which will be first seen by the user - or generally available - after exiting the SSW.

4.1.6.2 Stand-by Controller preparation

Upon cold power-on, SSW checks either the Stand-by Controller (SCR) reset was activated by EVR, indicated by SCU_PMSWSTAT.SCRST=1. If this is the case - meaning no exit from stand-by mode took place and this is an initial power-on of the SCR itself - SSW performs the following:

- initialize SCR XRAM
- release SCR reset - the 8-bit microcontroller is activated to execute its start-up procedure
- check either SCR has booted successfully
 - if yes - do nothing, upon SSW exit SCR will be ready and waiting for further handling by user software
 - if not - activate SCR reset, then disable the module by setting SCU_PWSWCR0.SCREN=0

Therefore, SCR usage will be only possible if at the user software start SCU_PWSWCR0.SCREN=1. In such a case, code can be downloaded from user Flash into SCR XRAM via QSPI interface and its execution by SCR started. For more information - refer to SCR documentation.

4.1.6.3 RAMs Handling

By default SSW in TC26x initializes SCR XRAM upon cold power-on, if no exit from stand-by (see above).

For all the security-relevant RAMs initialization upon power-on/system reset can be requested by **FLASH0_PROCOND** register content (refer to [Chapter 4.1.5.2](#)).

4.1.6.4 Select and Prepare Startup Modes

TC26x B-Step User Startup Configurations and modes are summarized below.

Internal Start

This is the basic TC26x type of operation in which the user code is started out of the Internal Flash Memory.

The User Start Address is set to A000'0020_H.

Bootstrap Loader Modes

Two Bootstrap Loader modes are supported in TC26x B-Step:

- ASC Bootloader - ASC communication protocol via ASC pins
- Generic Bootloader via CAN pins - the communication protocol is automatically selected by the SSW between ASC and CAN

TC26x B-Step BootROM Content

After downloading (in case) the code, the User Start Address STADD is set to the beginning of Program Scratchpad RAM PSPR at C000 0000_H.

Alternate Boot Mode

The handling of this (ABM) start-up mode is according to the flow at [Figure 4-1](#) and [Figure 4-2](#).

If this mode is selected by a valid BMI Header (or by pins if pin configuration is enabled in a valid Header) - the handling depends on code check result:

- if code check pass - the User Start Address STADD is set to the respective value from the header (STADABM_x) and the code will be started from that address
- if code check fails - the current Boot Mode Header is considered as not valid and not used for configuration respectively ABM not effectively taken, then
 - if more Header(s) exist - the next one is taken for evaluation
 - if selected by pin-configuration enabled by BMHD3 (i.e. no more Headers available) - Generic or ASC Bootloader mode is taken in accordance to the pin-selection (refer to [Table 4-3](#))

4.1.6.5 Final Chip Settings

The last device configuration steps performed by the SSW include:

Flash access handling

SSW flow here includes:

- fetches (code/data, from Program/Data Flash) control
 - if user code will be started from internal flash - all fetches are enabled
 - otherwise - according to the protection (active/not) state
- Program and Erase operations are enabled

No handling of Unique Chip ID and Calibration data

This information (Unique Chip ID and Calibration data) is now stored in User Configuration Block which is readable for the user. As consequence it is no more installed into DSPR but can be read directly by the application software.

Debug System handling

As first point of this processing, the SSW internal flag Unlock Debug Interface is set if debug access to device will be enabled according to the following evaluation sequence:

1. SSW checks either an external tool has requested debug access by writing a defined content (32 bit value CMD_KEY_EXCHANGE=76D6E24A_H for TC26x B-Step) into COMDATA register
 - a) if yes - continue with the next step

TC26x B-Step BootROM Content

- b) if not - go to step 4.
- 2. still in Cerberus Communication mode, SSW confirms request reception and receives 8 further words through COMDATA register
- 3. the data received (256 bits) is sent by SSW to PMU to be checked as debug interface password and the result is evaluated by SSW:
 - a) if OK - set SSW internal flag, debug interface will be unlocked, exit the sequence
 - b) if fail - continue with the next step
- 4. install into COMDATA 32-bit value serving for an external tool to identify the device connected - UNIQUE_CHIP_ID_32BIT
- 5. check if Flash protection is activated:
 - a) if yes - debug interface will be left locked, no debug access to device, exit the sequence
 - b) if not - set SSW internal flag, debug interface will be unlocked, exit the sequence

The following steps are introduced to support so called “Debug Self-destructive Entry” feature. The target is, if this feature is activated and debugging is requested by correct debug password (see steps 1...3 above) - to destroy vital functions of the device (so that it is not usable in a real application anymore) before effectively granting debug access.

The steps performed are:

- 6. check either “Debug Self-destructive Entry” feature is activated for the device (PROCONHSMOTP.DESTDBG=11_B):
 - a) if yes - continue with the next step
 - b) if no - debug access granted, go to step 4.b)
- 7. using PMU command sequences, erase and re-program UCB_DBG with the previous password but setting “Entered Debug Mode” marker (EDM=11_B) in PROCONDBG; then go to step 4.b) - debug access granted

Next, Halt after Reset is prepared if requested. The SSW processing here is as follows:

- check either external (debug) access to the device will be generally granted
 - if Not -> exit this procedure
- check either Halt After Reset is requested
 - if Not -> exit this procedure
- configure a Break After Make breakpoint at the last SSW instruction
- enable On-Chip Debug Support system.

Note: The final debug-related operation - unlocking (in case) the debug interface - is performed later.

ESR0 pin handling

If both the conditions

- ESR0CNT<>FFF_H in **FLASH0_PROCOND** (refer to register description in **Chapter 4.1.5.2**) AND
- ESR0-pin configuration upon SSW entry is open-drain reset output (SCU_IOCR.PC0 in [1110_B, 1010_B])

are satisfied, SSW will:

- release ESR0 pin - by installing SCU_ESROCFG.ARC:=1 which clears Application Reset Indicator in SCU_ESROCFG.ARI - with some delay after device internal reset is released (i.e. CPU0 started), the delay is defined as follows
 - if **FLASH0_PROCOND**.ESR0CNT=000_H - about 500µsec upon cold power-on, not longer than 20µsec otherwise
 - if **FLASH0_PROCOND**.ESR0CNT=001_H...FFE_H - (ESR0CNT)*10µsec
- wait until ESR0 pin is effectively high - indicated by SCU_IN.P0=1 - at the very SSW end and before jumping to the first user instruction

To generate configurable ESR0 delay after device reset release, SSW uses System Timer 0 (STM0) and especially takes into account the following default settings after power-on/system reset:

- STM is reset and starts counting from zero
- STM is clocked with 50MHz i.e. $f_{STM} = F_{EVR_OSC} / 2$

Attention: Both the above conditions could not be true after application reset, if default settings are changed by user code executed after power-on/system reset. In such a case, ESR0 handling by SSW will not work correctly, meaning the real prolongation will not correspond to ESR0CNT value configured.

Lockstep configuration

Upon cold power-on only SSW performs Lockstep configuration as follows:

- if valid BMI has been found during start-up mode evaluation:
 - Lockstep control for CPU1 is installed in LCLCON1.LSEN from BMI[9]

4.1.6.6 Ending the SSW and Starting the User Code

The last steps executed by the SSW are:

- activate the Startup Protection
- unlock (in case) Debug Interface
- jump to the first User Instruction (for TriCore) at address STADD.

Additionally, if all the following conditions are satisfied:

- the device is an ED
- debug access to device is allowed
- halt after reset is not requested
- the last reset has been a power-on (cold or warm)

SSW performs at its very end additional checks and in case all these succeed - the last SSW instruction jumps not to the “standard” STADD but to an address inside EMEM (Emulation Memory). This SSW part implements the sequence defined in Emulation Device Target Specification, Chapter “Startup with prolog code in EMEM”.

4.2 Bootstrap Loaders

These routines provide mechanisms to load an user program via selected interface by moving code into PMI Scratchpad RAM (an Internal Program memory). The loaded code is started after exiting the BootROM.

The exact communication modules and pins used in different modes are summarized in [Table 4-4](#).

4.2.1 ASC Bootstrap loader

The ASC Bootloading routine implements the following steps:

- Rx/D/TxD pins configuration (refer to [Table 4-4](#)) is done in accordance to the TC26x definitions, as well as depending either the routine is invoked upon “ASC Bootloader”-startup mode (ASC-only pins are used) or following an ASC-protocol detection upon “Generic Bootloader”-mode (CAN/ASC-shared pins are used but configured to ASC module)
- baudrate calculation is done based on the zero Byte sent by the host
- ASCLIN channel n (refer to [Table 4-4](#)) is initialized (without enabling the receiver) to the baudrate as determined, 8 data and 1 stop bit
- acknowledge byte D5_H is sent to the host indicating the device is ready to accept a data transfer
- after the acknowledge byte is transmitted, the receiver is enabled
- the bootloader enters a loop waiting to receive exactly 128 bytes which are stored as 32 words in CPU0 Program Scratchpad RAM starting from address C000 0000_H

Once 128 bytes are received, the SSW continues further - refer to [Figure 4-3](#). After exiting the SSW, user code will be started from address C000 0000_H (CPU0_PSPR).

4.2.2 CAN Bootstrap Loader

The CAN bootstrap loader transfers program code/data via node n (refer to [Table 4-4](#)) of the MultiCAN module into CPU0 Program Scratchpad RAM.

Note: TC26x B-Step requires that an external clock source (oscillator) must be connected between XTAL1/XTAL2 pins if CAN Bootstrap Loader mode will be selected upon power-on/system reset.

Oscillator frequency should be greater or equal to 20 MHz for a baud rate of 1 Mbit/s.

The CAN Bootstrap Loader transfers data from an external host to the TC26x B-Step using eight-byte data frames. The number of data frames to be received is programmable and determined by the 16-bit data message count value DMSGC.

The communication between TC26x B-Step and external host is based on the following three CAN standard frames:

- Initialization frame - sent by the external host to the TC26x B-Step

TC26x B-Step BootROM Content

- Acknowledge frame - sent by the TC26x B-Step to the external host
- Data frame(s) - sent by the external host to the TC26x B-Step

The initialization frame is used in the TC26x B-Step for baud rate detection. After a successful baud rate detection is reported to the external host by sending the acknowledge frame, data is transmitted using data frames.

Initialization Phase

The first task is to determine the CAN baud rate at which the external host is communicating. This task requires the external host to send initialization frames continuously to the TC26x B-Step. The first two data bytes of the initialization frame include a 2-byte baud rate detection pattern (5555_H), an 11-bit (2-byte) identifier ACKID for the acknowledge frame, a 16-bit data message count value DMSGC, and an 11-bit (2-byte) identifier DMSGID to be used by the data frame(s).

The CAN baud rate is determined by analyzing the received baud rate detection pattern (5555_H) and the baud rate registers of the MultiCAN module are set accordingly. The TC26x B-Step is now ready to receive CAN frames with the baud rate of the external host.

Acknowledge Phase

In the acknowledge phase, the bootstrap loader waits until it receives the next correctly recognized initialization frame from the external host, and acknowledges this frame by generating a dominant bit in its ACK slot. Afterwards, the bootstrap loader transmits an acknowledge frame back to the external host, indicating that it is now ready to receive data frames. The acknowledge frame uses the message identifier ACKID that has been received with the initialization frame.

Data Transmission Phase

In the data transmission phase, data frames are sent by the external host and received by the TC26x B-Step. The data frames use the 11-bit data message identifier DMSGID that has been sent with the initialization frame. Eight data bytes are transmitted with each data frame. The first data byte is stored in Program Scratchpad RAM starting from address C000 0000_H. Consecutive data bytes are stored at incrementing addresses.

Both communication partners evaluate the data message count DMSGC until the requested number of CAN data frames has been transmitted.

After the reception of the last CAN data frame, the SSW continues further - refer to [Figure 4-1](#). After exiting the SSW, user code will be started from address C000 0000_H (CPU0_PSPR).

4.2.3 Summary of Bootstrap Loader Modes

This table summarizes the external hardware provisions for bootstrap loader modes in TC26x B-Step.

Table 4-4 Configuration Data for Bootstrap Loader Modes

Bootstrap Loader Mode	Channel/ node	RxD Line	TxD Line	Transferred Data	Supported baudrates
ASC Bootstrap Loader mode	ASCLIN0	P15.3	P15.2	128 Bytes	28k...2500k
Generic Bootstrap Loader mode - ASC protocol	ASCLIN0	P14.1	P14.0	128 Bytes	28k...2500k
Generic Bootstrap Loader mode - CAN protocol	CAN1	P14.1	P14.0	8×n Bytes ¹⁾	up to 1000k ²⁾

1) n = DMSGC, Data Message Count sent by the host with Initialization frame.

2) with 20MHz XTAL clock source

4.3 Shutdown request handler

All the active CPUs in TC26x B-Step jump unconditionally to the entry point of this handler upon any warm reset request - refer to Reset Control Unit Section in SCU ITS.

It is guaranteed by hardware, this handler can not be interrupted by any other (interrupt/trap) request, and upon its end all the CPUs are in stable passive state reached by a controlled ramp-down sequence, preventing big current jumps.

At its entry point - common for all the CPUs - the firmware causes any running CPU to jump further to its own handler. For this purpose CORE_ID register is read and because this register value is individual for any CPU - upon CORE_ID=0, 1 or 2 the firmware jumps to the routine for respective CPU.

The functionality of all handlers is similar, namely:

- prepare work data for execution of average-power loop
- execute average-power loop until SCU_RSTCON2.TOUT_{yy}=1
- execute WAIT instruction

Upon completion of the above sequence, CPU_x has reached passive state *yy* microseconds after shutdown request activation, whereas:

- *yy*=60 for CPU0
- *yy*=40 for CPU1

4.4 Power Supply Friendly Debug Monitor

TC26x B-Step BootROM contains a routine called Power Supply Friendly Debug Monitor (PSFDM). The purpose on this routine is to minimize the risk of getting EVR voltage over/undershoot due to a sudden current drop when more than one CPU is halted by OCDS, respectively current peak when the CPUs are released from halt.

PSFDM routine is an “independent/stand alone” module inside BootROM which is never executed during device start-up (SSW) or production test (TSW). It is intended to be used by CPUs as debug trap handler instead of halting. This means the debugger must configure properly the debug trap vector - it should point to address AFFF C020_H (inside BootROM).

Upon a debug event, trap will be triggered for all configured CPUs starting PSFDM execution, then they can be halted individually by the tool if needed.

PSFDM routine contains a “repeat **action** until **condition**” loop whereas:

- the **action** represents a sequence which execution consumes power close to the consumption of an average program code. For this purpose mixture between two TC1.6 instructions is implemented:
 - most power intensive (MADD.Q)
 - less power intensive (NOP)
- the **condition** to exit the loop is CBS_TLS.TL2=1 meaning OTGS Trigger Line 2 being activated

To continue user code execution, the debugger must:

- release from halt those CPUs which have being halted - so all the CPUs are running PSFDM as debug trap handler
- activate OTGS Trigger Line 2 - all the CPUs exit PSFDM (by RTE) and continue user code execution

At the end, after debug trap the CPUs are restarted in parallel, with a few cycles slack due to the individual CBS_TLS.TL2 polling.

4.4.1 PSFDM code with inverse exit condition

The same routine as above described is implemented in BootROM starting at AFFF FC80_H with one difference:

- the **condition** to exit the loop is CBS_TLS.TL2=0 meaning OTGS Trigger Line 2 being deactivated - i.e. the exit condition here is inverse to the above

5 CPU Subsystem

This chapter describes the implementation-specific options of the TriCore CPUs found in the Aurix series of devices. Details of both TriCore1.6P (TC1.6P) and TriCore1.6E (TC1.6E) CPUs are included. The CPU and local memory configurations of the various members of the Aurix family are detailed in the following table.

This chapter should be read in conjunction with the TriCore Architecture Manual. Topics covered by the architecture manual include:-

- Architectural Overview
- Programing Model
- CPU Registers
- Tasks and Functions
- Interrupt Handling
- Traps
- Memory Protection System
- Temporal Protection System
- Floating Point Operations
- Debug
- Instruction Set

5.1 Aurix Family CPU configurations

The different CPU and local memory configurations for the Aurix family of devices are detail in the following tables:-

Table 5-1 Processor and local memory configuration of the TC29x

Processor	Core -ID	Program Cache	Program Scratch Pad	Data Cache	Data Scratch Pad	Standby SRAM	Lock-Step
TC1.6P	2	32 KB	32 KB	8 KB	240 KB	No	No
TC1.6P	1	32 KB	32 KB	8 KB	240 KB	No	Yes
TC1.6P	0	16 KB	32 KB	8 KB	120 KB	Yes	No

Table 5-2 Processor and local memory configuration of the TC27x

Processor	Core -ID	Program Cache	Program Scratch Pad	Data Cache	Data Scratch Pad	Standby SRAM	Lock-Step
TC1.6P	2	16 KB	32 KB	8 KB	120 KB	No	No
TC1.6P	1	16 KB	32 KB	8 KB	120 KB	No	Yes
TC1.6E	0	8 KB	24 KB	0 ¹⁾	112 KB	Yes	Yes

1) TC1.6E has a 128Byte read buffer in place of a data cache

Table 5-3 Processor and local memory configuration of the TC26x

Processor	Core -ID	Program Cache	Program Scratch Pad	Data Cache	Data Scratch Pad	Standby SRAM	Lock-Step
TC1.6P	1	16 KB	32 KB	8 KB	120 KB	No	Yes
TC1.6E	0	8 KB	16 KB	0 ¹⁾	72 KB	Yes	No

Table 5-4 Processor and local memory configuration of the TC24x

Processor	Core -ID	Program Cache	Program Scratch Pad	Data Cache	Data Scratch Pad	Standby SRAM	Lock-Step
TC1.6E	0	8 KB	16 KB	0 ¹⁾	80 KB	Yes	No

Table 5-5 Processor and local memory configuration of the TC23x

Processor	Core -ID	Program Cache	Program Scratch Pad	Data Cache	Data Scratch Pad	Standby SRAM	Lock-Step
TC1.6E	0	8 KB	8 KB	0 ¹⁾	184 KB	Yes	Yes

Table 5-6 Processor and local memory configuration of the TC22x

Processor	Core -ID	Program Cache	Program Scratch Pad	Data Cache	Data Scratch Pad	Standby SRAM	Lock-Step
TC1.6E	0	8 KB	8 KB	0 ¹⁾	88 KB	Yes	Yes

1) TC1.6E has a 128Byte read buffer in place of a data cache

5.2 Central Processing Unit Features

Key CPU Features include:-

Architecture

- 32-bit load store architecture
- 4 Gbyte address range (2^{32})
- 16-bit and 32-bit instructions for reduced code size
- Data types:
 - Boolean, integer with saturation, bit array, signed fraction, character, double-word integers, signed integer, unsigned integer, IEEE-754 single-precision floating point
- Data formats:
 - Bit, byte (8-bits), half-word (16-bits), word (32-bits), double-word (64-bits)
- Byte and bit addressing
- Little-endian byte ordering for data, memory and CPU registers
- Multiply and Accumulate (MAC) instructions: Dual 16×16 , 16×32 , 32×32
- Saturation integer arithmetic
- Packed data
- Addressing modes:
 - Absolute, circular, bit reverse, long + short, base + offset with pre- and post-update
- Instruction types:
 - Arithmetic, address arithmetic, comparison, address comparison, logical, MAC, shift, coprocessor, bit logical, branch, bit field, load/store, packed data, system
- General Purpose Register Set (GPRS):
 - Sixteen 32-bit data registers
 - Sixteen 32-bit address registers
 - Three 32-bit status and program counter registers (PSW, PC, PCXI)
- Debug support (OCDS):
 - Level 1, supported in conjunction with the CPS block
 - Level 3, supported in conjunction with the MCDS block (Emulation Device only).
- Flexible memory protection system providing multiple protection sets with multiple protection ranges per set.
- Temporal protection system allowing time bounded real time operation.

TC1.6P Implementation

- Most instructions executed in 1 cycle
- Branch instructions in 1, 2 or 3 cycles (using dynamic branch prediction)
- Wide memory interface for fast context switch
- Automatic context save-on-entry and restore-on-exit for: subroutine, interrupt, trap
- Four memory protection register sets
- Dual instruction issuing (in parallel into Integer Pipeline and Load/Store Pipeline)
- Third pipeline for loop instruction only (zero overhead loop)

- Single precision Floating Point Unit (IEEE-754 Compatible)
- Dedicated Integer divide unit
- Implementation optimised for performance.
- 16 data protection ranges, 8 code protection ranges

TC1.6E Implementation

- Most instructions executed in 1 cycle
- Branch instructions in 1 or 2 cycles (using static branch prediction)
- Wide memory interface for fast context switch
- Automatic context save-on-entry and restore-on-exit for: subroutine, interrupt, trap
- Four memory protection register sets
- Single instruction issue per cycle
- Single precision Floating Point Unit (IEEE-754 Compatible)
- Dedicated Integer divide unit
- Implementation optimised for power
- 16 data protection ranges, 8 code protection ranges

5.3 TC1.6P Implementation Overview

The following sections give an overview of the TC1.6P Implementation

5.3.1 CPU Diagram

The Central Processing Unit (CPU) comprises of an Instruction Fetch Unit, an Execution Unit, a General Purpose Register File (GPR), a CPU Slave interface (CPS), and Floating Point Unit (FPU).

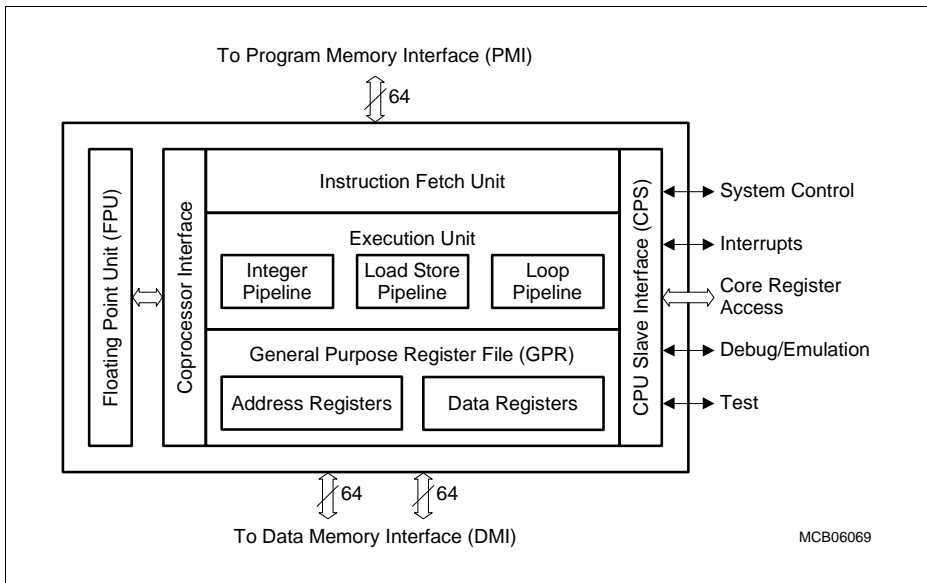


Figure 5-1 CPU Block Diagram

5.3.2 Instruction Fetch Unit

The Instruction Fetch Unit pre-fetches and aligns incoming instructions from the 64-bit wide Program Memory Interface (PMI). Instructions are placed in predicted program order in the Issue fifo. The Issue fifo buffers up to six instructions and directs the instruction to the appropriate execution pipeline.

The Instruction Protection Unit checks the validity of accesses to the PMI and the integrity of incoming instructions fetched from the PMI.

The branch unit examines the fetched instructions for branch conditions and predicts the most likely execution path based on previous branch behavior. The Program Counter Unit (PC) is responsible for updating the program counters.

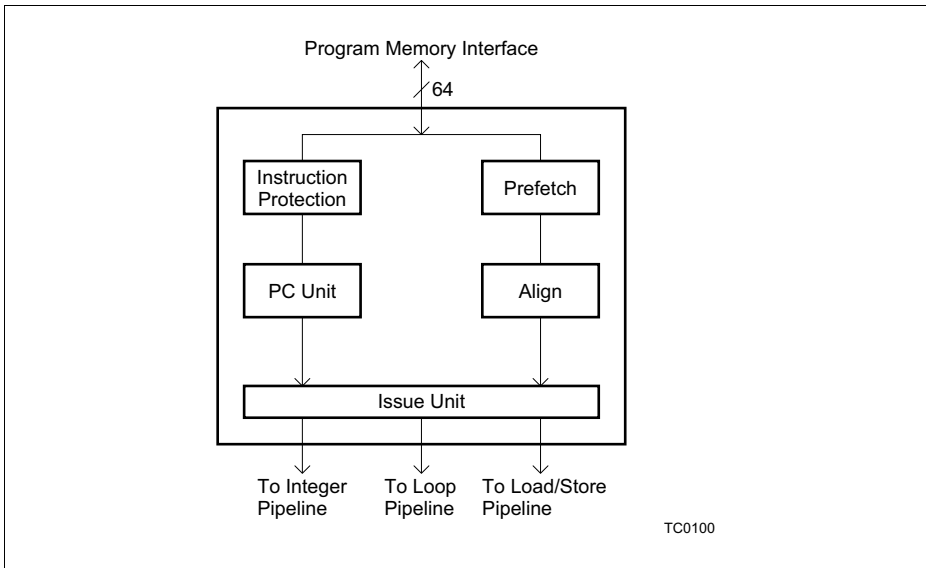


Figure 5-2 Instruction Fetch Unit

5.3.3 Execution Unit

The Execution Unit contains the Integer Pipeline, the Load/Store Pipeline and the Loop Pipeline. All three pipelines operate in parallel, permitting up to three instructions to be executed in one clock cycle. In the execution unit all instructions pass through a decode stage followed by two execute stages. Pipeline hazards (stalls) are minimised by the use of forwarding paths between pipeline stages allowing the results of one instruction to be used by a following instruction as soon as the result becomes available.

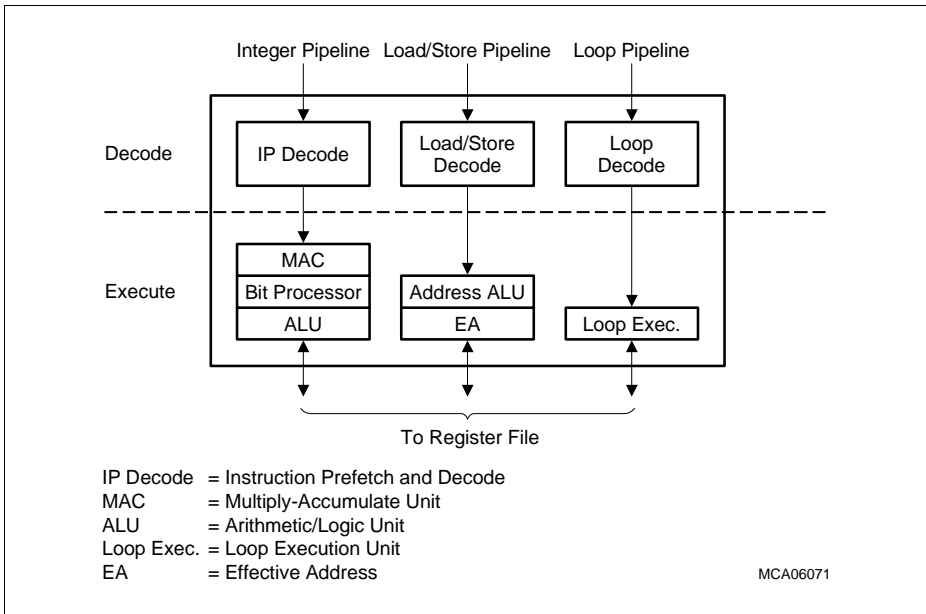


Figure 5-3 Execution Unit

5.3.4 General Purpose Register File

The CPU has a General Purpose Register (GPR) file, divided into an Address Register File (registers A0 through A15) and a Data Register File (registers D0 through D15).

The data flow for instructions issued to the Load/Store Pipeline is steered through the Address Register File.

The data flow for instructions issued to/from the Integer Pipeline and for data load/store instructions issued to the Load/Store Pipeline is steered through the Data Register File.

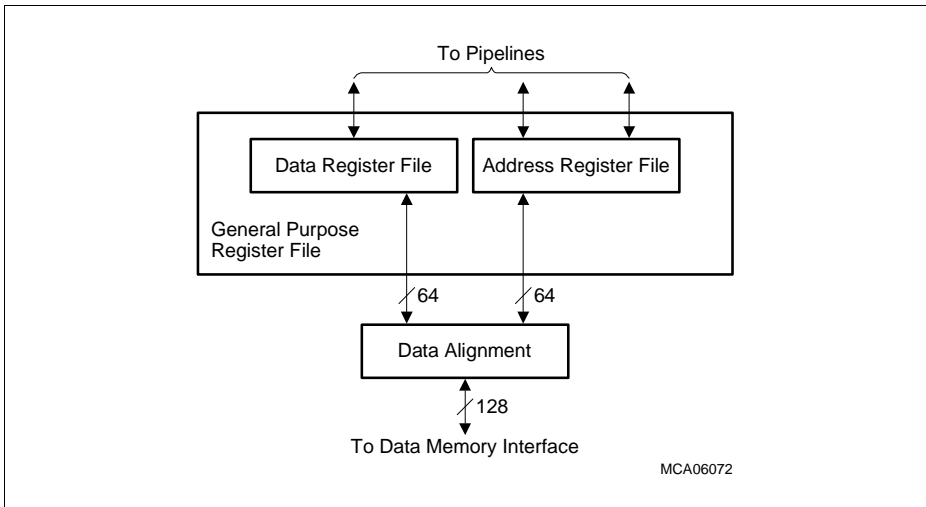


Figure 5-4 General Purpose Register File

5.4 TC1.6E Implementation Overview

The following sections give an overview of the TC1.6E Implementation

5.4.1 CPU Diagram

The Central Processing Unit (CPU) comprises of an Instruction Fetch Unit, an Execution Unit, a General Purpose Register File (GPR), a CPU Slave interface (CPS), and Floating Point Unit (FPU).

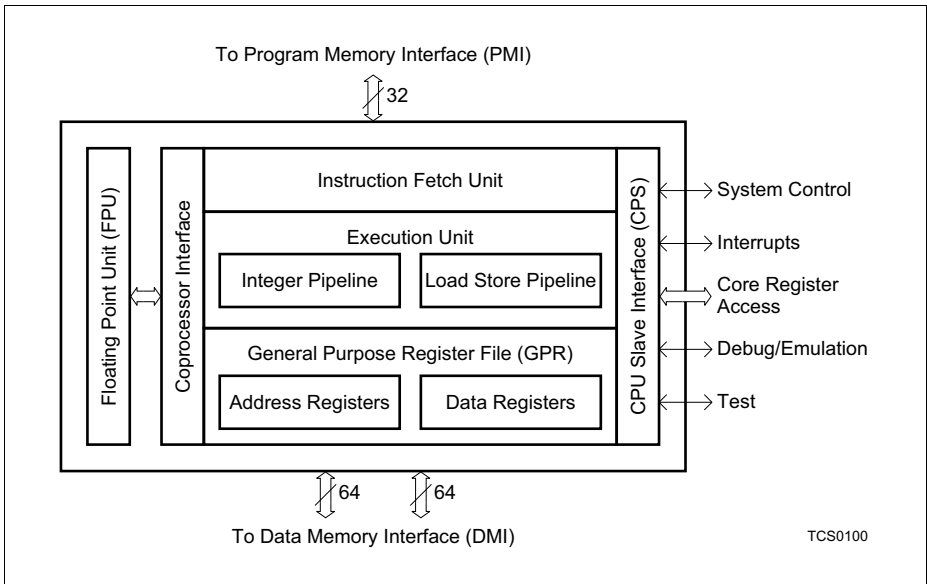


Figure 5-5 CPU Block Diagram

5.4.2 Instruction Fetch Unit

The Instruction Fetch Unit pre-fetches and aligns incoming instruction half-words from the 32-bit wide Program Memory Interface (PMI). The Issue Unit directs a single aligned instruction to the appropriate execution pipeline.

The Instruction Protection Unit checks the validity of accesses to the PMI and the integrity of incoming instructions fetched from the PMI.

The Program Counter Unit (PC) is responsible for updating the program counters.

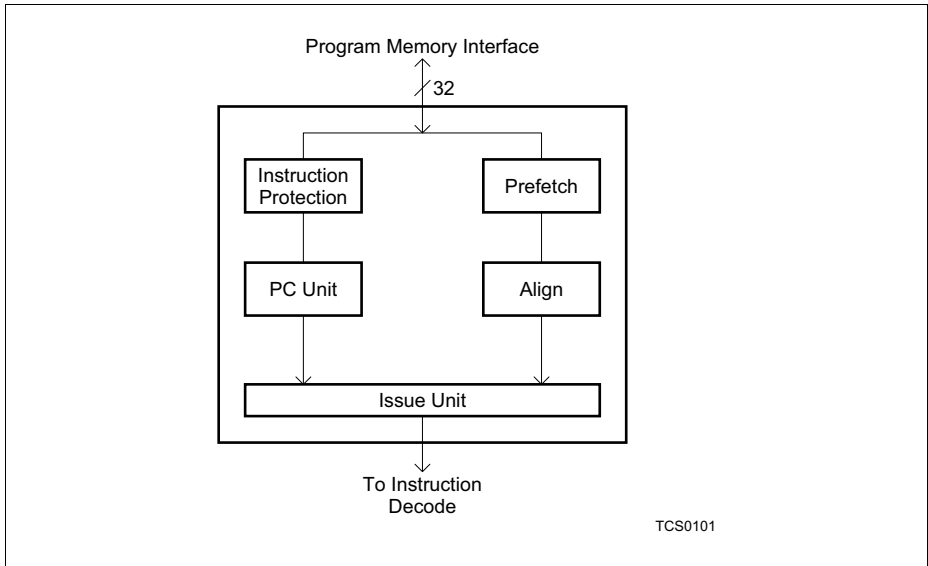


Figure 5-6 Instruction Fetch Unit

5.4.3 Execution Unit

The Execution Unit contains the Integer Pipeline and the Load/Store Pipeline. In TC1.6E, loop instructions are always executed by the Load/Store pipeline.

The TC1.6E issues a single instruction per clock cycle, and as such no more than one instruction will be executed in one clock cycle.

In the execution unit all instructions pass through a decode stage followed by two execute stages. Pipeline hazards (stalls) are minimised by the use of forwarding paths between pipeline stages allowing the results of one instruction to be used by a following instruction as soon as the result becomes available.

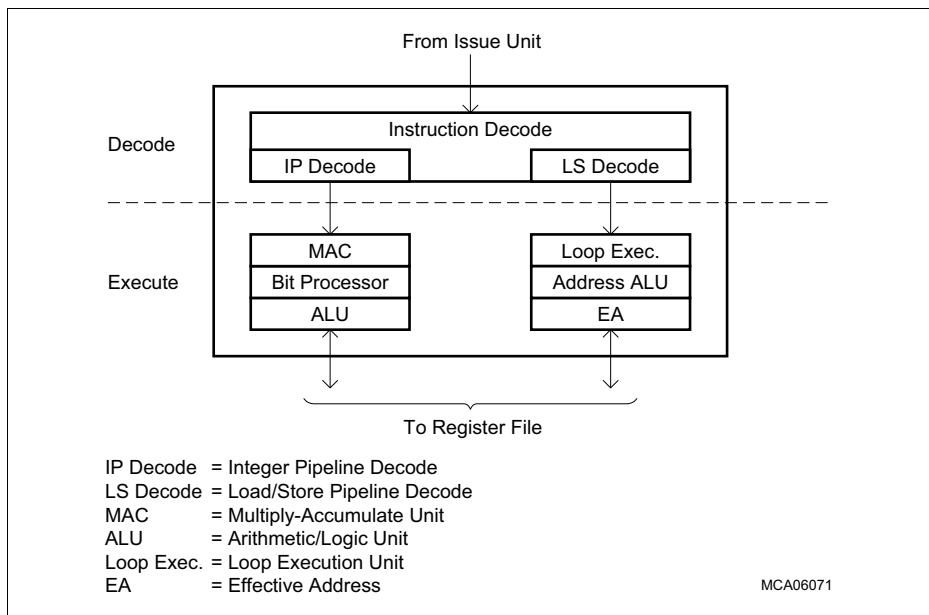


Figure 5-7 Execution Unit

5.4.4 General Purpose Register File

The CPU has a General Purpose Register (GPR) file, divided into an Address Register File (registers A0 through A15) and a Data Register File (registers D0 through D15).

The data flow for instructions issued to the Load/Store Pipeline is steered through the Address Register File.

The data flow for instructions issued to/from the Integer Pipeline and for data load/store instructions issued to the Load/Store Pipeline is steered through the Data Register File.

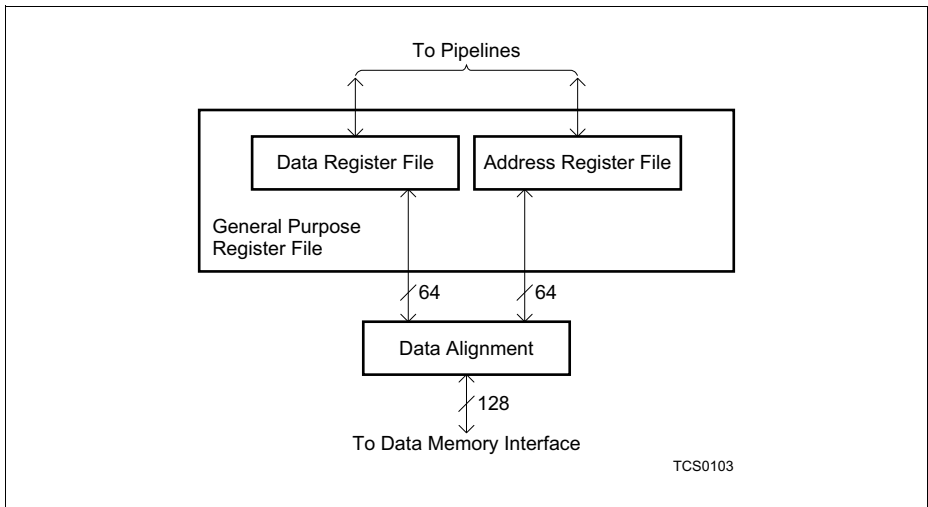


Figure 5-8 General Purpose Register File

5.5 Summary of functional changes from TC1.3.1

The TC1.6P and TC1.6E CPUs utilise different pipeline organisations than that used in the TC1.3. One effect of the new pipeline organisation is to increase the load-use penalty to 1 from 0. This necessitates re-scheduling of code to achieve optimum performance.

Other significant adaptations to the existing TC1.3.1 CPU are as follows:

- Fully Pipelined Floating Point Unit (FPU)
 - Most floating point instructions now have a repeat rate of 1
- Improved debug system - now decoupled from protection system.
 - 8 comparators providing up to 4 ranges, selectable for PC or load-store address
- Expanded and enhanced memory protection unit (MPU)
 - 16 data ranges and 8 code ranges.
- New Temporal protection system.
 - Guards against task runtime overrun.
- New Safety protection system. Tasks identified as safe by new PSW bit (PSW.S)
- New instructions for improved Interrupt and Data Cache manipulation support.
 - DISABLE, RESTORE, CACHE.I
- New instructions for Fast Integer Divide
 - DIV, DIV.U
- New Instructions for fast call and return with minimal saving of state.
 - FCALL,FCALLA,FCALLI, FRET
- Long offset addressing mode introduced for byte, half word and address accesses.
 - LD.BU, LD.B, LD.HU, LD.H, ST.B, ST.H, ST.A
- Extended range of 16 bit jumps
 - JEQ, JNE
- New Synchronisation Instructions
 - CMPSWAP.W, SWAPMSK.W
- New CRC instruction
 - CRC32
- New wait for interrupt instruction
 - WAIT
- Increased flexibility in the system address map.
- Full SECDED ECC protection for all scratch, cache and tag memory structures.
- Cache and Scratchpad memory systems now entirely separated.
 - Cache memories may be mapped as additional scratchpad.
- Selectable interrupt vector table size (32bytes/entry, 8bytes/entry).

5.6 CPU Implementation-Specific Features

This section describes the implementation-specific features of the TC1.6P and TC1.6E CPUs. For a complete description of all registers, refer to the TriCore Architecture Manual.

5.6.1 Context Save Areas / Context Operations

The CPU uses a uniform context-switching method for function calls, interrupts and traps. In all cases the Upper Context of the task is automatically saved and restored by hardware. Saving and restoring of the Lower Context may be optionally performed by software.

In TC1.6P, Context Save Areas (CSA) and addresses targeted by explicit context load/store instructions (e.g. LDLCX) may be placed in DSPR or external memory (cached or uncached).

In TC1.6E, Context Save Areas (CSA) and addresses targeted by explicit context load/store instructions may be placed in DSPR only.

CSA Placement in DSPR

The actual timing of context operations is dependent upon the placement of the Context Save Areas. Maximum performance is achieved when the Context Save Area is placed in DSPR. In this case all context save and restores operations take four cycles.

CSA Placement in Cached External Memory (TC1.6P Only)

In this case, the timing is also dependent on the state of the Data Cache. The best case Data Cache operation occurs when context saves do not incur a cache line writeback, and context restores hit in the data cache. In this case all context saves and restores take eight cycles.

5.6.2 Program Counter (PC) Register

The Program Counter (PC) holds the address of the instruction that is currently fetched and forwarded to the CPU pipelines. The CPU handles updates of the PC automatically.

Software can use the current value of the PC for various tasks, such as performing code address calculations. Reading the PC through software executed by the CPU must only be done with an MFCR instruction. Such a read will return the PC of the MFCR instruction itself. Explicit writes to the PC through an MTCR instruction must not be done due to possible unexpected behavior of the CPU. The PC may be written only when the CPU is halted.

The CPU must not perform Load/Store instructions to the mapped address of the PC in Segment 15. A MEM trap will be generated in such a case. Bit 0 of the PC register is read-only and hard-wired to 0.

5.6.3 Store Buffers

To increase performance the TC1.6P/E CPUs implements store buffering to decouple memory write operations from CPU instruction execution. All stores from the CPU are placed in the store buffer prior to being written to local memory or transferred via the bus system. Write data is taken from the store buffers and written to memory when the target memory or bus interface becomes available. In normal operation the CPU will prioritise memory load operations over store operations in order to improve performance unless:-

- The store buffer is full.
- The load is to peripheral space and a store to peripheral space exists in the store buffer. (In order peripheral space access).
- The load or store is part of an atomic operation.

Typically the operation of the store buffer is invisible to the end user. If there is a requirement that data is written to memory prior to execution of a subsequent instruction then a DSYNC instruction may be used to flush the store buffers.

To further improve performance consecutive byte writes to the same half word location are merged in the store buffer.

The TC1.6P CPU store buffer can hold the data for up to 6 stores. The TC1.6E CPU store buffer can hold the data for up to 2 stores.

Store buffer operation may be disabled by setting the SMACON.IODT bit. This should not be done in normal execution as it will severely limit performance.

5.6.4 Interrupt System

An interrupt request can be generated by the on-chip peripheral units, or it can be generated by external events. Requests can be targeted to any CPU.

The interrupt system evaluates service requests for priority and to identify whether the CPU should receive the request. The highest-priority service request is then presented to the CPU by way of an interrupt.

On taking an interrupt the CPU will vector to a unique PC generated from the interrupt priority number and the Base Interrupt Vector (BIV). The spacing between the vector PCs in the interrupt vector table may be selected to be either 32 Bytes or 8 Bytes using BIV[0].

Both the TC1.6P and TC1.6E implement a fast interrupt system. This system avoids unnecessary context save and restore operations and hence speeds up interrupt routine entry. A fast interrupt is triggered when:-

- An Interrupt is pending
- A Return From Interrupt instruction is being executed (RFE)
- The priority of the pending interrupt is greater than the priority level that would be returned to should the RFE be executed ($ICR.PIPN > PCXI.PCPN$).
- Interrupts will be enabled should the RFE be executed. ($PCXI.PIE == 1$)
- Fast Interrupts are not otherwise disabled due to the presence of MTCR instructions or context operations in the pipeline.

Without the fast interrupt operation the RFE would cause a restore of the upper context immediately followed by a save of the same upper context back to exactly the same memory location (Minimum 8 cycles). The fast interrupt system replaces this redundant restore/save sequence with a load of PCXI/PSW/A10/A11 (Minimum 1 Cycle) from the saved context. System state at the end of the fast interrupt is the same as if the standard RFE/Interrupt sequence had been performed.

5.6.5 Trap System

The following traps have implementation-specific properties.

UOPC - Unimplemented Opcode (TIN 2)

The UOPC trap is raised on optional MMU instructions, coprocessor two and coprocessor three instructions.

OPD - Invalid Operand (TIN 3)

The CPU raised OPD traps for instructions that take even-odd register pairs as an operand where if the operand specifier is odd.

DSE - Data Access Synchronous Error (TIN 2)

The Data Access Synchronous Bus Error (DSE) trap is generated by the DMI module when a load access from the CPU encounters certain error conditions, such as a Bus error, or an out-of-range access to DSPR. When a DSE trap is generated, the exact cause of the error can be determined by reading the Data Synchronous Trap Register, DSTR. For details of possible error conditions and the corresponding flag bits in DSTR, see **“DMI Trap Generation” on Page 5-103**.

DAE - Data Access Asynchronous Error (TIN 3)

The Data Access Asynchronous Error Trap (DAE) is generated by the DMI module when a store or cache management access from the CPU encounters certain error conditions, such as a Bus error. When a DAE trap is generated, the exact cause of the error can be determined by reading the Data Asynchronous Trap Register, DATR. For details of possible error conditions and the corresponding flag bits in DATR, see **“DMI Trap Generation” on Page 5-103**.

PIE Program Memory Integrity Error (TIN 5)

The PIE trap is raised whenever an uncorrectable memory integrity error is detected in an instruction fetch from a local memory or the SRI bus. The trap is synchronous to the erroneous instruction. The trap is of Class-4 and has a TIN of 5.

Program memories are protected from memory integrity errors on a 64 bit basis (TC1.6P) or 32 bit basis (TC1.6E). A PIE trap is raised when an attempt is made to execute an instruction from any fetch group containing a memory integrity error.

The PIEAR and PIETR registers may be interrogated to determine the source of any error more precisely.

DIE Data Memory Integrity Error (TIN 6)

The DIE trap is raised whenever an uncorrectable memory integrity error is detected in a data access to a local memory or the SRI bus. The trap is of Class-4 and has a TIN of 6.

DIE traps are always asynchronous independent of the operation which encountered the error.

A DIE trap is raised if any memory half word (local memory) or double word (SRI bus) accessed by a load/store operation contains an uncorrectable error. The DIEAR and DIETR registers may be interrogated to determine the source of any error more precisely.

MPX Memory Protection Execute (TIN 4)

The MPX trap is raised whenever a program attempts to execute an instruction from a memory area for which it does not have execute permission and memory protection is enabled. The trap is of Class-1 and has a TIN of 4.

The TC1.6E compares the PC of the instruction with the range(s) defined by the memory protection system to determine whether or not execution is permitted.

The TC1.6P compares the 64bit aligned fetch group address with the range(s) defined by the memory protection system to determine whether or not execution is permitted.

5.6.6 Memory Integrity Error Handling

The TriCore CPUs contain integrated support for the detection and handling of memory integrity errors. The handling of memory integrity errors for the various memory types in the CPU is as follows:

5.6.6.1 Program Side Memories

The program side memories of the CPU consist of two independent memory structures:- The Program Scratchpad RAM (PSPR) and Program Cache (PCACHE). Both memory structures are ECC protected from memory integrity errors on a 64bit basis (TC1.6P) or on a 32 bit basis (TC1.6E). Any sub-width write access to the PSPR from the Bus interface is converted to a Read-Modify-Write sequence by the PMI module.

Program Scratchpad RAM (PSPR)

The Scratchpad RAM of the CPU is protected from memory integrity errors on a 64bit basis (TC1.6P) or 32 bit basis (TC1.6E). ECC protection of is enabled via the MBIST ECCS register.

For instruction fetch requests from the TriCore CPU to PSPR, the ECC bits are read along with the data bits and are passed to the CPU along with their corresponding instructions. Whenever an attempt is made to issue an instruction containing an uncorrectable memory integrity error a synchronous PIE trap is raised. The trap handler is then responsible for correcting the memory entry and re-starting program execution.

For PSPR read operations from the Bus interface, either from the DMI module or another Bus master agent, an access that results in the detection of an uncorrectable memory integrity error in the requested data causes a bus error to be returned for the bus transaction. Since the TriCore CPU may not be involved in the transaction, a separate error is also flagged to the SCU module to optionally generate an NMI trap back to the CPU.

Writes to program scratchpad memory are only ever performed from the bus interface. For write operations less than the protection width of the PSPR the memory transaction is transformed into a read-modify-write sequence inside the PMI module. Such a write

operation may result in the detection of an uncorrectable memory integrity errors during the read phase which are handled as standard read operations.

Program Cache (PCACHE)

The Scratchpad RAM of the CPU is protected from memory integrity errors on a 64bit basis (TC1.6P) or 32 bits (TC1.6E). ECC protection of is enabled via the MBIST ECCS register.

For instruction fetch requests from the TriCore CPU to PCACHE, the ECC bits are read along with the data bits of all cache ways, and an uncorrectable error signal generated for each cache way. In the case of a tag hit, the uncorrectable error signals for the corresponding cache way are passed to the CPU along with their corresponding instructions. Whenever an attempt is made to issue an instruction containing an uncorrectable error a synchronous PIE trap is raised. The trap handler is then responsible for checking the source of the memory integrity error.

Program Tag (Ptag)

ECC protection of is enabled via the MBIST ECCS register.

For instruction fetch requests from the TriCore CPU to PCACHE, the program tag ECC bits are read along with the data bits and an error flag is computed. A way hit is triggered only if the tag address comparison succeeds, the valid bit is set and no ECC error in the associated tag way is detected, any other result is considered a miss. In the normal case where no error is detected in either cache way then the cache line is filled/refilled as normal. In the case where an error is detected the cache controller replacement algorithm forces the way indicating an error to be replaced. In the case where one cache way flags a cache hit, and another cache way detects an uncorrectable ECC error, the error condition is masked and has no effect on the memory integrity error handling mechanisms.

5.6.6.2 Data Side Memories

The TC1.6P CPU implements both data scratch and data cache memories. The TC1.6E implements a data scratch memory but replaces the data cache with a data read buffer (DRB). All data memory structures are ECC protected from memory integrity errors on a per-half word basis. Any byte write access to either DSPR or DCache (TC1.6P) is converted to a halfword Read-Modify-Write sequence. In normal operation isolated byte write transactions to the data memories result in no additional stall cycles.

Data Scratchpad Ram (DSPR)

The DSPR memory of both TC1.6P and TC1.6E are protected from memory integrity errors on a per-halfword basis. ECC protection of is enabled via the MBIST ECCS register.

CPU Subsystem

For data load requests from the TriCore CPU to DSPR, the ECC bits are read along with the data bits and an uncorrectable error signal is generated for each half-word. If an error is detected associated with any of the data half-words passed to the CPU an error is flagged to the CPU. If such an error condition is detected an asynchronous DIE trap is raised. The trap handler is then responsible for correcting the memory entry, or for taking alternative action (such as system soft reset) if correction of the data is not possible.

For DSPR read operations from the Bus interface, either from the PMI module or another Bus master agent, an access that results in the detection of an uncorrectable error in the requested data half-words causes a bus error to be returned for the bus transaction. Since the TriCore CPU may not be involved in the transaction, a separate error is also flagged to the SCU module to optionally generate an NMI trap back to the CPU.

For write operations to DSPR of half-word size or greater, the ECC bits are pre-calculated and written to the memory in parallel with the data bits. For byte write operations the memory transaction is transformed into a half-word read-modify-write sequence inside the DMI module. As such, byte write operations may result in the detection of uncorrectable memory integrity errors, which are handled as per standard read operations.

Data Cache (DCache) - TC1.6P Only

ECC protection of is enabled via the MBIST ECCS register.

For data load requests from the TriCore CPU to DCache, the ECC bits are read along with the data bits of both cache ways, and an uncorrectable error flag computed for each half-word of each cache way. In the case where an error is detected with any of the requested data half-words in a cache way which has a corresponding tag hit, an error is flagged to the CPU. If such an error condition is detected an asynchronous DIE trap is raised. The trap handler is then responsible for correcting the memory entry, or for taking alternative action (such as system soft reset) if correction of the data is not possible.

For write operations of half-word size or greater, the check bits are pre-calculated and written to the memory in parallel with the data bits. For byte write operations the memory transaction is transformed into a half-word read-modify-write sequence inside the DMI module. As such, byte write operations may result in the detection of uncorrectable memory integrity errors as for read operations.

For cache line writeback, uncorrectable error detection is performed as dirty data is transferred to the store buffers. In all cases (normal cache line eviction, cachex.xx instruction) where an error condition is detected in a valid cache line a DIE trap is raised. The trap handler is then responsible for taking corrective action (such as system soft reset) since correction of the data is not possible.

Data Tag (DTag) - TC1.6P Only

ECC protection of is enabled via the MBIST ECCS register.

For data load or store requests from the TriCore CPU to DCache, the data tag ECC bits are read along with the data bits and an uncorrectable error flag is computed. A way hit is triggered only if the tag address comparison succeeds, the tag location is valid and no uncorrectable error in the associated tag way is detected, any other result is considered a miss. In the normal case where no error is detected in either tag way then the cache line is filled/refilled as normal. In the case of a cache miss where an error is detected in one of the tag ways and the cache line does not contain dirty data the cache controller replacement algorithm forces the way indicating an error to be replaced when the refill operation returns. In the case where one cache way flags a cache hit, and the another way detects an uncorrectable error, the error condition is masked and has no effect on the memory integrity error handling mechanisms. If a cache miss occurs, with an uncorrectable error detected on the associated data tag way and dirty data detected, then an asynchronous DIE trap is signalled to the CPU and any writeback / refill sequence aborted. The trap handler is responsible for invalidating the cache line and processing any associated dirty data if possible, or taking other corrective action. Similar action is taken for forced cache writeback using the cache manipulation instructions.

5.6.6.3 Memory Initialisation

To avoid the generation of spurious ECC errors the DSPR, PSPR must be fully initialised prior to use. This may be done either by software or by automatically by hardware (see the PMU.PROCOND register for details). The entire physical memory as detailed in [Chapter 5.1](#) must be initialised irrespective of any memory size variants produced for derivative products.

5.6.7 WAIT Instruction

The WAIT instruction will suspend execution until the occurrence of one of the following events.

- Enabled Interrupt
- Non-Maskable Interrupt
- Asynchronous Trap
- Idle Request
- Suspend Request
- Asynchronous Debug Halt or Trap Request

5.6.8 Instruction Memory Range Limitations

To ensure the processor cores are provided with a constant stream of instructions the Instruction Fetch Units will speculatively fetch instructions from up to 64 bytes ahead of the current PC.

If the current PC is within 64 bytes of the top of an instruction memory the Instruction Fetch Unit may attempt to speculatively fetch instruction from beyond the physical

memory range. This may then lead to error conditions and alarms being triggered by the bus and memory systems.

It is therefore recommended that the upper 64 bytes of any memory are not used for instruction storage.

5.6.9 Atomicity of Data Accesses

The data alignment rules along with the number of bus transactions for each access type are detailed in the following tables: -

Alignment Rules

Table 5-7 Alignment rules for non-peripheral space

Access type	Access size	Alignment of address in memory	Min/Max number of SRI bus transactions
Load, Store Data Register	Byte	Byte (1 _H)	1/1
	Half-Word	2 bytes (2 _H)	1/1
	Word	2 bytes (2 _H)	1/2 *
	Double-Word	2 bytes (2 _H)	1/2 *
Load, Store Address Register	Word	4 bytes (4 _H)	1/1
	Double-Word	4 bytes (4 _H)	1/2 *
SWAP.W, LDMST, CMPSWAP.W, SWAPMSK.W	Word	4 bytes (4 _H)	1/1
ST.T	Byte	Byte (1 _H)	1/1
Context Operations	16 x 32-bit registers	64 bytes (40 _H)	2/2 *

Table 5-8 Alignment rules for peripheral space

Access type	Access size	Alignment of address in memory	Min/Max Number of SPB bus transactions	Min/Max Number of SRI bus transactions
Load, Store Data Register	Byte	Byte (1 _H)	1/1	1/1
	Half-Word	2 bytes (2 _H)	1/1	1/1
	Word	4 bytes (4 _H)	1/1	1/1
	Double-Word	8 bytes (8 _H)	1/1	1/1
Load, Store Address Register	Word	4 bytes (4 _H)	1/1	1/1
	Double-Word	8 bytes (8 _H)	1/1	1/1
SWAP.W, LDMST, CMPSWAP.W, SWAPMSK.W	Word	4 bytes (4 _H)	1/1	1/1
ST.T	Byte	Bytes (1 _H)	1/1	1/1
Context Operations	16 x 32-bit registers	Not Permitted	-	-

In the case where a single access results in a single bus transaction atomicity of the data is preserved when viewed from any bus master.

In the case where a single access leads to multiple bus transactions (marked as “*” in the above tables) then atomicity needs to be considered. In these accesses it is possible for another bus master to read or write the target memory location between the bus transactions required to complete the access.

In the case of word and double word access to the SRI bus the number of bus transactions will be 1 for naturally aligned data values and hence atomicity will be preserved.

5.6.10 A11 usage

In normal usage A11 will always contain the target of the next RET or RFE instruction. The processor uses this fact to speculatively load the return target ahead of the execution of the RET/RFE instruction. Code that modifies the A11 (e.g. test code) should be aware that any value stored in A11 may be used as the target of such speculation. If the value in A11 is not a valid address the speculation may lead to error conditions and alarms being triggered by the bus and memory systems.

It is therefore recommended that A11 should only ever contain a valid address value.

5.7 Memory Addressing

This chapter details the CPU specific addressing.

5.7.1 CSFR and SFR base Locations

Each CPU has a dedicated set of control and status registers accessed at the addresses detailed in the following table. These registers are divided into Special Function Registers (SFRs) and Core Special Function Registers (CSFRs).

A CPU must access its own CSFR registers using MTCR and MFCR instructions. CSFR registers of other CPUs may be accessed using load and store instructions via the XBAR_SRI.

SFR registers of any CPU may only be accessed using load and store instructions via XBAR_SRI. Currently the overlay control and the access protection registers of CPUx are mapped into CPUx SFR address range.

The base locations for the TC1.6P and TC1.6E SFR and CSFR registers are as follows:-

Table 5-9 CSFR and SFR Base Locations

Core-ID Value	CSFR Base Address	SFR Base Address
0	F881_0000 _H	F880_0000 _H
1	F883_0000 _H	F882_0000 _H
2	F885_0000 _H	F884_0000 _H

5.7.2 Local and Global Addressing

The TriCore architecture supports closely coupled program and data SRAM memories known as Program Scratch Pad RAM (PSPR) and Data Scratch Pad RAM (DSPR). The local PSPR memory is always located at C0000000_H. The local DSPR is always located at D0000000_H. In a multiprocessor system the local scratch pad memories appear in the global address map in the following locations:-

Table 5-10 Global Address Locations

Core-ID Value	PSPR_base	DSPR_base
0	70100000 _H	70000000 _H
1	60100000 _H	60000000 _H
2	50100000 _H	50000000 _H

The CPUs always use the global addresses for bus transactions. Thus a data load from C0000000_H (a CPU's local PSPR) will result in a bus transaction with an address in the range 50100000_H - 701FFFFF_H dependent on the Core-ID value of the processor.

CPU Subsystem

Similarly a code fetch from $D0000000_H$ (a CPU's local DSPR) will result in a bus transaction with an address in the range $50000000_H - 700FFFFFF_H$ dependent on the Core-ID value of the processor.

5.7.3 Cache Memory Access

The cache and tag memories may be mapped into the CPU's address space. When mapped the cache memories are contiguous with the DSPR/PSPR memories as detailed in the following table. When mapped the cache memories behave identically to the PSPR/DSPR memories and may be used as standard memory.

The mapping of cache and tag memories to the TriCore address space is controlled by the MTU_MEMMAP register. See the MTU chapter for details.

Table 5-11 Cache Memory Locations when mapped

Memory	Local Address	Global Address
Program Cache	$C000_0000_H +$ $PSPR_Memory_Size$	$PSPR_Base +$ $PSPR_Memory_Size$
Data Cache (TC1.6P)	$D000_0000_H +$ $DSPR_Memory_Size$	$DSPR_Base +$ $DSPR_Memory_Size$

When mapped the tag memories are available at the locations detailed in the following table. The mapping of the Tag memories is provided test purposes only. They may not be used as standard memory.

Table 5-12 Tag Memory Locations when mapped

Memory	Local Address	Global Address
Program Tag	$C00C_0000_H$	$PSPR_Base + 0x000C_0000_H$
Data Tag (TC1.6P)	$D00C_0000_H$	$DSPR_Base + 0x000C_0000_H$

The CACHEI.* instructions require a way and index value to be supplied in a valid address. The location of these bits in the 32bit address is as follows.

Table 5-13 Way and Index Location

Function	Address Bits
Way	[0]
Index	[11:5]

5.8 CPU Subsystem Registers

This section describes the implementation-specific features of the CPU Subsystem registers listed in [Table 5-14](#). For complete descriptions of all registers refer to the TriCore Architecture Manual.

Table 5-14 CPU Subsystem Registers

Registers	Purpose	Description
CPU Core Special Function Registers (CSFRs)	Program state information, context and stack management, interrupt and trap control, system control	see Page 5-28
CPU General Purpose Registers (GPRs)	General Purpose Address and Data Registers	see Page 5-40
CPU Memory Protection Registers (CSFRs)	Memory protection control and mode selection	see Page 5-40
FPU Registers (CSFRs)	Support for the standard floating point instructions.	see Page 5-41
Memory Integrity Registers (CSFRs)	Integrity and Protection Core Special Function Registers.	see Page 5-42
Core Debug Registers (CSFRs)	Debug control	see Page 5-53
Implementation Specific Reset Values	Reset values for CPU registers not defined in this chapter	see Page 5-56
Program Memory Interface Registers (PMI CSFRs)	PMI program cache control, status and trap information	see Page 5-93
Data Memory Interface Registers (DMI CSFRs)	DMI data cache control, status and trap information	see Page 5-105

Note: To increase performance all stores to CSFR or SFR register locations are posted writes and complete silently. Stores to non-existent CSFR or SFR locations are not errored.

5.8.1 CPU Core Special Function Registers (CSFR)

This section describes implementation specific features of the Core Special Function Registers.

5.8.1.1 Registers

The implementation-specific Program Status Word Register (PSW) is an extension of the PSW description in the TriCore Architecture Manual. The status flags used for FPU operations overlay the status flags used for Arithmetic Logic Unit (ALU) operations.

Program Status Word Register

Note: The non-shaded areas in the register description define the implementation-specific bits/bit fields. The shaded areas are defined in the TriCore Architecture Manual.

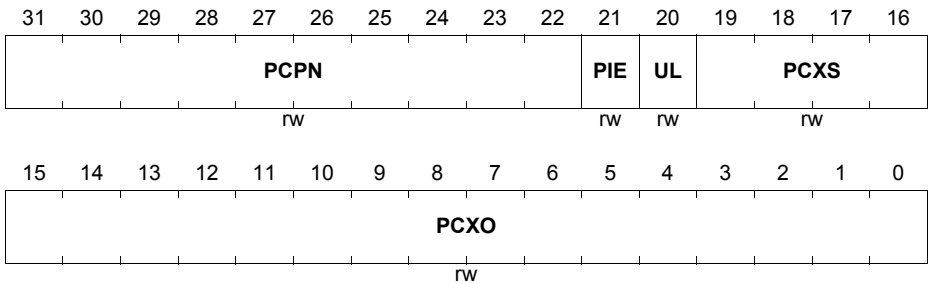
PSW

Program Status Word Register (CSFR_Base + FE04_H) Reset Value: 0000 0B80_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C or FS	V or FI	SV or FV	AV or FZ	SAV or FU	FX	RM		0							
rwh	rwh	rwh	rwh	rwh	rwh	rw		r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		S	PRS		IO		IS	GW	CDE	CDC					
r		rwh	rwh		rwh		rwh	rwh	rwh	rwh					

Field	Bits	Type	Description
RM	[25:24]	rw	FPU Rounding Mode Selection
FX	26	rwh	FPU Inexact Flag
SAV	27	rh	Sticky Advance Overflow Flag
FU		rwh	FPU Underflow Flag
AV	28	rwh	Advance Overflow Flag
FZ			FPU Divide by Zero Flag
SV	29	rwh	Sticky Overflow Flag
FV			FPU Overflow Flag

Field	Bits	Type	Description
V	30	rwh	Overflow Flag
FI			FPU Invalid Operation Flag
C	31	rwh	Carry Flag
FS			FPU Some Exception Flag

Previous Context Information Register
PCXI
Previous Context Information Register
(CSFR_Base + FE00_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
PCXO	[15:0]	rw	Previous Context Pointer Offset Field The PCXO and PCXS fields form the pointer PCX, which points to the CSA of the previous context.
PCXS	[19:16]	rw	Previous Context Pointer Segment Address Contains the segment address portion of the PCX. This field is used in conjunction with the PCXO field.
UL	20	rw	Upper or Lower Context Tag Identifies the type of context saved. If the type does not match the type expected when a context restore operation is performed, a trap is generated. 0 _B Lower Context 1 _B Upper Context
PIE	21	rw	Previous Interrupt Enable Indicates the state of the interrupt enable bit (ICR.IE) for the interrupted task.
PCPN	[31:22]	rw	Previous CPU Priority Number Contains the priority level number of the interrupted task.

Address Space Identifier Register (TASK_ASI)

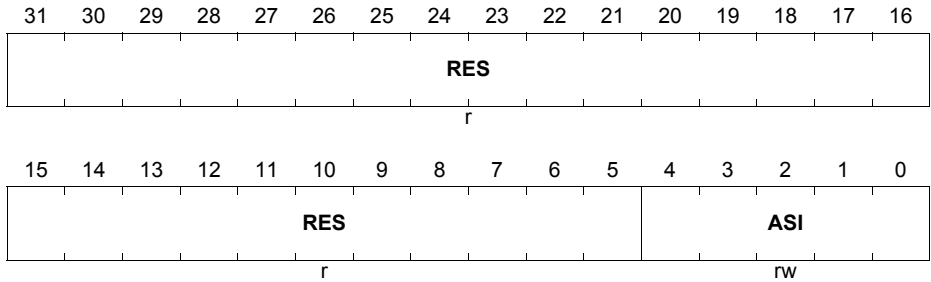
The Task Address Space Identifier (ASI) register description.

TASK_ASI

Task Address Space Identifier Register

(CSFR_Base + 8004_H)

Reset Value: 0000 001F_H



Field	Bits	Type	Description
RES	[31:5]	r	Reserved
ASI	[4:0]	rw	Address Space Identifier The ASI register contains the Address Space Identifier of the current process.

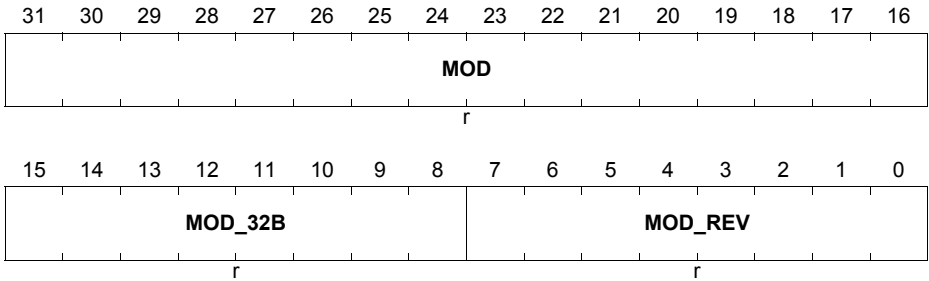
CPU Identification Register (TC1.6P)

CPU_ID

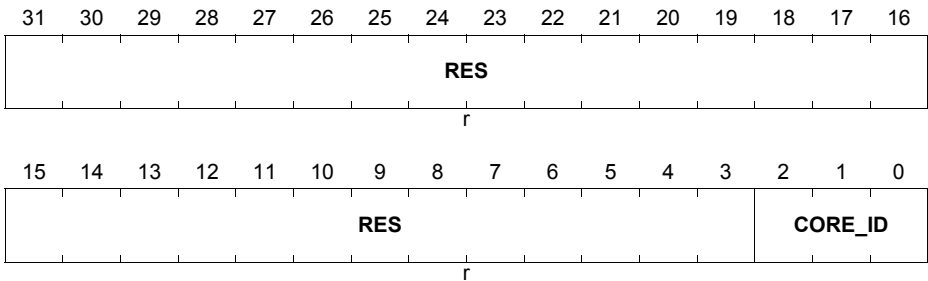
CPU Identification Register

(CSFR_Base + FE18_H)

Reset Value: 00C0 C012_H



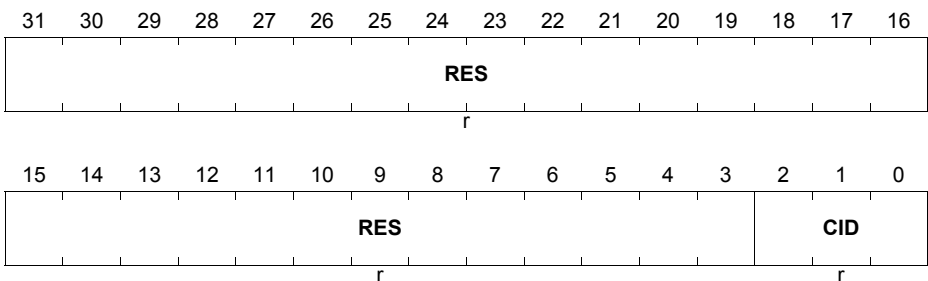
Field	Bits	Type	Description
MOD_REV	[7:0]	r	Revision Number 12 _H Reset value
MOD_32B	[15:8]	r	32-Bit Module Enable C0 _H A value of C0 _H in this field indicates a 32-bit module with a 32-bit module ID register.
MOD	[31:16]	r	Module Identification Number 00C0 _H For module identification

Core Identification Register
CORE_ID
Core Identification Register
(CSFR_Base + FE1C_H)
Reset Value: 0000 000X_H


Field	Bits	Type	Description
RES	[31:3]	r	Reserved
CORE_ID	[2:0]	r	Core Identification Number The identification number of the core.

Customer ID Register

In circumstances where the Infineon defined Core identification numbering scheme is insufficient or incompatible with a customer numbering scheme a customer ID value may be supplied via the CUS_ID register.

CUS_ID
Customer ID register
(CSFR_Base + FE50_H)
Reset Value: 0000 000X


Field	Bits	Type	Description
RES	[31:3]	r	Reserved
CID	[2:0]	r	Customer ID

Physical Memory Attributes Registers

The Physical Memory Attributes registers (PMA0,PMA1,PMA2) define the physical memory attribute for each segment in the physical address space. The register is ENDINIT protected and can be read with the MFCR instruction and written by the MTCR instruction. Note that when changing the value of the registers both the instruction and data caches should be invalidated, a DSYNC instruction should be executed immediately prior to the MTCR with an ISYNC instruction executed immediately following. This is required to maintain coherency of the processors view of memory.

The register PMA0 defines the data access cacheability of the segment in the physical address space. If bit n in the register is set then segment-n will be seen as cacheable for data accesses.

The register PMA1 defines the code access cacheability of the segment in the physical address space. If bit n in the register is set then segment-n will be seen as cacheable for code accesses.

The register PMA2 defines the peripheral space identifier of the segment in the physical address space. If bit n in the register is set then segment-n will be seen as a peripheral segment. PMA2 is a read-only register.

Registers PMA0 and PMA1 are freely programmable with the following restrictions:-

- In PMA0 Segment-C and Segment[7-CoreID] must have the same value.
- In PMA1 Segment-D and Segment[7-CoreID] must have the same value.

Irrespective of the setting of the PMA registers the following constraints are always enforced.

- Segments-F and segment-E are constrained to be Peripheral space and hence non-cacheable.
- Segment-A is constrained to be non-cacheable memory
- Segment-D and Segment[7-CoreID] are constrained to be non-cacheable for data accesses.
- Segment-C and Segment[7- CoreID] are constrained to be non-cacheable for code accesses.

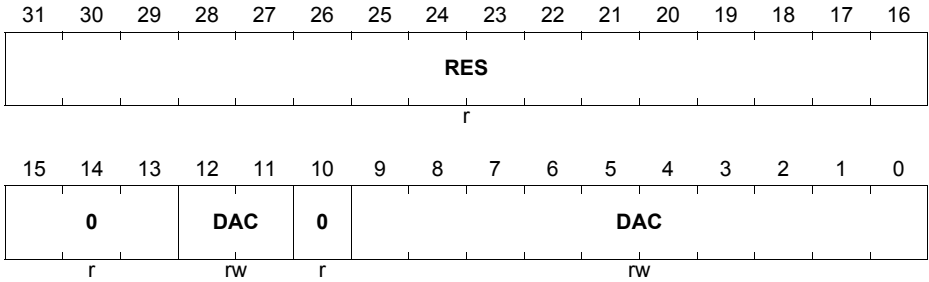
Note: The PMA registers are ENDINIT protected.

PMA0

Data Access Cacheability Register

(CSFR_Base + 8100_H)

Reset Value: 0000 0300_H



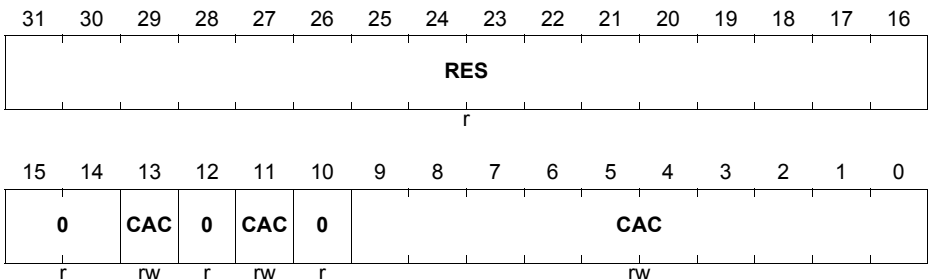
Field	Bits	Type	Description
RES	[31:16]	r	Reserved
DAC	[15:13]	r	Data Access Cacheability Segments F _H , E _H , D _H (non-cacheable)
DAC	[12:11]	rw	Data Access Cacheability Segments C _H , B _H
DAC	10	r	Data Access Cacheability Segments A _H (non-cacheable)
DAC	[9:0]	rw	Data Access Cacheability Segments 9 _H -0 _H

PMA1

Code Access Cacheability Register

(CSFR_Base + 8104_H)

Reset Value: 0000 0300_H



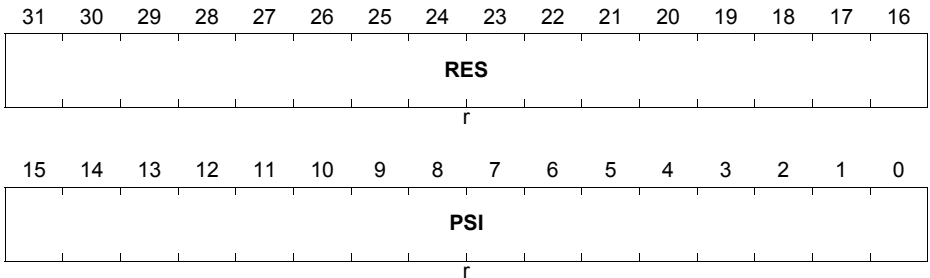
Field	Bits	Type	Description
RES	[31:16]	r	Reserved
CAC	[15:14]	r	Code Access Cacheability Segments F_H,E_H (non-cacheable)
CAC	13	rw	Code Access Cacheability Segments D_H
CAC	12	r	Code Access Cacheability Segments C_H (non-cacheable)
CAC	11	rw	Code Access Cacheability Segments B_H
CAC	10	r	Code Access Cacheability Segments A_H (non-cacheable)
CAC	[9:0]	rw	Code Access Cacheability Segments 9_H-0_H

PMA2

Peripheral Space Identifier Register

(CSFR_Base + 8108_H)

Reset Value: 0000 C000_H



Field	Bits	Type	Description
RES	[31:16]	r	Reserved
PSI	[15:0]	r	Peripheral Attribute Segments F_H-0_H = C000 _H

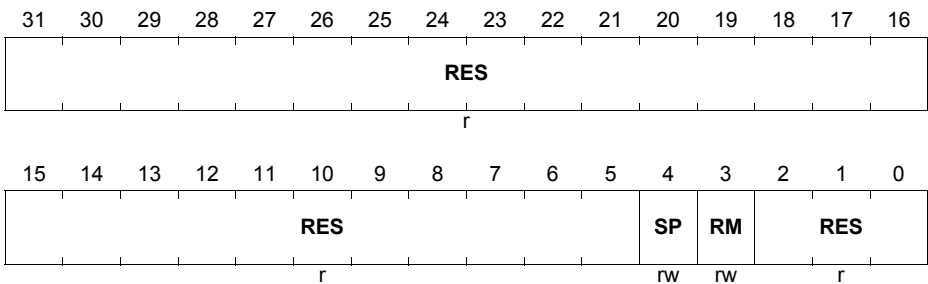
Compatibility Control Register

The Compatibility Control Register (COMPAT) is an implementation-specific CSFR which allows certain elements of backwards compatibility with TriCore 1.3.x behavior to be forced. The reset value of the COMPAT register ensures that backwards compatibility with TriCore 1.3 is enabled by default. This register is safety_endinit protected.

COMPAT

Compatibility Control Register

(CSFR_Base + 9400_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
RES	[2:0]	r	Reserved Read as 1; should be written with 1.
RM	3	rw	Rounding Mode Compatibility 0 _B PSW.RM not restored by RET. 1 _B PSW.RM restored by RET (TC1.3 behavior).
SP	4	rw	SYSCON Safety Protection Mode Compatibility 0 _B SYSCON[31:1] safety endinit protected. 1 _B SYSCON[31:1] not safety endinit protected (TC1.3 behavior).
RES	[31:5]	r	Reserved Read as 1; should be written with 1.

5.8.2 CPU General Purpose Registers

There are no implementation specific features of the General Purpose Registers. They are described in detail in the TriCore Architecture Manual.

5.8.3 CPU Memory Protection Registers

There are four Memory Protection Register Sets per CPU. The sets specify memory protection ranges and permissions for code and data. The PSW.PRS bit field determines which of these sets is currently in use by the CPU. The CPUs each implement 16 data and 8 code range comparators. These may be flexibly shared amongst the of protection sets to provide a maximum of 16 data ranges and 8 code ranges per set.

The protection registers protect the whole address space (In previous TriCore implementations the memory protection system did not cover peripheral space.) The granularity of the protection ranges is 8 bytes.

Code protection ranges define which memory areas the CPU may fetch instructions from. Instruction fetches from an area outside a valid code protection range will lead to a trap condition.

Data protection ranges define which memory areas the CPU may access for read and/or write operations. Each range may be separately enabled for read and/or write access. Data read accesses from an area outside a valid data protection range with read permissions will lead to a trap condition. Data write accesses to an area outside a valid data protection range with write permissions will lead to a trap condition

There are no implementation specific features of the Memory Protection Registers. They are described in detail in the TriCore Architecture Manual.

5.8.4 Temporal Protection Registers

To Guard against task runtime overrun the CPU implements a temporal protection system. This system consists of three independent decrementing counters (TPS_TIMERn) arranged to generate a Temporal Asynchronous Error trap (TAE - Class-4, Tin-7) on decrement to zero. A control register (TPS_CON) contains status and control bits for temporal protection system. The temporal protection system is enabled by the SYSCON.TPROTEN register bit. The Temporal Protection Registers are Core Special Function Registers. They are described in detail in the TriCore Architecture Manual.

There are no implementation specific features of the Memory Protection Registers. They are described in detail in the TriCore Architecture Manual.

5.8.5 FPU Registers

There are no implementation specific features of the FPU Registers. They are described in detail in the TriCore Architecture Manual.

5.8.6 Memory Integrity Registers

To monitor and debug the integrity of the memory subsystems the following registers are provided.

Table 5-15 Memory Integrity Registers

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
PIEAR	Program Integrity Error Address Register	9210 _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
PIETR	Program Integrity Error Trap Register	9214 _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
DIEAR	Data Integrity Error Address Register	9020 _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
DIETR	Data Integrity Error Trap Register	9024 _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
SMACON	SIST Mode Access Control Register	900C _H	U, SV, 32	SV, SE, 32, P	Application 0000 0000 _H

5.8.6.1 Register Descriptions

Program Integrity Error Information Registers

Two architecturally visible registers (PIETR, PIEAR) allow software to localise the source of the last detected program memory integrity error.

These registers are updated when a program integrity error condition is detected and the PIETR.IED bit is zero. On update the PIETR.IED bit is set to one and remains set until cleared by software. Whilst PIETR.IED is set further hardware updates of PIETR and PIEAR are inhibited. The various error scenarios are defined below.

Program Integrity errors during instruction fetch from program memory

When an error is detected during a program fetch from a program memory the IE_S, IE_C bits are updated to denote in which memory structure the error was detected. If the IE_C bit is set the E_INFO field will be updated with the cache way. The PIEAR register is updated with the address of the access. If the error detected is an uncorrectable bit error the IE_UNC bit is set. The IED bit is set and all other PIETR register bits are set to zero. Since instruction fetches are speculative, the PIETR and PIEAR registers may be updated without a corresponding PIE trap

Program Integrity errors during a memory read initiated by an external access

When an error is detected during an external bus read from program memory the IE_S, IE_C bits are updated to denote in which memory structure the error was detected. The IE_BS is set and E_INFO updated. The PIEAR register is updated with the address of the access. If the error detected is an uncorrectable error the IE_UNC bit is set. The IED bit is set and all other PIETR register bits are set to zero. Note that a memory read can be generated by a sub word write operation. The IE_C bit can only ever be set if the cache is mapped into memory in SIST mode. In this case the E_INFO field indicates the tag of the requesting master.

Program Integrity errors due to safety protection

When a safety protection violation is detected during a program memory access or during access to the CPU registers the IE_SP, IE_BS bits are set and the E_INFO field updated. The PIEAR register is updated with the address of the access. The IED bit is set and all other PIETR register bits are set to zero.

Program Integrity errors due ECC errors at the bus interface

When an ECC error is detected at the program bus interface (either master or slave) the IE_BI bit is set. If the error is during an external access the IE_BS bit is set and the E_INFO fields updated. The PIEAR register is updated with the address of the access.

If the error detected is an uncorrectable error the IE_UNC bit is set. The IED bit is set and all other PIETR register bits are set to zero.

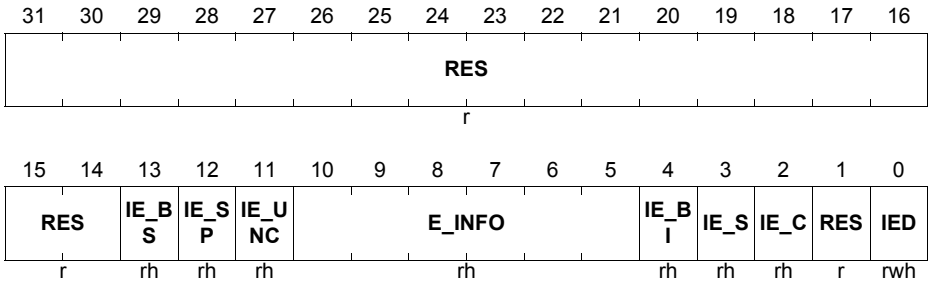
Program Integrity Error Trap Register (PIETR)

PIETR

Program Integrity Error Trap Register

(CSFR_Base + 9214_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
IED	0	rwh	Integrity Error Detected Read Operation: 0 _B No program integrity error condition occurred. 1 _B Program integrity error condition detected. PIETR and PIEAR contents valid, further updates disabled. Write Operation: 0 _B Clear IED bit, re-enable updates. 1 _B No effect.
RES	1	r	Reserved
IE_C	2	rh	Integrity Error - Cache Memory
IE_S	3	rh	Integrity Error - Scratchpad Memory
IE_BI	4	rh	Integrity Error - Bus Interface
E_INFO	[10:5]	rh	Error Information If IE_BS= 1: Bus Master Tag ID of requesting master If IE_C = 1: Cache way.
IE_UNC	11	rh	Integrity Error - Uncorrectable Error Detected
IE_SP	12	rh	Safety Protection Error Detected
IE_BS	13	rh	Bus Slave Access Indicator
RES	[31:14]	r	Reserved Read as 0; should be written with 0.

Program Integrity Error Address Register

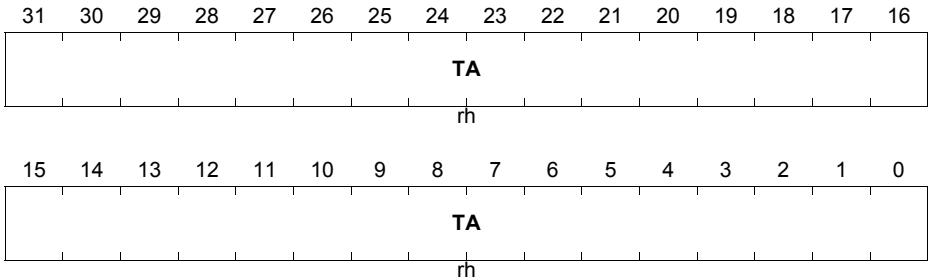
This register contains the physical address accessed by the operation that encountered a uncorrectable program memory integrity error. This register is only updated if PIETR.IED is zero.

PIEAR

Program Integrity Error Address Register

(CSFR_Base + 9210_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TA	[31:0]	rh	Transaction Address Physical address being accessed by operation that encountered program integrity error.

Data Integrity Error Information Registers

Two architecturally visible registers (DIETR, DIEAR) allow software to localise the source of the last detected uncorrectable data memory integrity error.

These registers are updated when an uncorrectable data integrity error condition is detected and the DIETR.IED bit is zero. On update the DIETR.IED bit is set to one and remains set until cleared by software. Whilst DIETR.IED is set further hardware updates of DIETR and DIEAR are inhibited. The various error scenarios are defined below.

Note that the TC1.6E has no data cache but will report Data Read Buffer errors as cache errors from way0.

Data Integrity errors during load from data memory

When an error is detected during a load from a data memory the IE_S, IE_C and IE_T bits are updated to denote in which memory structure the error was detected. If the IE_C bit is set the E_INFO field will be updated with the cache way. The DIEAR register is updated with the address of the access. If the error detected is an uncorrectable error the IE_UNC bit is set. The IED bit is set and all other DIETR register bits are set to zero. Note that a memory read may also be generated by a sub word write operation or a cache line eviction.

Data Integrity errors during a memory read initiated by an external access

When an error is detected during an external bus read from data memory the IE_S, IE_C bits are updated to denote in which memory structure the error was detected. The IE_BS is set and E_INFO updated. The DIEAR register is updated with the address of the access. If the error detected is an uncorrectable error the IE_UNC bit is set. The IED bit is set and all other DIETR register bits are set to zero. Note that a memory read may also be generated by a sub word write operation or a cache line eviction. The IE_C bit can only ever be set if the cache is mapped into memory in SIST mode. In this case the E_INFO field indicates the tag of the requesting master.

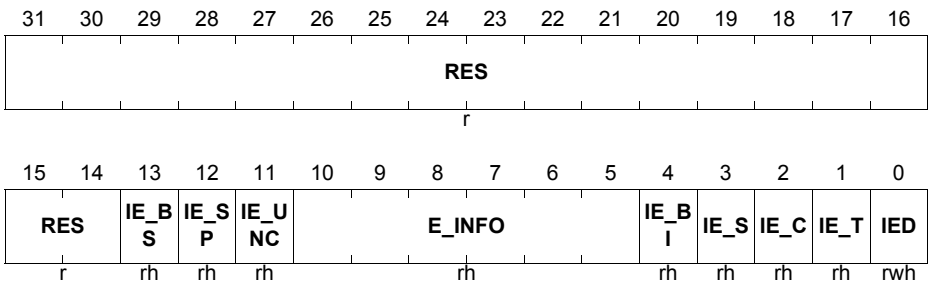
Data Integrity errors due to safety protection

When a safety protection violation is detected during a data memory access the IE_SP, IE_BS bits are set and the E_INFO field updated. The DIEAR register is updated with the address of the access. The IED bit is set and all other DIETR register bits are set to zero.

Data Integrity errors due ECC errors at the bus interface

When an ECC error is detected at the data bus interface (either master or slave) the IE_BI bit is set. If the error is during an external access the IE_BS bit is set and the E_INFO fields updated. The DIEAR register is updated with the address of the access.

If the error detected is an uncorrectable error the IE_UNC bit is set. The IED bit is set and all other DIETR register bits are set to zero.

Data Integrity Error Trap Register (DIETR)
DIETR
Data Integrity Error Trap Register
(CSFR_Base + 9024_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
IED	0	rwh	Integrity Error Detected Read Operation: 0 _B No data integrity error condition occurred. 1 _B Data integrity error condition detected. DIETR and DIEAR contents valid, further DIETR and DIEAR updates disabled. Write Operation: 0 _B Clear IED bit, re-enable DIETR and DIEAR update. 1 _B No effect.
IE_T	1	rh	Integrity Error - Tag Memory
IE_C	2	rh	Integrity Error - Cache Memory
IE_S	3	rh	Integrity Error - Scratchpad Memory
IE_BI	4	rh	Integrity Error - Bus Interface
E_INFO	[10:5]	rh	Error Information If IE_BS = 1: Bus Master Tag ID of requesting master If IE_C = 1: Cache way.
IE_UNC	11	rh	Dual Bit Error Detected
IE_SP	12	rh	Safety Protection Error Detected

Field	Bits	Type	Description
IE_BS	13	rh	Bus Slave Access Indicator
RES	[31:14]	r	Reserved Read as 0; should be written with 0.

Data Integrity Error Address Register

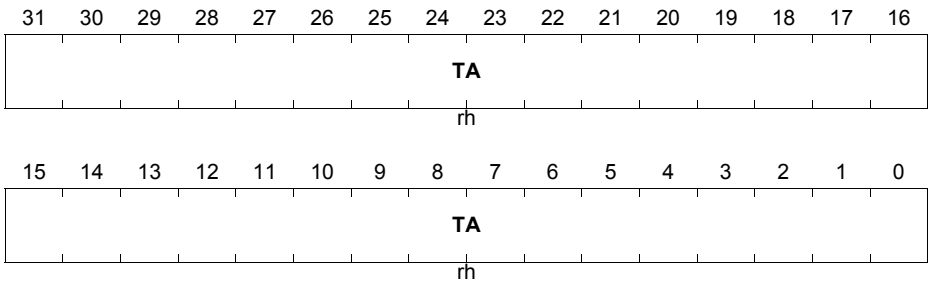
This register contains the physical address accessed by the operation that encountered a uncorrectable data memory integrity error. This register is only updated if DIETR.IED is zero.

DIEAR

Data Integrity Error Address Register

(CSFR_Base + 9020_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TA	[31:0]	rh	Transaction Address Physical address being accessed by operation that encountered data integrity error.

SIST (Software In-System) Test Support

The CPU protects against memory integrity errors by ECC protection of the local CPU memories. This has the side-effect of requiring memory blocks wider than the normal data access path to the memory. The additional ECC storage bits are not easily accessible via the existing data paths, causing problems where software based testing of the memories is required. The CPU memories also include embedded memory arrays, such as the tag memories, which are not ordinarily accessible by the usual CPU datapaths. In order to address this problem, the CPUs include a modes allowing all local memory arrays to be accessed, both as a backup for MBIST based memory test and to allow the test and debug of the fault tolerant memory systems.

Each memory may be access either in normal operation, data array only or ECC check array only. This is controlled by the ECCS registers of the associated MBIST controller.

The IODT bit controls read/write operation ordering. In normal operation (IODT=0) non-dependent read operations may overtake write operations. When SMA.IODT is set all memory operations are performed in program order.

The SMACON register is protected by the safety_endinit signal.

The mapping of cache and tag memories to the TriCore address space is controlled by the MTU_MEMMAP register. See the MTU chapter for details.

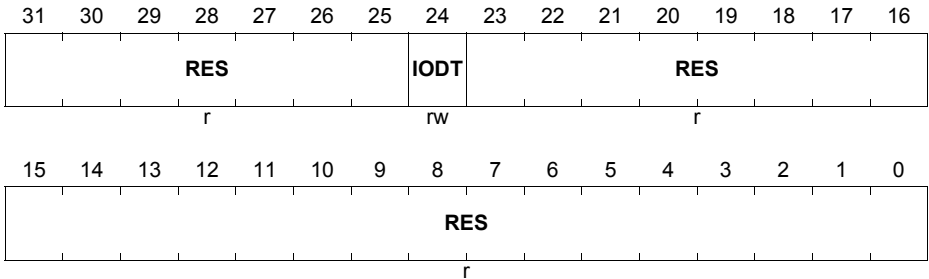
SIST Mode Access Control Register

SMACON

SIST Mode Access Control Register

(CSFR_Base + 900C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RES	[31:25]	r	Reserved Read as 0; should be written with 0.
IODT	24	rw	In-Order Data Transactions 0 _B Normal operation, Non-dependent loads bypass stores. 1 _B In-order operation, Loads always flush preceding stores, processor store buffer disabled.
RES	[23:0]	r	Reserved Read as 0; should be written with 0.

5.8.7 CPU Core Debug and Performance Counter Registers

The CPU Core Debug and performance counter registers are available for debug purposes. For a complete description of all registers, refer to the TriCore Architecture Manual.

5.8.7.1 Counter Source Details

Details of the Performance counter sources is given below.

IP_DISPATCH_STALL

The counter is incremented on every cycle in which the Integer dispatch unit is stalled for whatever reason.

LS_DISPATCH_STALL

The counter is incremented on every cycle in which the Load-Store dispatch unit is stalled for whatever reason.

LP_DISPATCH_STALL

The counter is incremented on every cycle in which the Loop dispatch unit is stalled for whatever reason.

PCACHE_HIT

The counter is incremented whenever the target of a cached fetch request from the fetch unit is found in the program cache.

PCACHE_MISS

The counter is incremented whenever the target of a cached fetch request from the fetch unit is not found in the program cache and hence a bus fetch is initiated.

MULTI_ISSUE

The counter is incremented in any cycle where more than one instruction is issued.

DCACHE_HIT

The counter is incremented whenever the target of a cached request from the Load-Store unit is found in the data cache.

DRB_HIT

The counter is incremented whenever the target of a cached request from the Load-Store unit is found in the data read buffer.

DCACHE_MISS_CLEAN

The counter is incremented whenever the target of a cached request from the Load-Store unit is not found in the data cache and hence a bus fetch is initiated with no dirty cache line eviction.

DRB_MISS

The counter is incremented whenever the target of a cached request from the Load-Store unit is not found in the data read buffer and hence a bus fetch is initiated.

DCACHE_MISS_DIRTY

The counter is incremented whenever the target of a cached request from the Load-Store unit is not found in the data cache and hence a bus fetch is initiated with the writeback of a dirty cache line.

TOTAL_BRANCH

The counter is incremented in any cycle in which a branch instruction is in a branch resolution stage of the pipeline (IP_EX1, LS_DEC, LP_DEC).

PMEM_STALL

The counter is incremented whenever the fetch unit is requesting an instruction and the Instruction memory is stalled for whatever reason.

DMEM_STALL

The counter is incremented whenever the Load-Store unit is requesting a data operation and the data memory is stalled for whatever reason.

Performance Counter Registers
Table 5-16 OCDS Control Registers

Register	Description	Offset Address
CCTRL	Counter Control Register.	FC00 _H
CCNT	CPU Clock Count Register.	FC04 _H
ICNT	Instruction Count Register.	FC08 _H
M1CNT	Multi Count Register 1.	FC0C _H
M2CNT	Multi Count Register 2.	FC10 _H

Table 5-16 OCDS Control Registers (cont'd)

Register	Description	Offset Address
M3CNT	Multi Count Register 3.	FC14 _H

Table 5-17 MultiCount Configuration (TC1.6P)

CFG	M1CNT	M2CNT	M3CNT
000	IP_DISPATCH_STALL	LS_DISPATCH_STALL	LP_DISPATCH_STALL
001	PCACHE_HIT	PCACHE_MISS	MULTI_ISSUE
010	DCACHE_HIT	DCACHE_MISS_CLEAN	DCACHE_MISS_DIRTY
011	TOTAL_BRANCH	PMEM_STALL	DMEM_STALL

Table 5-18 MultiCount Configuration (TC1.6E)

CFG	M1CNT	M2CNT	M3CNT
000	IP_DISPATCH_STALL	LS_DISPATCH_STALL	-
001	PCACHE_HIT	PCACHE_MISS	-
010	DRB_HIT	DRB_MISS	-
011	TOTAL_BRANCH	PMEM_STALL	DMEM_STALL

5.8.8 Summary of CSFR Reset Values and Access Modes

This section summarizes the reset values and access modes of the CPU CSFR registers.

Table 5-19 Core Special I Function Registers

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
TASK_ASI	TASK Address Space Identifier	8004 _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
PCXI	Previous Context Information Register	FE00 _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
PSW	Program Status Word Register	FE04 _H	U, SV, 32	SV, 32, P	Application 0000 0B80 _H
PC	Program Counter Register	FE08 _H	U, SV, 32	SV, 32, P	Application XXXX XXXX _H
SYSCON	System Configuration Register	FE14 _H	U, SV, 32	SV, 32, P, SE	Application 0000 0000 _H
CPU_ID (TC1.6P)	CPU Identification Register	FE18 _H	U, SV, 32	SV, 32, P, NC	Application 00C0 C012 _H
CPU_ID (TC1.6E)	CPU Identification Register	FE18 _H	U, SV, 32	SV, 32, P, NC	Application 00B7 C002 _H
CORE_ID	Core Identification Register	FE1C _H	U, SV, 32	SV 32, P, NC	Application 00000 000X _H
BIV	Interrupt Vector Table Pointer Register	FE20 _H	U, SV, 32	SV, 32, P, CEx	Application 0000 0000 _H
BTV	Trap Vector Table Pointer Register	FE24 _H	U, SV, 32	SV, 32, P, CEx	Application A000 0100 _H
ISP	Interrupt Stack Pointer Register	FE28 _H	U, SV, 32	SV, 32, P, CEx	Application 0000 0100 _H
ICR	ICU Interrupt Control Register	FE2C _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
FCX	Free Context List Head Pointer Register	FE38 _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
LCX	Free Context List Limit Pointer Register	FE3C _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
PMA0	Physical Memory Attributes Register 0	8100 _H	U, SV, 32	SV, 32, P, CEx	Application 0000 0300 _H

Table 5-19 Core Special (cont'd)I Function Registers

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
PMA1	Physical Memory Attributes Register 1	8104 _H	U, SV, 32	SV,32, P, CEx	Application 0000 0300 _H
PMA2	Physical Memory Attributes Register 2	8108 _H	U, SV, 32	SV,32, P, NC	Application 0000 C000 _H
SMACON	SIST Mode Access Control register	900C _H	U, SV, 32	SV,32, P, SE	Application 0000 0000 _H
COMPAT	Compatibility Control Register	9400 _H	U, SV, 32	SV,32 P, SE	Application FFFF FFFF _H

Table 5-20 GPR Registers

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
D0	Data Register 0	FF00 _H	U, SV, 32	SV,32 P	Application XXXX XXXX _H
D1	Data Register 1	FF04 _H	U, SV, 32	SV,32 P	Application XXXX XXXX _H
D2	Data Register 2	FF08 _H	U, SV, 32	SV,32 P	Application XXXX XXXX _H
D3	Data Register 3	FF0C _H	U, SV, 32	SV,32 P	Application XXXX XXXX _H
D4	Data Register 4	FF10 _H	U, SV, 32	SV,32 P	Application XXXX XXXX _H
D5	Data Register 5	FF14 _H	U, SV, 32	SV,32 P	Application XXXX XXXX _H
D6	Data Register 6	FF18 _H	U, SV, 32	SV,32 P	Application XXXX XXXX _H
D7	Data Register 7	FF1C _H	U, SV, 32	SV,32 P	Application XXXX XXXX _H
D8	Data Register 8	FF20 _H	U, SV, 32	SV,32 P	Application XXXX XXXX _H
D9	Data Register 9	FF24 _H	U, SV, 32	SV,32 P	Application XXXX XXXX _H

Table 5-20 GPR Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
D10	Data Register 10	FF28 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
D11	Data Register 11	FF2C _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
D12	Data Register 12	FF30 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
D13	Data Register 13	FF34 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
D14	Data Register 14	FF38 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
D15	Data Register 15	FF3C _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A0	Address Register 0 (Global Address Register)	FF80 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A1	Address Register 1 (Global Address Register)	FF84 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A2	Address Register 2	FF88 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A3	Address Register 3	FF8C _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A4	Address Register 4	FF90 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A5	Address Register 5	FF94 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A6	Address Register 6	FF98 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A7	Address Register 7	FF9C _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A8	Address Register 8 (Global Address Register)	FFA0 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A9	Address Register 9 (Global Address Register)	FFA4 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H

Table 5-20 GPR Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
A10	Address Register 10 (Stack Pointer)	FFA8 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A11	Address Register 11 (Return Address)	FFAC _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A12	Address Register 12	FFB0 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A13	Address Register 13	FFB4 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A14	Address Register 14	FFB8 _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H
A15	Address Register 15	FFBC _H	U, SV, 32	SV, 32 P	Application XXXX XXXX _H

Table 5-21 Memory Protection Registers

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
DPR0_L	Data Protection Range 0, Lower Bound Register	C000 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR0_U	Data Protection Range 0, Upper Bound Register	C004 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR1_L	Data Protection Range 1, Lower Bound Register	C008 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR1_U	Data Protection Range 1, Upper Bound Register	C00C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR2_L	Data Protection Range 2, Lower Bound Register	C010 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR2_U	Data Protection Range 2, Upper Bound Register	C014 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR3_L	Data Protection Range 3, Lower Bound Register	C018 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR3_U	Data Protection Range 3, Upper Bound Register	C01C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H

Table 5-21 Memory Protection Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
DPR4_L	Data Protection Range 4, Lower Bound Register	C020 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR4_U	Data Protection Range 4, Upper Bound Register	C024 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR5_L	Data Protection Range 5, Lower Bound Register	C028 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR5_U	Data Protection Range 5, Upper Bound Register	C02C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR6_L	Data Protection Range 6, Lower Bound Register	C030 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR6_U	Data Protection Range 6, Upper Bound Register	C034 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR7_L	Data Protection Range 7, Lower Bound Register	C038 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR7_U	Data Protection Range 7, Upper Bound Register	C03C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR8_L	Data Protection Range 8, Lower Bound Register	C040 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR8_U	Data Protection Range 8, Upper Bound Register	C044 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR9_L	Data Protection Range 9, Lower Bound Register	C048 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR9_U	Data Protection Range 9, Upper Bound Register	C04C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR10_L	Data Protection Range 10, Lower Bound Register	C050 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR10_U	Data Protection Range 10, Upper Bound Register	C054 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR11_L	Data Protection Range 11, Lower Bound Register	C058 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR11_U	Data Protection Range 11, Upper Bound Register	C05C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H

Table 5-21 Memory Protection Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
DPR12_L	Data Protection Range 12, Lower Bound Register	C060 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR12_U	Data Protection Range 12, Upper Bound Register	C064 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR13_L	Data Protection Range 13, Lower Bound Register	C068 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR13_U	Data Protection Range 13, Upper Bound Register	C06C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR14_L	Data Protection Range 14, Lower Bound Register	C070 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR14_U	Data Protection Range 14, Upper Bound Register	C074 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR15_L	Data Protection Range 15, Lower Bound Register	C078 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPR15_U	Data Protection Range 15, Upper Bound Register	C07C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR0_L	Code Protection Range 0, Lower Bound Register	D000 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR0_U	Code Protection Range 0, Upper Bound Register	D004 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR1_L	Code Protection Range 1, Lower Bound Register	D008 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR1_U	Code Protection Range 1, Upper Bound Register	D00C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR2_L	Code Protection Range 2, Lower Bound Register	D010 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR2_U	Code Protection Range 2, Upper Bound Register	D014 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR3_L	Code Protection Range 3, Lower Bound Register	D018 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR3_U	Code Protection Range 3, Upper Bound Register	D01C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H

Table 5-21 Memory Protection Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
CPR4_L	Code Protection Range 4, Lower Bound Register	D020 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR4_U	Code Protection Range 4, Upper Bound Register	D024 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR5_L	Code Protection Range 5, Lower Bound Register	D028 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR5_U	Code Protection Range 5, Upper Bound Register	D02C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR6_L	Code Protection Range 6, Lower Bound Register	D030 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR6_U	Code Protection Range 6, Upper Bound Register	D034 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR7_L	Code Protection Range 7, Lower Bound Register	D038 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPR7_U	Code Protection Range 7, Upper Bound Register	D03C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPXE_0	Code Protection Set-0 Execute Enable	E000 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPXE_1	Code Protection Set-1 Execute Enable	E004 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPXE_2	Code Protection Set-2 Execute Enable	E008 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
CPXE_3	Code Protection Set-3 Execute Enable	E00C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPRE_0	Data Protection Set-0 Read Enable	E010 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPRE_1	Data Protection Set-1 Read Enable	E014 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPRE_2	Data Protection Set-2 Read Enable	E018 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPRE_3	Data Protection Set-3 Read Enable	E01C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H

Table 5-21 Memory Protection Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
DPWE_0	Data Protection Set-0 Write Enable	E020 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPWE_1	Data Protection Set-1 Write Enable	E024 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPWE_2	Data Protection Set-2 Write Enable	E028 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
DPWE_3	Data Protection Set-3 Write Enable	E02C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H

Table 5-22 Temporal Protection System Registers

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
TPS_CON	Control Register	E400 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
TPS_TIMER0	Timer 0 Register	E404 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
TPS_TIMER1	Timer 1 Register	E408 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
TPS_TIMER2	Timer 2 Register	E40C _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H

Table 5-23 Floating Point Special Function Registers

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
FPU_TRAP_CON	Trap Control Register	A000 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
FPU_TRAP_PC	Trapping Instruction Program Counter Register	A004 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
FPU_TRAP_OPC	Trapping Instruction Opcode Register	A008 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
FPU_TRAP_SRC1	Trapping Instruction Operand Register	A010 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H

Table 5-23 Floating Point Special Function Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
FPU_TRAP_SRC2	Trapping Instruction Operand Register	A014 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H
FPU_TRAP_SRC3	Trapping Instruction Operand Register	A018 _H	U, SV, 32	SV, 32 P	Application 0000 0000 _H

Table 5-24 Core Debug and Performance Counter Registers

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
CCTRL	Counter Control Register	FC00 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
CCNT	CPU Clock Count Register	FC04 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
ICNT	Instruction Count Register	FC08 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
M1CNT	Multi-Count Register 1	FC0C _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
M2CNT	Multi-Count Register 2	FC10 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
M3CNT	Multi-Count Register 3	FC14 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
DBGSR	Debug Status Register	FD00 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
EXEVT	External Event Register	FD08 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
CREVT	Core Register Access Event Register	FD0C _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
SWEVT	Software Debug Event Register	FD10 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TRIG_ACC	Debug Trigger Accumulation Register	FD30 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
DMS	Debug Monitor Start Address Register	FD40 _H	U, SV, 32	SV, 32 P	Application CPU Dependent

Table 5-24 Core Debug and Performance Counter Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
DCX	Debug Context Save Area Pointer Register	FD44 _H	U, SV, 32	SV, 32 P	Application CPU Dependent
DBGTCR	Debug Trap Control Register	FD48 _H	U, SV, 32	SV, 32 P	Application, Debug 0000 0001 _H
TR0EVT	Trigger Event 0 Configuration Register	F000 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR0ADR	Trigger Event 0 Address Register	F004 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR1EVT	Trigger Event 1 Configuration Register	F008 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR1ADR	Trigger Event 1 Address Register	F00C _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR2EVT	Trigger Event 2 Configuration Register	F010 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR2ADR	Trigger Event 2 Address Register	F014 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR3EVT	Trigger Event 3 Configuration Register	F018 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR3ADR	Trigger Event 3 Address Register	F01C _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR4EVT	Trigger Event 4 Configuration Register	F020 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR4ADR	Trigger Event 4 Address Register	F024 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR5EVT	Trigger Event 5 Configuration Register	F028 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR5ADR	Trigger Event 5 Address Register	F02C _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR6EVT	Trigger Event 6 Configuration Register	F030 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H
TR6ADR	Trigger Event 6 Address Register	F034 _H	U, SV, 32	SV, 32 P	Debug 0000 0000 _H

Table 5-24 Core Debug and Performance Counter Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
TR7EVT	Trigger Event 7 Configuration Register	F038 _H	U, SV, 32	SV,32 P	Debug 0000 0000 _H
TR7ADR	Trigger Event 7 Address Register	F03C _H	U, SV, 32	SV,32 P	Debug 0000 0000 _H

Table 5-25 DMI Registers

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
DCON0	DMI Control Register 0	9040 _H	U, SV, 32	SV,32 P,CE _x	Application 0000 0002 _H
DCON2	DMI Control Register 2	9000 _H	U, SV, 32	SV,32 P,CE _x	Application CPU Dependent
DSTR	DMI Synchronous Trap Flag Register	9010 _H	U, SV, 32 ¹⁾	SV,32 P	Application 0000 0000 _H
DATR	DMI Asynchronous Trap Flag Register	9018 _H	U, SV, 32 ¹⁾	SV,32 P	Application 0000 0000 _H
DEADD	DMI Data Error Address Register	901C _H	U, SV, 32	SV,32 P	Application 0000 0000 _H
DIEAR	Data Integrity Error Address Register	9020 _H	U, SV, 32	SV,32 P	Application 0000 0000 _H
DIETR	Data Integrity Error Trap Register	9024 _H	U, SV, 32	SV,32 P	Application 0000 0000 _H
SEGEN	SRI Error Generation	1030 _H	U, SV, 32	SV,32 P,CE _x	Application 0000 0000 _H

1) Writing these registers clears the contents independent of the data value.

Table 5-26 PMI Registers

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
PCON0	PMI Control Register 0	920C _H	U, SV, 32	SV, 32, P, CEx	Application 0000 0002 _H
PCON1	PMI Control Register 1	9204 _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
PCON2	PMI Control Register 2	9208 _H	U, SV, 32	SV, 32, P, NC	Application CPU Dependent
PSTR	PMI Synchronous Trap Register	9200 _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
PIEAR	Program Integrity Error Address Register	9210 _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H
PIETR	Program Integrity Error Trap Register	9214 _H	U, SV, 32	SV, 32, P	Application 0000 0000 _H

5.8.9 Summary of SFR Reset Values and Access modes

This section summarizes the reset values and access modes of the CPU SFR registers.

Table 5-27 Safety Protection Registers

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
RGNLA0	Safety Protection Lower Range Register 0	E000 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H
RGNUA0	Safety Protection Upper Range Register 0	E004 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H
RGNACCEN A0	Safety Protection Access Enable Register A Range 0	E008 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H
RGNACCEN B0	Safety Protection Access Enable Register B Range 0	E00C _H	U, SV, 32	U, SV, 32, P, NC	Application 0000 0000 _H

Table 5-27 Safety Protection Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
RGNLA1	Safety Protection Lower Range Register 1	E010 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H
RGNUA1	Safety Protection Upper Range Register 1	E014 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H
RGNACCEN A1	Safety Protection Access Enable Register A Range 1	E018 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H
RGNACCEN B1	Safety Protection Access Enable Register B Range 1	E01C _H	U, SV, 32	U, SV, 32, P, NC	Application 0000 0000 _H
RGNLA02	Safety Protection Lower Range Register 2	E020 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H
RGNUA2	Safety Protection Upper Range Register 2	E024 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H
RGNACCEN A2	Safety Protection Access Enable Register A Range 2	E028 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H
RGNACCEN B2	Safety Protection Access Enable Register B Range 2	E02C _H	U, SV, 32	U, SV, 32, P, NC	Application 0000 0000 _H
RGNLA03	Safety Protection Lower Range Register 3	E030 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H
RGNUA3	Safety Protection Upper Range Register 3	E034 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H
RGNACCEN A3	Safety Protection Access Enable Register A Range 3	E038 _H	U, SV, 32	U, SV, 32, P, SE	Application 0000 0000 _H

Table 5-27 Safety Protection Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
RGNACCEN B3	Safety Protection Access Enable Register B Range 3	E03C _H	U, SV, 32	U,SV, 32,P, NC	Application 0000 0000 _H
RGNLA04	Safety Protection Lower Range Register 4	E040 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H
RGNUA4	Safety Protection Upper Range Register 4	E044 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H
RGNACCEN A4	Safety Protection Access Enable Register A Range 4	E048 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H
RGNACCEN B4	Safety Protection Access Enable Register B Range 4	E04C _H	U, SV, 32	U,SV, 32,NC	Application 0000 0000 _H
RGNLA05	Safety Protection Lower Range Register 5	E050 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H
RGNUA5	Safety Protection Upper Range Register 5	E054 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H
RGNACCEN A5	Safety Protection Access Enable Register A Range 5	E058 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H
RGNACCEN B5	Safety Protection Access Enable Register B Range 5	E05C _H	U, SV, 32	U,SV, 32,NC	Application 0000 0000 _H
RGNLA06	Safety Protection Lower Range Register 6	E060 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H
RGNUA6	Safety Protection Upper Range Register 6	E064 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H
RGNACCEN A6	Safety Protection Access Enable Register A Range 6	E068 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H
RGNACCEN B6	Safety Protection Access Enable Register B Range 6	E06C _H	U, SV, 32	U,SV, 32,NC	Application 0000 0000 _H
RGNLA07	Safety Protection Lower Range Register 7	E070 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H
RGNUA7	Safety Protection Upper Range Register 7	E074 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H
RGNACCEN A7	Safety Protection Access Enable Register A Range 7	E078 _H	U, SV, 32	U,SV, 32,SE	Application 0000 0000 _H

Table 5-27 Safety Protection Registers (cont'd)

Short Name	Description	Offset Address	Access Mode		Reset Type, Value
			Read	Write	
RGNACCEN B7	Safety Protection Access Enable Register B Range 7	E07C _H	U, SV, 32	U, SV, 32, NC	Application 0000 0000 _H
ACCENA	Access Enable Register A	E100 _H	U, SV, 32	U, SV, 32, SE	Application 0000 0000 _H
ACCENB	Access Enable Register B	E104 _H	U, SV, 32	U, SV, 32, SE	Application 0000 0000 _H

Note: A disallowed access to any CPU register (e.g. attempted write to read only register, attempted user mode access to SV, attempted access to E without Endinit, etc.) will NOT result in a Bus Error.

5.9 CPU Instruction Timing

This section gives information on CPU instruction timing by execution unit. The Integer Pipeline and Load/Store Pipeline are always present, and the Floating Point Unit (FPU) is optional. The Load/Store unit implements the optional TLB instructions.

Definition of Terms:

- **Repeat Rate**

Assuming the same instruction is being issued sequentially, repeat is the minimum number of clock cycles between two consecutive issues. There may be additional delays described elsewhere due to internal pipeline effects when issuing a different subsequent instruction.

- **Result Latency**

The number of clock cycles from the cycle when the instruction is issued to the cycle when the result value is available to be used as an operand to a subsequent instruction or written into a GPR. Result latency is not meaningful for instructions that do not write a value into a GPR.

- **Address Latency**

The number of clock cycles from the cycle when the instruction is issued to the cycle when the addressing mode updated value is available as an operand to a subsequent instruction or written into an Address Register.

- **Flow Latency**

The number of clock cycles from the cycle when the instruction is issued to the cycle when the next instruction (located at the target location or the next sequential instruction if the control change is conditional) is issued.

5.9.1 Integer-Pipeline Instructions

These are the Integer-Pipeline instruction timings for each instruction.

5.9.1.1 Simple Arithmetic Instruction Timings

Each instruction is single issued.

Table 5-28 Simple Arithmetic Instruction Timing

Instruction	Result Latency		Repeat Rate		Instruction	Result Latency		Repeat Rate	
	TC16P	TC16E	TC16P	TC16E		TC16P	TC16E	TC16P	TC16E
Integer Pipeline Arithmetic Instructions									
ABS	1	1	1	1	MAX.H	1	1	1	1
ABS.B	1	1	1	1	MAX.HU	1	1	1	1
ABS.H	1	1	1	1	MAX.U	1	1	1	1
ABSDIF	1	1	1	1	MIN	1	1	1	1
ABSDIF.B	1	1	1	1	MIN.B	1	1	1	1
ABSDIF.H	1	1	1	1	MIN.BU	1	1	1	1
ABSDIFS	2	1	1	1	MIN.H	1	1	1	1
ABSDIFS.H	2	1	1	1	MIN.HU	1	1	1	1
ABSS	2	1	1	1	MIN.U	1	1	1	1
ABSS.H	2	1	1	1	RSUB	1	1	1	1
ADD	1	1	1	1	RSUBS	2	1	1	1
ADD.B	1	1	1	1	RSUBS.U	2	1	1	1
ADD.H	1	1	1	1	SAT.B	1	1	1	1
ADDC	1	1	1	1	SAT.BU	1	1	1	1
ADDI	1	1	1	1	SAT.H	1	1	1	1
ADDIH	1	1	1	1	SAT.HU	1	1	1	1
ADDS	2	1	1	1	SEL	1	1	1	1
ADDS.H	2	1	1	1	SELN	1	1	1	1
ADDS.HU	2	1	1	1	SUB	1	1	1	1
ADDS.U	2	1	1	1	SUB.B	1	1	1	1
ADDX	1	1	1	1	SUB.H	1	1	1	1
CADD	1	1	1	1	SUBC	1	1	1	1

Table 5-28 Simple Arithmetic Instruction Timing (cont'd)

Instruction	Result Latency		Repeat Rate		Instruction	Result Latency		Repeat Rate	
	TC16P-TC16E	TC16P-TC16E	TC16P-TC16E	TC16P-TC16E		TC16P-TC16E	TC16P-TC16E	TC16P-TC16E	TC16P-TC16E
CADDN	1	1	1	1	SUBS	2	1	1	1
CSUB	1	1	1	1	SUBS.H	2	1	1	1
CSUBN	1	1	1	1	SUBS.HU	2	1	1	1
MAX	1	1	1	1	SUBS.U	2	1	1	1
MAX.B	1	1	1	1	SUBX	1	1	1	1
MAX.BU	1	1	1	1					
Compare Instructions									
EQ	1	1	1	1	LT.B	1	1	1	1
EQ.B	1	1	1	1	LT.BU	1	1	1	1
EQ.H	1	1	1	1	LT.H	1	1	1	1
EQ.W	1	1	1	1	LT.HU	1	1	1	1
EQANY.B	1	1	1	1	LT.U	1	1	1	1
EQANY.H	1	1	1	1	LT.W	1	1	1	1
GE	1	1	1	1	LT.WU	1	1	1	1
GE.U	1	1	1	1	NE	1	1	1	1
LT	1	1	1	1					
Count Instructions									
CLO	1	1	1	1	CLS.H	1	1	1	1
CLO.H	1	1	1	1	CLZ	1	1	1	1
CLS	1	1	1	1	CLZ.H	1	1	1	1
Extract Instructions									
DEXTR	2	1	1	1	INS.T	1	1	1	1
EXTR	2	1	1	1	INSN.T	1	1	1	1
EXTR.U	2	1	1	1	INSERT	2	1	1	1
IMASK	2	1	1	1					
Logical Instructions									
AND	1	1	1	1	OR.EQ	1	1	1	1
AND.AND.T	1	1	1	1	OR.GE	1	1	1	1
AND.ANDN.T	1	1	1	1	OR.GE.U	1	1	1	1

Table 5-28 Simple Arithmetic Instruction Timing (cont'd)

Instruction	Result Latency		Repeat Rate		Instruction	Result Latency		Repeat Rate	
	TC16P-TC16E	TC16P-TC16E	TC16P-TC16E	TC16P-TC16E		TC16P-TC16E	TC16P-TC16E	TC16P-TC16E	TC16P-TC16E
AND.EQ	1	1	1	1	OR.LT	1	1	1	1
AND.GE	1	1	1	1	OR.LT.U	1	1	1	1
AND.GE.U	1	1	1	1	OR.NE	1	1	1	1
AND.LT	1	1	1	1	OR.NOR.T	1	1	1	1
AND.LT.U	1	1	1	1	OR.OR.T	1	1	1	1
AND.NE	1	1	1	1	OR.T	1	1	1	1
AND.NOR.T	1	1	1	1	ORN	1	1	1	1
AND.OR.T	1	1	1	1	ORN.T	1	1	1	1
AND.T	1	1	1	1	XNOR	1	1	1	1
ANDN	1	1	1	1	XNOR.T	1	1	1	1
ANDN.T	1	1	1	1	XOR	1	1	1	1
NAND	1	1	1	1	XOR.EQ	1	1	1	1
NAND.T	1	1	1	1	XOR.GE	1	1	1	1
NOR	1	1	1	1	XOR.GE.U	1	1	1	1
NOR.T	1	1	1	1	XOR.LT	1	1	1	1
OR	1	1	1	1	XOR.LT.U	1	1	1	1
OR.AND.T	1	1	1	1	XOR.NE	1	1	1	1
OR.ANDN.T	1	1	1	1	XOR.T	1	1	1	1
Move Instructions									
CMOV	1	1	1	1	MOV.U	1	1	1	1
CMOVN	1	1	1	1	MOVH	1	1	1	1
MOV (32 Bit)	1	1	1	1	MOV (64bit)	2	1	1	1
Shift Instructions									
SH	1	1	1	1	SH.NE	1	1	1	1
SH.AND.T	1	1	1	1	SH.NOR.T	1	1	1	1
SH.ANDN.T	1	1	1	1	SH.OR.T	1	1	1	1
SH.EQ	1	1	1	1	SH.ORN.T	1	1	1	1
SH.GE	1	1	1	1	SH.XNOR.T	1	1	1	1
SH.GE.U	1	1	1	1	SH.XOR.T	1	1	1	1

Table 5-28 Simple Arithmetic Instruction Timing (cont'd)

Instruction	Result Latency		Repeat Rate		Instruction	Result Latency		Repeat Rate	
	TC16P-TC16E	TC16P-TC16E	TC16P-TC16E	TC16P-TC16E		TC16P-TC16E	TC16P-TC16E	TC16P-TC16E	TC16P-TC16E
SH.H	1	1	1	1	SHA	1	1	1	1
SH.LT	1	1	1	1	SHA.H	1	1	1	1
SH.LT.U	1	1	1	1	SHAS	2	1	1	1
SH.NAND.T	1	1	1	1					

Coprocessor 0 Instructions

BMERGE	2	2	1	1	IXMIN	2	2	1	1
BSPLIT	2	2	1	1	UNPACK	2	2	1	1
PARITY	2	2	1	1	IXMAX	2	2	1	1
PACK	2	2	1	1	IXMAX.U	2	2	1	1
IXMIN.U	2	2	1	1	CRC32	2	2	1	1

Integer Divide Instructions

DVADJ	2	2	1	1	DVSTEP	6	4	4	3
DVINIT	2	2	1	1	DVSTEP.U	6	4	4	3
DVINIT.U	2	2	1	1	DIV	4-11	3-10	3-9	3-9
DVINIT.B	2	2	1	1	DIV.U	4-11	3-10	3-9	3-9
DVINIT.H	2	2	1	1					
DVINIT.BU	2	2	1	1					
DVINIT.HU	2	2	1	1					

The latency and repeat rate values listed for the DIV and DIV.U instructions are the minimum and maximum values. The algorithm used allows for early termination of the instruction once the full result is available.

5.9.1.2 Multiply Instruction Timings

Each instruction is single issued.

Table 5-29 Multiply Instruction Timing

Instruction	Result Latency TC16P-TC16E		Repeat Rate TC16P-TC16E		Instruction	Result Latency TC16P -TC16E		Repeat Rate TC16P-TC16E	
MUL	2	2	1	1	MUL.Q	3	2	1	1
MUL.U	2	2	1	1	MULM.H	3	2	1	1
MULS	3	2	1	1	MULR.H	3	2	1	1
MULS.U	3	2	1	1	MULR.Q	3	2	1	1
MUL.H	3	2	1	1					

5.9.1.3 Multiply Accumulate (MAC) Instruction Timing

Each instruction is single issued.

Table 5-30 Multiply Accumulate Instruction Timing

Instruction	Result Latency TC16P-TC16E		Repeat Rate TC16P-TC16E		Instruction	Result Latency TC16-TC16E		Repeat Rate TC16P-TC16E	
MADD	3	2	1	1	MSUB	3	2	1	1
MADD.U	3	2	1	1	MSUB.U	3	2	1	1
MADDS	3	2	1	1	MSUBS	3	2	1	1
MADDS.U	3	2	1	1	MSUBS.U	3	2	1	1
MADD.H	3	2	1	1	MSUB.H	3	2	1	1
MADD.Q	3	2	1	1	MSUB.Q	3	2	1	1
MADDM.H	3	2	1	1	MSUBM.H	3	2	1	1
MADDMS.H	3	2	1	1	MSUBMS.H	3	2	1	1
MADDR.H	3	2	1	1	MSUBR.H	3	2	1	1
MADDR.Q	3	2	1	1	MSUBR.Q	3	2	1	1
MADDRS.H	3	2	1	1	MSUBRS.H	3	2	1	1
MADDRS.Q	3	2	1	1	MSUBRS.Q	3	2	1	1
MADDS.H	3	2	1	1	MSUBS.H	3	2	1	1
MADDS.Q	3	2	1	1	MSUBS.Q	3	2	1	1
MADDSU.H	3	2	1	1	MSUBAD.H	3	2	1	1
MADDSUM.H	3	2	1	1	MSUBADM.H	3	2	1	1
MADDSUMS.H	3	2	1	1	MSUBADMS.H	3	2	1	1
MADDSUR.H	3	2	1	1	MSUBADR.H	3	2	1	1
MADDSURS.H	3	2	1	1	MSUBADRS.H	3	2	1	1
MADDSUS.H	3	2	1	1	MSUBADS.H	3	2	1	1

For All MADD, MSUB and MUL type instructions the result latency is reduced to 1 for accumulator forwarding between similar instructions.

For MADD.Q, MADDS.Q, MSUB.Q, MSUBS.Q Instructions:

MADD.Q, MADDS.Q, MSUB.Q, MSUBS.Q	Result Latency TC1.6P	Result Latency TC1.6E	Repeat Rate TC1.6P	Repeat Rate TC1.6E
16 × 16	3	2	1	1
16 × 32	3	2	1	1
32 × 32	3	2	1	1

5.9.1.4 Control Flow Instruction Timing TC1.6P

Control flow instruction timing for TC1.6P is complicated by the use of branch target buffers and fetch FIFOs.

- Incorrectly predicted LS instructions incur a three cycle branch recovery penalty.
- Incorrectly predicted IP instructions incur a four cycle branch recovery penalty.
- Correctly predicted not taken branches incur no penalty
- Correctly predicted taken branch incur a penalty of up to two cycles depending on the state of the fetch FIFOs and the branch target buffer.
- Loop instructions incur the same penalty as an LS conditional jump instruction.

Assumptions

- All target locations yield a full instruction in one access (i.e. not 16-bits of a 32-bit instruction).
- All code fetches take a single cycle.
- Timing is best case; no cache misses for context operations, no pending stores.

Table 5-31 Control flow timing(TC1.6P)

Prediction-Result	Flow Latency (LS, LP)	Repeat rate (LS,LP)	Flow Latency (IP)	Repeat Rate (IP)
Correct Not-Taken	1	1	1	1
Correct Taken	1-2	1-2	1-2	1-2
Incorrect Not-Taken	3	4	3	4
Incorrect Taken	3	4	3	4

5.9.1.5 Control Flow Instruction Timing TC1.6E

The TC1.6E implements a simple static branch prediction scheme. 16 bit instruction format (either direction) and 32 bit format backwards conditional branches are predicted taken. 32 bit format forwards conditional branches are predicted not taken.

- Correctly predicted not taken branches incur no penalty.
- Correctly predicted taken branches incur a single cycle penalty.
- Incorrectly predicted branches incur a two cycle branch recovery penalty.
- Loop instructions incur the same penalty as LS conditional jump instructions.

Assumptions

- All target locations yield a full instruction in one access (i.e. not 16-bits of a 32-bit instruction).
- All code fetches take a single cycle.
- Timing is best case; no cache misses for context operations, no pending stores.

Table 5-32 Control flow timing(TC1.6E)

Prediction-Result	Flow Latency	Repeat rate
Correctly predicted, not taken	1	1
Correctly predicted, taken	2	2
Incorrectly predicted	3	3

5.9.2 Load-Store Pipeline Instructions

This section summarizes the Load-Store Pipeline instructions.

5.9.2.1 Address Arithmetic Timing

Each instruction is single issued.

Table 5-33 Address Arithmetic Instruction Timing

Instruction	Result Latency		Repeat Rate		Instruction	Result Latency		Repeat Rate	
	TC16P-	TC16E	TC16P-	TC16E		TC16P-	TC16E	TC16P-	TC16E

Load Store Arithmetic Instructions

ADD.A	1	1	1	1	GE.A	1	1	1	1
ADDIH.A	1	1	1	1	LT.A	1	1	1	1
ADDSC.A	2	1	1	1	NE.A	1	1	1	1
ADDSC.AT	2	1	1	1	NEZ.A	1	1	1	1
EQ.A	1	1	1	1	SUB.A	1	1	1	1
EQZ.A	1	1	1	1	NOP	1	1	1	1

Trap and Interrupt Instructions

DEBUG	–	–	1	1	TRAPSV¹⁾	–	–	1	1
DISABLE	–	–	1	1	TRAPV¹⁾	–	–	1	1
ENABLE	–	–	1	1	RSTV	–	–	1	1
RESTORE	–	–	1	1	WAIT²⁾	–	–	1	1

Move Instructions

MFCR	2	2	1	1	MOV.A	1	1	1	1
MTCR	1	1	1	1	MOV.AA	1	1	1	1
MOVH.A	1	1	1	1	MOV.D	1	1	1	1

Sync Instructions

DSYNC	–	–	1	1	ISYNC³⁾	–	–	1	1
--------------	---	---	---	---	---------------------------	---	---	---	---

1) Execution cycles when no TRAP is taken. The execution timing in the case of raising these TRAPs is the same as other TRAPs such as SYSCALL.

2) The latency of the WAIT instruction is hidden by the context save of the interrupt. Effective latency is zero.

3) Repeat rate assumes that code refetch takes a single cycle.

5.9.2.2 CSA Control Flow Instruction Timing

This section summarizes the timing of CSA Control Flow instructions.

- All targets yield a full instruction in one access (not 16-bits of a 32-bit instruction).
- All code fetches take a single cycle. Timing is best case; no cache misses for context operations, no pending stores.

Table 5-34 CSA Control Flow Instruction Timing

Instruction	Result Latency TC16P-TC16E		Repeat Rate TC16P-TC16E		Instruction	Result Latency TC16P-TC16E		Repeat Rate TC16P-TC16E	
CALL	4-8	4	4-8	4	SYSCALL	4-8	4	4-8	4
CALLA	4-8	4	4-8	4	SVLCX	4-8	4	4-8	4
CALLI	4-8	4	4-8	4	RSLCX	4-8	4	4-8	4
RET	4-8	4	4-8	4	RFE	4-8	4	4-8	4
BISR	4-8	4	4-8	4	RFM	4-8	4	4-8	4
FCALL	1	1	1	1	FCALLA	1	1	1	1
FCALLI	1	1	1	1	FRET	1	1	1	1

Access to DSPR require 4 cycles, accesses to cached external memory require 8 cycles (TC1.6P only).

5.9.2.3 Load Instruction Timing

Load instructions can produce two results if they use the pre-increment, post-increment, circular or bit-reverse addressing modes. Hence, in those cases there are two latencies that must be specified, the result latency for the value loaded from memory and the address latency for using the updated address register result.

- Each instruction is single issued.
- The memory references is naturally aligned.
- The memory accessed takes a single cycle to return a data item.
- Timing is best case; no cache misses, no pending stores.

Table 5-35 Load Instruction Timing TC1.6P

Instruction	Address Latency	Result Latency	Repeat Rate	Instruction	Address Latency	Result Latency	Repeat Rate
Load Instructions							
LD.A	1	3	1	LD.Q	1	2	1
LD.B	1	2	1	LD.W	1	2	1

Table 5-35 Load Instruction Timing TC1.6P (cont'd)

Instruction	Address Latency	Result Latency	Repeat Rate	Instruction	Address Latency	Result Latency	Repeat Rate
LD.BU	1	2	1	LDLCX	5-9	5-9	5
LD.D	1	2	1	LDUCX	5-9	5-9	5
LD.DA	1	3	1	SWAP.W	2	3	2
LD.H	1	2	1	LEA ¹⁾	–	1	1
LD.HU	1	2	1	CMPSWAP	2	3	1
SWAPMSK	2	3	1				

1) The addressing mode returning an updated address is not relevant for this instruction.

Table 5-36 Load Instruction Timing TC1.6E

Instruction	Address Latency	Result Latency	Repeat Rate	Instruction	Address Latency	Result Latency	Repeat Rate
Load Instructions							
LD.A	1	3	1	LD.Q	1	2	1
LD.B	1	2	1	LD.W	1	2	1
LD.BU	1	2	1	LDLCX	5	5	5
LD.D	1	2	1	LDUCX	5	5	5
LD.DA	1	3	1	SWAP.W	2	3	2
LD.H	1	2	1	LEA ¹⁾	–	1	1
LD.HU	1	2	1	CMPSWAP	2	3	1
SWAPMSK	2	3	1				

1) The addressing mode returning an updated address is not relevant for this instruction.

5.9.2.4 Store Instruction Timing

Cache and Store instructions similar to Load instructions will have a result for the pre-increment, post-increment, circular or bit-reverse addressing modes, but do not produce a 'memory' result.

- Each instruction is single issued.
- The memory references is naturally aligned.
- The memory accessed takes a single cycle to accept a data item.
- Timing is best case; no cache misses, no pending stores.

Table 5-37 Cache and Store Instruction Timing

Instruction	Result Latency				Repeat Rate				
	TC16P		-TC16E		TC16P		-TC16E		
Cache Instructions									
CACHEA.I	1	1	1	1	CACHEA.WI¹⁾	1	1	1	1
CACHEA.W¹⁾	1	1	1	1	CACHEI.W	1	1	1	1
CACHEI.WI	1	1	1	1	CACHEI.I	1	1	1	1
Store Instructions									
ST.A	1	1	1	1	ST.T	1	1	2	2
ST.B	1	1	1	1	ST.W	1	1	1	1
ST.D	1	1	1	1	STLCX	1	1	9	5
ST.DA	1	1	1	1	STUCX	1	1	9	5
ST.H	1	1	1	1	LDMST	1	1	2	2
ST.Q	1	1	1	1					

1) Repeat rate assumes that no memory writeback operation occurs. Otherwise the repeat rate will depend upon the time for the castout buffers to clear.

5.9.3 Floating Point Pipeline Timing

These instructions are only valid if the optional Floating Point Unit is implemented. Each instruction is single issued.

Table 5-38 Floating Point Instruction Timing

Instruction	Result Latency				Repeat Rate				
	TC16P		TC16E		TC16P		TC16E		
ADDF	2	2	1	1	ITOF	2	1	1	1
CMP.F	1	1	1	1	MADD.F	3	2	1	1
DIV.F	8	7	6	6	MSUB.F	3	2	1	1
FTOI	2	1	1	1	MUL.F	2	2	1	1
FTOIZ	2	1	1	1	Q31TOF	2	1	1	1
FTOQ31	2	1	1	1	QSEED.F	1	1	1	1
FTOQ31Z	2	1	1	1	SUB.F	2	2	1	1
FTOU	2	1	1	1	UPDFL	–	–	1	1
FTOUZ	2	1	1	1	UTOF	2	1	1	1

5.10 Program Memory Interface (PMI)

The program Memory Interface (PMI) provides the instruction stream to the CPU.

5.10.1 TC1.6P PMI Description

Figure 5-9 shows the block diagram of the Program Memory Interface (PMI) of the TC1.6P.

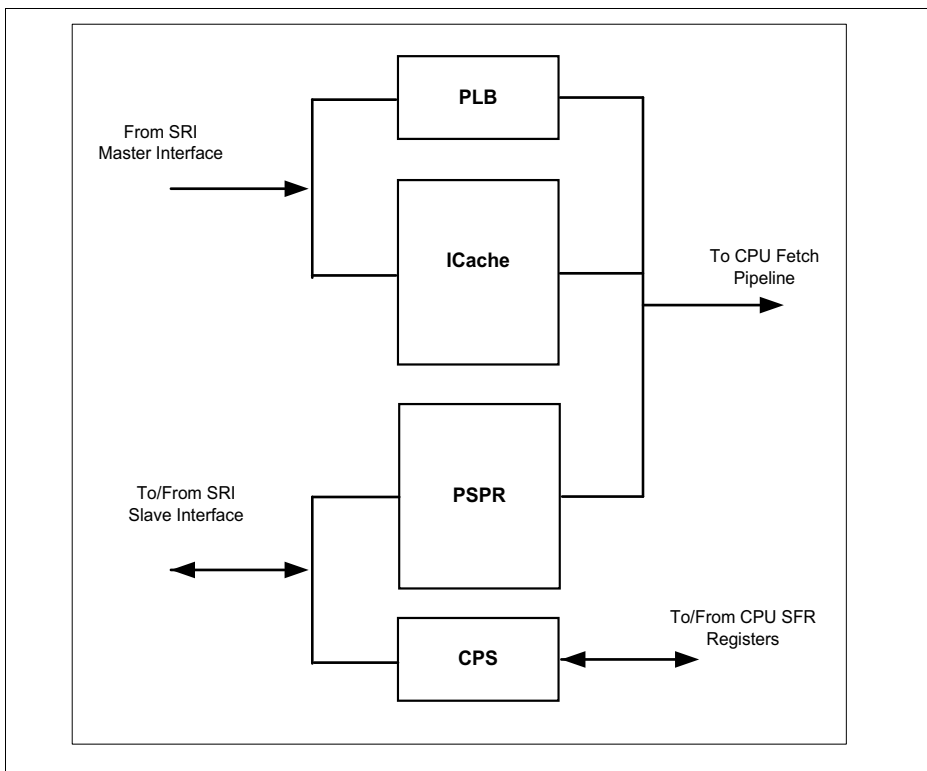


Figure 5-9 PMI Block Diagram

The Program Memory Interface (PMI) has the following features:

- Program Cache (PCACHE)
 - Two-way set associative cache
 - LRU (Least-Recently Used) replacement algorithm
 - Cache line size: 256 bits (4 double-words)
 - Validity granularity: One valid bit per cache line
 - Single cycle access for hit.

CPU Subsystem

- PCACHE can be globally invalidated to provide support for software cache coherency (to be handled by the programmer)
- PCACHE can be bypassed to provide a direct fetch from the CPU to on-chip and off-chip resources
- PCACHE refill mechanism:
 - Critical word first, line wrap around, streaming to CPU
- Program Scratchpad memory (PSPR)
- CPU interface
 - Supporting 64bit aligned fetches. (TC1.6P)
- CPU Slave interface (CPS)
- Shared Resource Interconnect Bus (SRI) Master Interface
- Shared Resource Interconnect Bus (SRI) Slave Interface to scratchpad RAM and CPS.
- All PMI SRAMs (PSPR, PCACHE, and cache tag SRAM) are ECC protected
 - ECC is calculated on 64bit data for the PSPR and PCACHE

5.10.1.1 TC1.6P Scratchpad RAM

The TC1.6P of Program scratchpad RAM provides a fast, deterministic program fetch access from the CPU for use by performance critical code sequences.

- CPU program fetch accesses to scratchpad RAM are never cached in the program cache and are always directly targeted to the scratchpad RAM.

The TC1.6P CPU fetch interface will generate aligned accesses (64-bit), which will result in 64-bits of instruction being returned to the CPU.

Note that the CPU Fetch Unit can only read from the scratchpad RAM and can never write to it.

The scratchpad RAM may also be accessed from the SRI Slave interface by another bus master, such as the Data Memory Interface (DMI). The scratchpad RAM may be both read and written from the SRI. The SRI Slave interface supports all SRI transaction types.

The scratchpad RAM is ECC protected across 64bit data.

5.10.1.2 TC1.6P Program Cache

The program Cache is a two-way set-associative cache with a Least-Recently-Used (LRU) replacement algorithm. Each PCACHE line contains 256 bits of instruction and associated ECC bits.

CPU program fetch accesses which target a cacheable memory segment (and where the PCACHE is not bypassed) target the PCACHE. If the requested address and its associated instruction are found in the cache (Cache Hit), the instruction is passed to the CPU Fetch Unit without incurring any wait states. If the address is not found in the cache (Cache Miss), the PMI cache controller issues a cache refill sequence and wait states

CPU Subsystem

are incurred whilst the cache line is refilled. The fetch request always returns an aligned packet to the CPU.

The program cache is ECC protected on 64bit data.

The program TAG Ram is ECC protected on 22bit data.

Errors detected at the ECC decoders are signaled to the EMM via the MBIST logic of the associated SRAM

Program Cache Refill Sequence

Program Cache refills are performed using a critical double-word first strategy with cache line wrapping such that the refill size is always 4 double-words. PCACHE refills are always performed in 64-bit quantities. A refill sequence will always affect only one cache line. There is no prefetching of the next cache line.

PCACHE refills are therefore implemented using an SRI Burst Transfer 4 (BTR4) transfers. The program cache supports instruction streaming, meaning that it can deliver available instruction half-words to the CPU Fetch Unit whilst the refill operation is ongoing.

Program Cache Bypass

The Program Cache may be bypassed, under control of PCON0.PCBYP, to provide a direct instruction fetch path for the CPU Fetch Unit. The default value of PCON0.PCBYP is such that the PCACHE is bypassed after reset.

Whilst PCACHE bypass is enabled, a fetch request by the CPU to a cacheable address will result in a forced cache miss, such that the cache controller issues a standard refill sequence and supplies instruction half-words to the CPU using instruction streaming, without updating the cache contents. Any valid cache lines within the PCACHE will remain valid and unchanged whilst the PCACHE is bypassed. As such, instruction fetch requests to cacheable addresses with PCACHE bypass enabled behave identically to instruction fetch requests to non-cacheable addresses.

The PCON0 register is endinit protected.

Program Cache Invalidation

The PMI does not have automatic cache coherency support. Changes to the contents of memory areas external to the PMI that may have already been cached in the PCACHE are not detected. Software must provide the cache coherency in such a case. The PMI supports this via the cache invalidation function. The PCACHE contents may be globally invalidated by writing a '1' to PCON1.PCINV. The PCACHE invalidation is performed over 64 cycles by a hardware state machine which cycles through the PCACHE entries marking each as invalid. During an invalidate sequence the CPU may continue to fetch instructions from non-cacheable memory. Any attempt to fetch instructions from a cacheable memory location during an invalidation sequence will result in the CPU

stalling until the sequence completes. The status of the PCACHE invalidation sequence may be determined by reading the PCON1.PCINV bit.

5.10.1.3 TC1.6P Program Line Buffer

The PMI module contains a 256-bit Program Line Buffer (PLB). Program fetch requests to non-cacheable addresses (or to cacheable addresses with the cache in bypass) utilize the PLB as a single line cache. A single valid bit is associated with the PLB, denoting that the PLB contents are valid. As such all fetch requests resulting in an update of the PLB, whether to a cacheable address or not, are implemented as SRI Burst Transfer 4 (BTR4) transactions, with the critical double-word of the PLB line being fetched first size. The PLB may be invalidated by writing PCON1.PBINV.

5.10.1.4 TC1.6P CPU Slave Interface (CPS)

The CPU Slave Interface provides access from the SRI bus to the CPU CSFR and SFR registers.

5.10.2 TC1.6E PMI Description

Figure 5-10 shows the block diagram of the Program Memory Interface (PMI) of the TC1.6E.

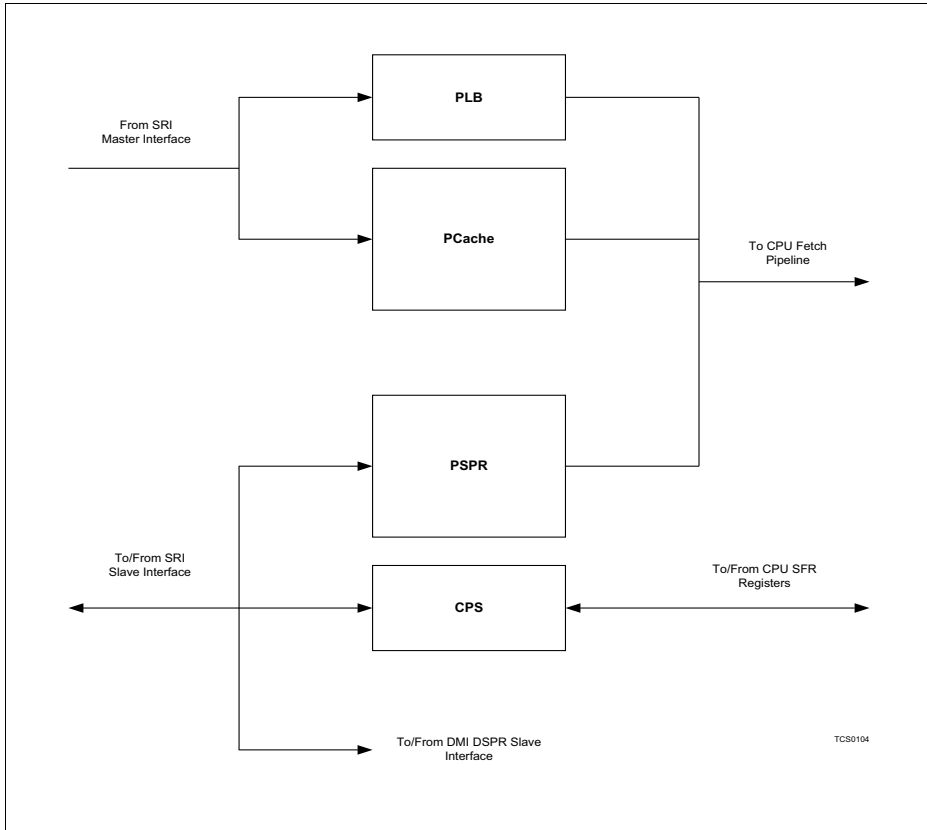


Figure 5-10 PMI Block Diagram

The Program Memory Interface (PMI) has the following features:

- Program Cache (PCACHE)
 - Two-way set associative cache
 - LRU (Least-Recently Used) replacement algorithm
 - Cache line size: 256 bits (4 double-words)
 - Validity granularity: One valid bit per cache line

CPU Subsystem

- PCACHE can be globally invalidated to provide support for software cache coherency (to be handled by the programmer)
- PCACHE can be bypassed to provide a direct fetch from the CPU to on-chip and off-chip resources
- PCACHE refill mechanism:
 - Critical word first, line wrap around, streaming to CPU
- Program Scratchpad memory (PSPR)
- CPU interface
 - Supporting 32-bit aligned fetches. (TC1.6E)
- CPU Slave interface (CPS)
- Shared Resource Interconnect Bus (SRI) Master Interface
- Shared Resource Interconnect Bus (SRI) Slave Interface to Program Scratchpad RAM and CPU Slave interface.
 - SRI Slave Interface is shared with DMI for Data Scratchpad RAM access.
- All PMI SRAMs (PSPR, PCACHE, and cache tag SRAM) are ECC protected.
 - ECC is calculated on 32bit data for the PSPR and PCACHE

5.10.2.1 TC1.6E Program Scratchpad RAM

The Program Scratchpad RAM provides a fast, deterministic program fetch access from the CPU for use by performance critical code sequences.

- CPU program fetch accesses to scratchpad RAM are never cached in the program cache and are always directly targeted to the scratchpad RAM.

The TC1.6E CPU fetch interface will generate aligned accesses (32-bit), which will result in 32-bits of instruction being returned to the CPU.

Note that the CPU Fetch Unit can only read from the scratchpad RAM and can never write to it.

The scratchpad RAM may also be accessed from the SRI Slave interface by another bus master, such as the Data Memory Interface (DMI). The scratchpad RAM may be both read and written from the SRI. The SRI Slave interface supports all SRI transaction types.

The scratchpad RAM is ECC protected across 32bit data.

Errors detected at the ECC decoders are signaled to the EMM via the MBIST logic of the associated SRAM

5.10.2.2 TC1.6E Program Cache

The Program Cache is a two-way set-associative cache with a Least-Recently-Used (LRU) replacement algorithm. Each PCACHE line contains 256 bits of instruction along with associated ECC bits.

CPU program fetch accesses which target a cacheable memory segment (and where the PCACHE is not bypassed) target the PCACHE. If the requested address and its

CPU Subsystem

associated instruction are found in the cache (Cache Hit), the instruction is passed to the CPU Fetch Unit without incurring any wait states. If the address is not found in the cache (Cache Miss), the PMI cache controller issues a cache refill sequence and wait states are incurred whilst the cache line is refilled. The fetch request always returns an aligned packet to the CPU.

The Program cache is ECC protected on 32bit data.

The Program TAG Ram is ECC protected on 22bit data.

Errors detected at the ECC decoders are signaled to the EMM via the MBIST logic of the associated SRAM

Program Cache Refill Sequence

Program Cache refills are performed using a critical double-word first strategy with cache line wrapping such that the refill size is always 4 double-words. PCACHE refills are always performed in 64-bit quantities. A refill sequence will always affect only one cache line. There is no prefetching of the next cache line.

PCACHE refills are therefore implemented using an SRI Burst Transfer 4 (BTR4) transfers. The program cache supports instruction streaming, meaning that it can deliver available instruction half-words to the CPU Fetch Unit whilst the refill operation is ongoing.

Program Cache Bypass

The Program Cache may be bypassed, under control of PCON0.PCBYP, to provide a direct instruction fetch path for the CPU Fetch Unit. The default value of PCON0.PCBYP is such that the PCACHE is bypassed after reset.

Whilst PCACHE bypass is enabled, a fetch request by the CPU to a cacheable address will result in a forced cache miss, such that the cache controller issues a standard refill sequence and supplies instruction half-words to the CPU using instruction streaming, without updating the cache contents. Any valid cache lines within the PCACHE will remain valid and unchanged whilst the PCACHE is bypassed. As such, instruction fetch requests to cacheable addresses with PCACHE bypass enabled behave identically to instruction fetch requests to non-cacheable addresses.

The PCON0 register is endinit protected.

Program Cache Invalidation

The PMI does not have automatic cache coherency support. Changes to the contents of memory areas external to the PMI that may have already been cached in the PCACHE are not detected. Software must provide the cache coherency in such a case. The PMI supports this via the cache invalidation function. The PCACHE contents may be globally invalidated by writing a '1' to PCON1.PCINV. The PCACHE invalidation is performed over 64 cycles by a hardware state machine which cycles through the PCACHE entries

marking each as invalid. During an invalidate sequence the CPU may continue to fetch instructions from non-cacheable memory. Any attempt to fetch instructions from a cacheable memory location during an invalidation sequence will result in the CPU stalling until the sequence completes. The status of the PCACHE invalidation sequence may be determined by reading the PCON1.PCINV bit.

5.10.2.3 TC1.6E Program Line Buffer

The PMI module contains a 256-bit Program Line Buffer (PLB). Program fetch requests to non-cacheable addresses (or to cacheable addresses with the cache in bypass) utilize the PLB as a single line cache. A single valid bit is associated with the PLB, denoting that the PLB contents are valid. As such all fetch requests resulting in an update of the PLB, whether to a cacheable address or not, are implemented as SRI Burst Transfer 4 (BTR4) transactions, with the critical double-word of the PLB line being fetched first size. The PLB may be invalidated by writing PCON1.PBINV.

5.10.2.4 TC1.6E CPU Slave Interface (CPS)

The CPU Slave Interface provides access from the SRI bus to the CPU CSFR and SFR registers.

5.10.3 PMI Registers

Three control registers are control the operation of the Program Memory Interface. These registers and their bits are described in this section.

5.10.3.1 PMI Register Descriptions

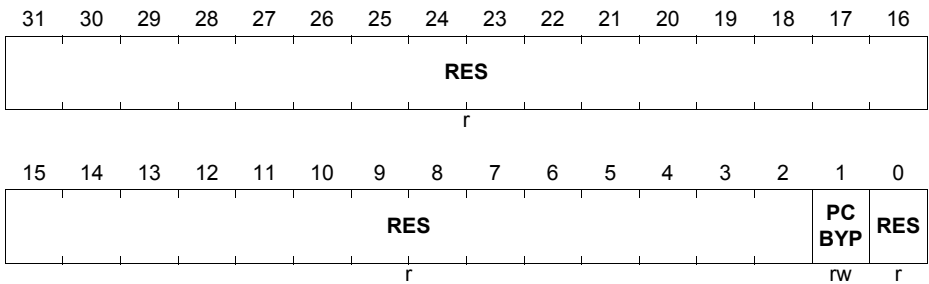
PMI Control Register 0

PCON0

Program Memory Control Register 0

(CSFR_Base + 920C_H)

Reset Value: 0000 0002_H



Field	Bits	Type	Description
RES	0	r	Reserved Read as 0; should be written with 0.
PCBYP	1	rw	Program Cache Bypass 0 _B Cache enabled 1 _B Cache bypassed (disabled)
RES	[31:2]	r	Reserved Read as 0; should be written with 0.

Note: This register is endinit protected.

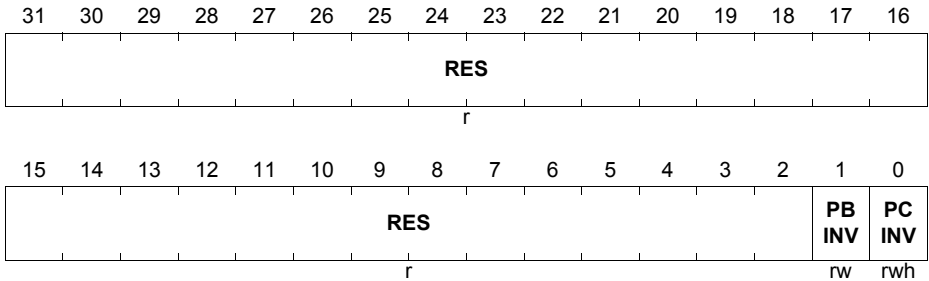
PMI Control Register 1

PCON1

Program Memory Control Register 1

(CSFR_Base + 9204_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
PCINV	0	rwh	<p>Program Cache Invalidate</p> <p>Write Operation:</p> <ul style="list-style-type: none"> 0_B No effect. Normal program cache operation. 1_B Initiate invalidation of entire program cache. <p>Read Operation:</p> <ul style="list-style-type: none"> 0_B Normal operation. Program cache available. 1_B Program cache invalidation in progress. Program cache unavailable.
PBINV	1	rw	<p>Program Buffer Invalidate</p> <p>Write Operation:</p> <ul style="list-style-type: none"> 0_B No effect. Normal program line buffer operation. 1_B Invalidate the program line buffer. <p>This field returns 0 when read.</p>
RES	[31:2]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

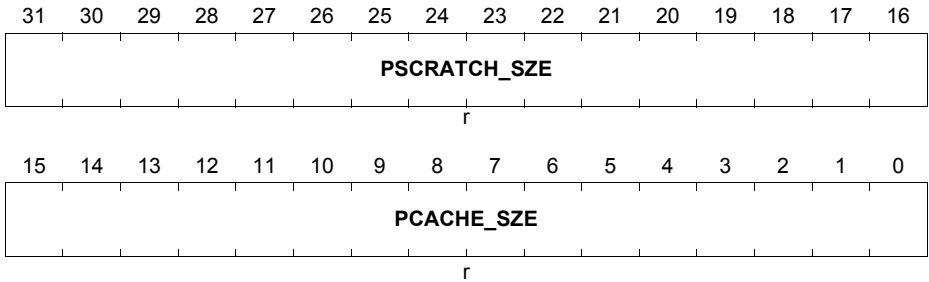
PMI Control Register 2

PCON2 contains size information for the Program memory system.

PCON2

Program Memory Control Register 2

(CSFR_Base + 9208_H) Reset Value: CPU Dependent

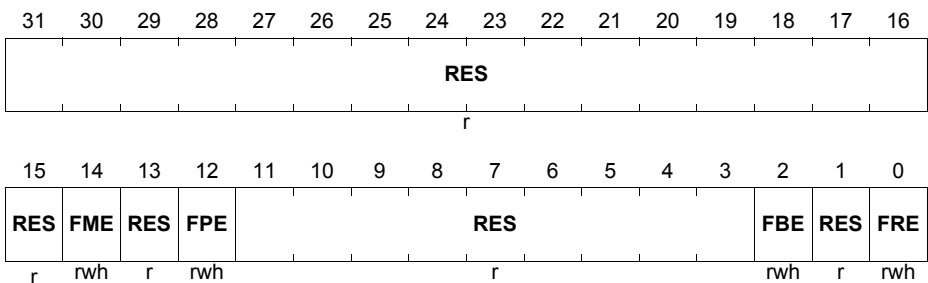


Field	Bits	Type	Description
PCACHE_SIZE	[15:0]	r	Program Cache (PCACHE) Size In KBytes
PSCRATCH_SIZE	[31:16]	r	Program Scratch Size In KBytes

Program Memory Interface Synchronous Trap Register (PSTR)

PSTR contains synchronous trap information for the program memory system. The register is updated with trap information for PSE traps to aid the localisation of faults.

The register is only set whenever a trap is detected and the register has no bits already set. It is cleared by a CSFR write (independent of data value).

PSTR
Program Synchronous Trap Register
(CSFR_Base + 9200_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
FRE	0	rwh	Fetch Range Error Code fetch from local PSPR area outside of physically implemented memory
RES	1	r	Reserved
FBE	2	rwh	Fetch Bus Error Code fetch from bus address giving an error
RES	[11:3]	r	Reserved
FPE	12	rwh	Fetch Peripheral Error Code fetch from peripheral space
RES	13	r	Reserved
FME	14	rwh	Fetch MSIST Error Code fetch from memory mapped PTAG.
RES	[31:15]	r	Reserved

Fetch Range Error

A Fetch Range Error occurs whenever an access to the Program Scratch is outside the range of the SRAM.

Fetch Global Address Error

A Fetch Global Address Error occurs whenever there is access to an address that cannot be translated to a valid global address. This will occur for accesses in the range D1000000 to D7FFFFFFF.

Fetch Bus Error

A Fetch bus error will be set whenever the SRI flags an error due to a fetch from external memory. This will be set for both direct fetches from the bus and for cache refills.

Fetch Peripheral Error

A Fetch peripheral error will be flagged whenever a fetch is attempted to peripheral space.

Fetch MSIST Error

During SISTR mode, a fetch from the PTAG will cause a PSE trap to occur.

5.11 Data Memory Interface (DMI)

The Data Memory Interface (DMI) provides data values to the CPU and stores data values supplied by the CPU.

5.11.1 TC1.6P DMI Description

This figure shows the block diagram of the Data Memory Interface (DMI) of the TC1.6P.

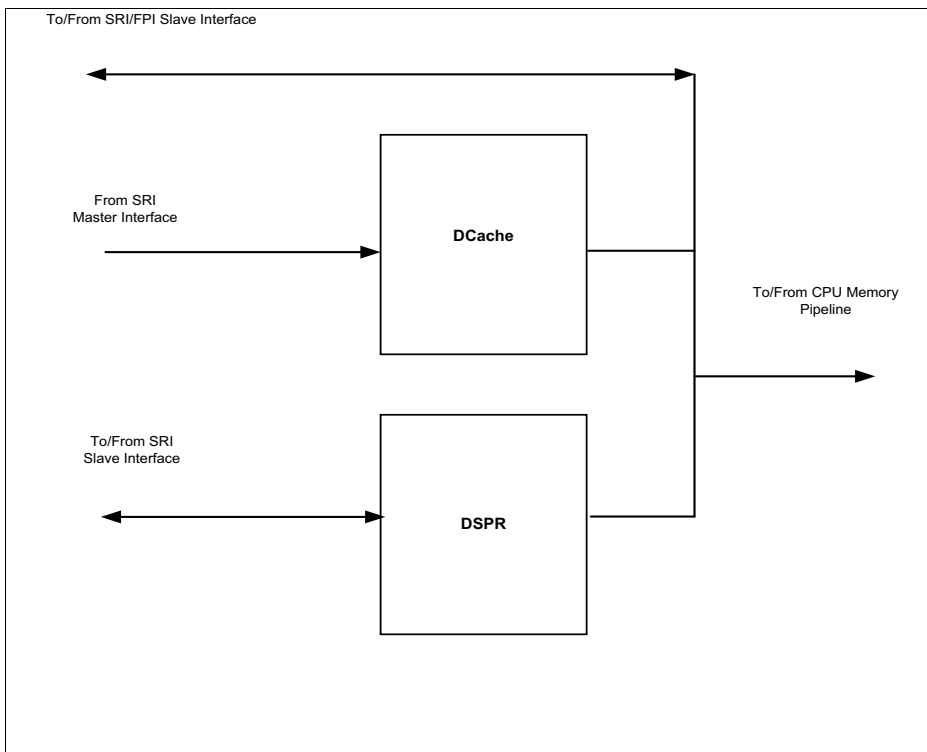


Figure 5-11 DMI Block Diagram

The Data Memory Interface (DMI) has the following features:

- Data Scratchpad Ram (DSPR)
 - Supporting unaligned access (16-bit aligned) with no penalty.
- Data Memory (DCACHE):
 - Two-way set associative cache, Least recently used (LRU) replacement algorithm
 - Cache line size: 256 bits
 - Validity granularity: One valid bit per cache line

- Write-back Cache: Writeback granularity: 256 bits
- Refill mechanism: full cache line refill
- Single cycle access for hit.
- CPU interface
- Shared Resource Interconnect Bus (SRI) Master interface
- Shared Resource Interconnect (SRI) Slave interface to DSPR
- Flexible Peripheral Interconnect Bus (FPI) Master interface for fast access to peripherals
- All DMI SRAMs (DSPR, DCACHE, and cache tag SRAM) are ECC protected

5.11.1.1 TC1.6P Data Scratchpad RAM (DSPR)

The DSPR provides fast, deterministic data access to the CPU for use by performance critical code sequences.

The DSPR is organised as multiple memory “towers”. This organisation allow the CPU to access 64bits of data from any 16bit aligned address

The DSPR may also be accessed from the SRI Slave interface by another bus master, with both read and write transactions supported. The DSPR may be accessed by the SRI Slave interface using any SRI transaction type, including burst transfers. In accordance with the SRI protocol, accesses to the SRI Slave interface must be naturally aligned.

All TC1.6P Data scratchpad RAMs are ECC protected.

Errors detected at the ECC decoders are signaled to the EMM via the MBIST logic of the associated SRAM

5.11.1.2 TC1.6P Data Cache (DCACHE)

The DCache is a Two-way set-associative cache with a Least-Recently-Used (LRU) replacement algorithm. Each line contains 256 bits of data along with ECC bits. A single valid bit and a single dirty bit are associated with each line.

CPU data accesses to a cacheable memory segment target the DCache. If the requested address and its associated data are found in the cache (Cache Hit), the data is passed to/from the CPU Load-Store Unit without incurring any wait states. If the address is not found in the cache (Cache Miss), the DMI cache controller issues a cache refill sequence and wait states are incurred whilst the cache line is refilled. The CPU load-store interface will generate unaligned accesses (16-bit aligned), which will result in up to 64-bits of data being transferred to or from the CPU (for non-context operations). If the data access is made within a DCache line, no matter the alignment, and a cache hit is detected then the requested data is returned to the CPU in a single cycle. If the data access is made to the end of a DCache line, such that the requested data would span two DCache lines, a single wait cycle is incurred (if both cache lines are present in the cache, otherwise a refill sequence is required for the missing cache line(s)).

CPU Subsystem

The TC1.6P data cache is of the writeback type. When the CPU writes to a cacheable location the data is merged with the corresponding cache line and not written to main memory immediately. Associated with each cache line is a single 'dirty' bit, to denote that the data in the cache line has been modified. Whenever a CPU load-store access results in a cache miss, and each of the potential cache ways that could hold the requested cache line are valid, one of the cache lines is chosen for eviction based upon the LRU replacement algorithm. The line selected for eviction is then checked to determine if it has been modified using its dirty bit. If the line has not been modified the line is discarded and the refill sequence started immediately. If the line has been modified then the dirty data is first written back to main memory before the refill is initiated. Due to the single dirty bit per cache line, 256 bits of data will always be written back, resulting in a SRI Burst-4 Transfer (BTR4) transactions.

Data Cache refills always result in the full cache line being refilled, with the critical double-word of the DCache line being fetched first. A refill sequence will always affect only one cache line. There is no prefetching of the next cache line. Due to the uniform size of DCache refill sequences, such refills are always implemented using SRI Burst Transfer 4 (BTR4) transactions.

All TC1.6P Cache SRAMs are ECC protected.

All TC1.6P TAG SRAMs are ECC protected

Errors detected at the ECC decoders are signaled to the EMM via the MBIST logic of the associated SRAM

Data Cache Bypass

The Data Cache may be bypassed, under control of DCON0.DCBYP, to provide a direct data access path to memory. The default value of DCON0.DCBYP is such that the data cache is bypassed after reset.

Whilst the data cache bypass is enabled, a data access request by the CPU to a cacheable address will result in a forced cache miss. Any valid cache lines within the data cache will remain valid and unchanged whilst the data cache is bypassed. As such data accesses to cacheable addresses with the data cache bypass enabled behave identically to data accesses to non-cacheable addresses.

The DCON0 register is endinit protected.

5.11.2 TC1.6E DMI Description

This figure shows the block diagram of the Data Memory Interface (DMI) of the TC1.6E.

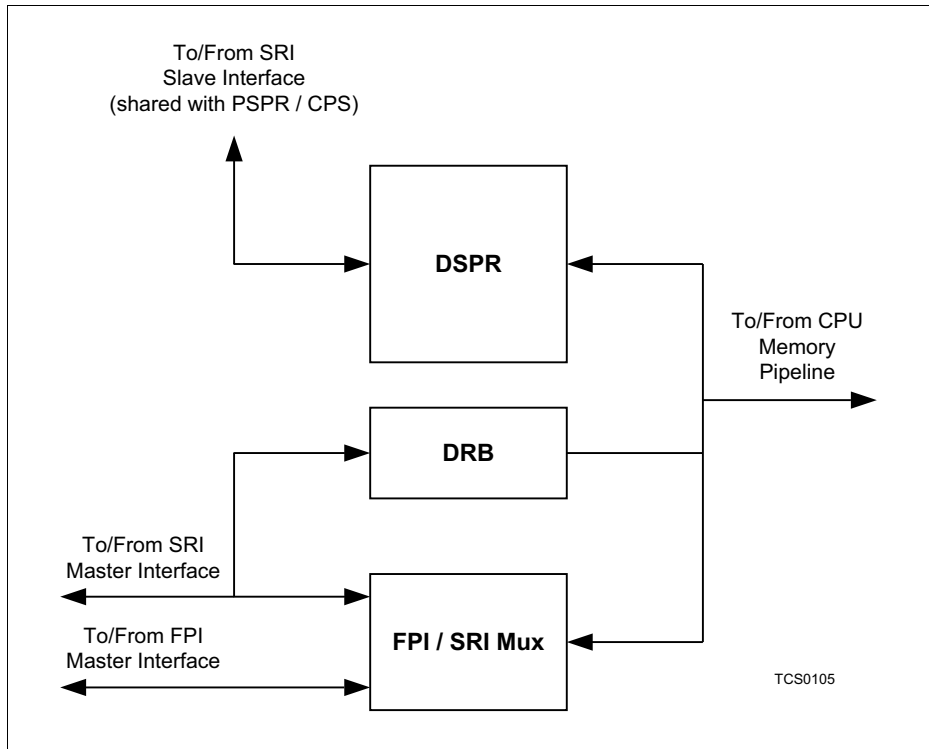


Figure 5-12 DMI Block Diagram

The Data Memory Interface (DMI) has the following features:

- Data Scratchpad Ram (DSPR)
 - Supporting unaligned access (16-bit aligned) with no penalty.
- Data Read Buffer (DRB)
 - Four-entry fully associative Data Read Buffer
 - Least recently used (LRU) replacement algorithm
 - Buffer line size: 256 bits
 - Validity granularity: One valid bit per buffer line
 - Refill mechanism: full buffer line refill, critical double-word first
- CPU interface
- Shared Resource Interconnect Bus (SRI) Master interface

CPU Subsystem

- Flexible Peripheral Interconnect Bus (FPI) Master interface for fast access to peripherals
 - CPU accesses to lower half of segment F_H are directed to FPI master interface, other accesses directed to SRI master interface
- Shared Resource Interconnect (SRI) Slave interface to DSPR
 - SRI slave interface shared with PMI for access to PSPR / CPS
- All TC1.6E DMI SRAM (DSPR) ECC protected

5.11.2.1 TC1.6E Data Scratchpad RAM (DSPR)

The DSPR provides fast, deterministic data access to the CPU for use by performance critical code sequences.

The DSPR is organised as multiple memory “towers”. This organisation allow the CPU to access 64bits of data from any 16bit aligned address

The DSPR may also be accessed from the SRI Slave interface by another bus master, with both read and write transactions supported. The DSPR may be accessed by the SRI Slave interface using any SRI transaction type, including burst transfers. In accordance with the SRI protocol, accesses to the SRI Slave interface must be naturally aligned.

Errors detected at the ECC decoders are signaled to the EMM via the MBIST logic of the associated SRAM

5.11.2.2 TC1.6E Data Read Buffer (DRB)

The TC1.6E implements a 128-byte Data Read Buffer (DRB). The DRB contains four fully associative buffer lines, with a Least-Recently-Used (LRU) replacement algorithm. Each buffer line contains 256-bits of data. A single valid bit is associated with each buffer line. The DRB will buffer CPU read data from cacheable memory segments.

CPU load data accesses, where the addressed memory segment is defined as cacheable, target the DRB. If the requested address and its associated data are found in the DRB (Buffer Hit), the data is passed to/from the CPU Load-Store Unit without incurring any wait states. If the address is not found in the DRB (Buffer Miss), the DMI controller issues a buffer refill sequence and wait states are incurred whilst the buffer line is refilled. The CPU load-store interface will generate unaligned accesses (16-bit aligned), which will result in up to 64-bits of data being transferred to or from the CPU (for non-context operations). If the data access is made within a DRB buffer line, no matter the alignment, and a buffer hit is detected then the requested data is returned to the CPU in a single cycle. If the data access is made to the end of a DRB line, such that the requested data would span two DRB lines, a single wait cycle is incurred (if both buffer lines are present in the DRB, otherwise a refill sequence is required for the missing buffer line(s)).

DRB refills always result in the full buffer line being refilled, with the critical double-word of the DRB line being fetched first. A refill sequence will always affect only one buffer

line. There is no prefetching of the next buffer line. Due to the uniform size of DRB refill sequences, such refills are always implemented using SRI Burst Transfer 4 (BTR4) transactions.

The TC1.6E DRB will buffer read data only. Any CPU store access which results in a hit in a DRB line will invalidate that DRB line and the store will continue as normal, bypassing the DRB. In addition, the execution of any CACHEA.xx or CACHEI.xx instruction by a TC1.6E CPU will invalidate all its DRB lines.

DRB Bypass

The DRB may be bypassed, under control of DCON0.DCBYP, to provide a direct data access path to memory. The default value of DCON0.DCBYP is such that the DRB is bypassed after reset.

Whilst the DRB bypass is enabled, a data read request by the CPU to a cacheable address will result in a forced buffer miss. Any valid lines within the DRB will remain valid and unchanged whilst the DRB is bypassed. As such data reads to cacheable addresses with the DRB bypass enabled behave identically to data accesses to non-cacheable addresses.

The DCON0 register is endinit protected.

5.11.3 DMI Trap Generation

CPU data accesses to the DMI may encounter one of a number of potential error conditions, which result in one of the following trap conditions being reported by the DMI.

ALN Trap

An ALN trap is raised for the following conditions:

- An access whose effective address does not conform to the alignment rules
- An access where the length, size or index of a circular buffer is incorrect

Whenever an ALN trap occurs, the DSTR (Data Synchronous Trap Register) and the DEADD (Data Error Address Register) CSFRs are updated.

MEM Trap

A MEM trap is raised for the following conditions:

- An access whose effective address has a different segment to that of the base address (Segment Difference Error)
- An access whose effective address causes the data to span two segments (Segment Crossing Error)
- A memory address is used to access a CSFR area (CSFR Access Error)
- A context load or store instruction is executed where the effective address is not within the local DSPR range (Context Location Error)(TC1.6E only)

Whenever a MEM trap occurs, the DSTR (Data Synchronous Trap Register) and the DEADD (Data Error Address Register) CSFRs are updated.

DSE Trap

A DSE trap is raised for the following conditions:

- An access outside the range of the DSPR (Scratch Range Error)
- An access to the lower half of segment C which cannot be translated into a global address, i.e. from C1000000 to C7FFFFFF (Global Address Error)
- An error on the bus for an external accesses due to a load (Load Bus Error)
- An error from the bus during a cache refill (Cache Refill Error)
- An error during a load whilst in SIST mode (Load MSIST Error)
- An error generated by the overlay system during a load.

Whenever a DSE trap occurs, the DSTR (Data Synchronous Trap Register) and the DEADD (Data Error Address Register) CSFRs are updated.

DAE Trap

A DAE trap is raised for the following conditions:

- An error on the bus for an external accesses due to a store (Store Bus Error)
- An error on the bus due to a cache writeback (Cache writeback Error)(TC1.6P only)
- An error from the bus due to a cache flush (Cache Flush Error)(TC1.6P only)
- An error due to a store whilst in SIST mode (Store MSIST Error)
- An error generated by the overlay system during a store.

Whenever a non-inhibited DAE trap occurs, the DATR (Data Asynchronous Trap Register) and the DEADD (Data Error Address Register) CSFRs are updated. DAE traps are inhibited if the DATR register is non-zero.

Data Memory Protection Traps

Data memory protection traps (MPW, MPR, MPP, MPN) are raised by the memory protection system when a protection violation occurs. Whenever a data memory protection trap occurs the DSTR (Data synchronous trap register) and the DEADD (Data Error Address Register) are updated.

5.11.4 DMI Registers

Two control registers and three trap flag registers control the operation of the DMI. These registers and their bits are described in this section.

5.11.4.1 DMI Register Descriptions

Note: There is no DCON1 register in this implementation.

Data Memory Control Register 0 (DCON0)

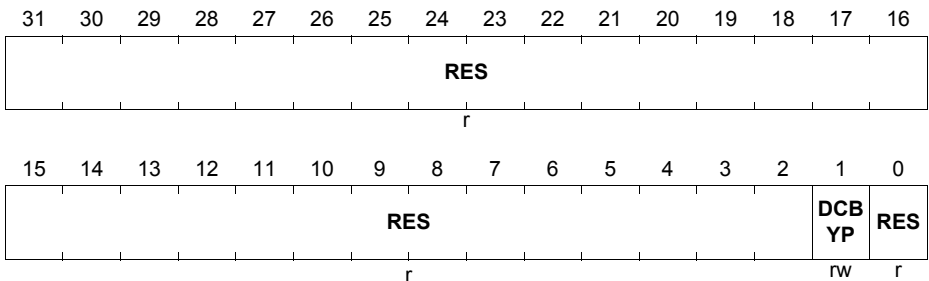
The DCON0 register allows the Data cache to be bypassed.

DCON0

Data Memory Control Register

(CSFR_Base + 9040_H)

Reset Value: 0000 0002_H



Field	Bits	Type	Description
RES	0	-	Reserved
DCBYP	1	rw	Data Cache/Data Read Buffer Bypass 0 _B DCache / DRB enabled 1 _B DCache / DRB Bypassed (disabled)
RES	[31:2]	-	Reserved

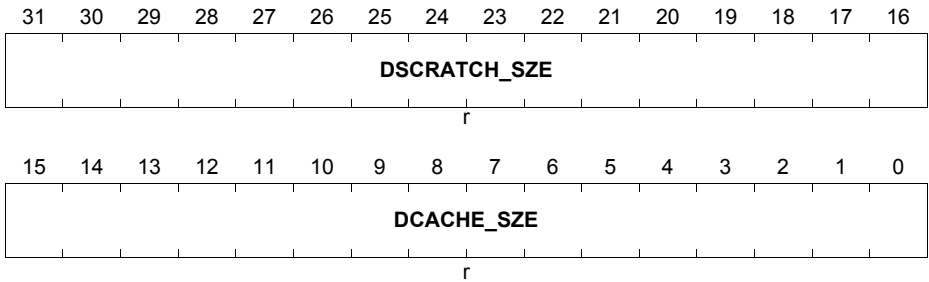
Note: This register is endinit protected

Data Control Register 2 (DCON2)

DCON2 contains size information for the Data memory system.

DCON2

Data Control Register 2 (CSFR_Base + 9000_H) Reset Value: CPU Dependent



Field	Bits	Type	Description
DCACHE_SIZE	[15:0]	r	Data Cache Size In KBytes
DSCRATCH_SIZE	[31:16]	r	Data Scratch Size In KBytes

DMI Synchronous Trap Flag Register

The DSTR contains synchronous trap information for the data memory system. The register is updated with trap source information to aid the localisation of faults.

The register is updated whenever a valid trap is detected and the register has no bits already set. It is cleared by a write (independent of data value).

DSTR
Data Synchronous Trap Register
(CSFR_Base + 9010_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						ALN	RES			CLE	MPE	CAC	SCE	SDE	
r						rwh	r			rwh	rwh	rwh	rwh	rwh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOE	DTME	RES					CRE	RES			LBE	GAE	SRE		
rwh	rwh	r					rwh	r			rwh	rwh	rwh		

Field	Bits	Type	Description
SRE	0	rwh	Scratch Range Error Data access to data scratch memory region outside of physically implemented memory
GAE	1	rwh	Global Address Error Load or store to local code scratch address outside of the lower 1MByte
LBE	2	rwh	Load Bus Error Data load from bus causing error
RES	[5:3]	r	Reserved
CRE	6	rwh	Cache Refill Error Bus error during cache refill
RES	[13:7]	r	Reserved
DTME	14	rwh	DTAG MSIST Error Access to memory mapped DTAG range outside of physically implemented memory
LOE	15	rwh	Load Overlay Error Load to invalid overlay address

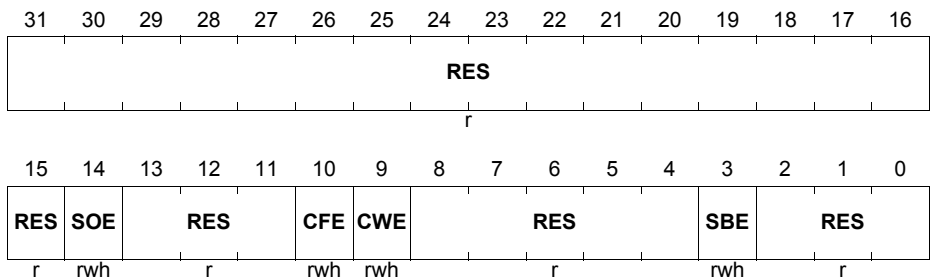
Field	Bits	Type	Description
SDE	16	rwh	Segment Difference Error Load or store access where base address is in different segment to access address
SCE	17	rwh	Segment Crossing Error Load or store access across segment boundary
CAC	18	rwh	CSFR Access Error Load or store to local CSFR space
MPE	19	rwh	Memory Protection Error Data access violating memory protection.
CLE	20	rwh	Context Location Error Context operation to invalid location
RES	[23:21]	r	Reserved
ALN	24	rwh	Alignment Error Data access causing alignment error
RES	[31:25]	r	Reserved

DMI Asynchronous Trap Flag Register

The DATR contains asynchronous trap information for the data memory system. The register is updated with trap information for DAE traps to aid the localisation of faults.

The register is updated whenever a valid trap is detected and the register has no bits already set. It is cleared by a write (independent of data value).

DAE traps are inhibited if the DATR register is non-zero.

DATR
Data Asynchronous Trap Register
(CSFR_Base + 9018_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RES	[2:0]	r	Reserved
SBE	3	rwh	Store Bus Error Data store to bus causing error
RES	[8:4]	r	Reserved
CWE	9	rwh	Cache Writeback Error Bus error during cache writeback operation
CFE	10	rwh	Cache Flush Error Bus error during cache flush operation
RES	[13:11]	r	Reserved
SOE	14	rwh	Store Overlay Error Store to invalid overlay address
RES	[31:15]	r	Reserved

Data Error Address Register

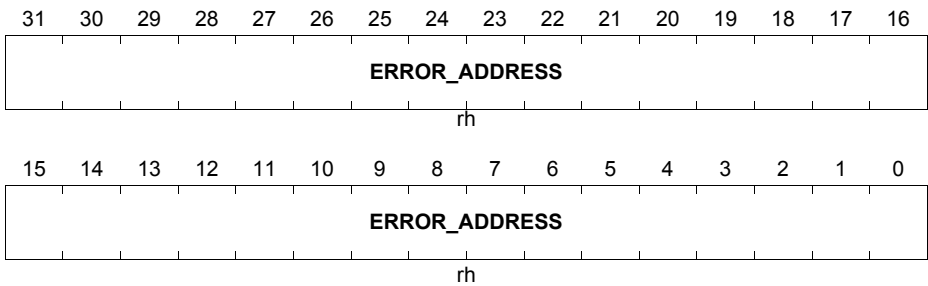
DEADD contains trap address information for the Data memory system. The register is updated with trap information for MEM, ALN, DSE or DAE traps to aid the localisation of faults.

The register is only set whenever a trap is detected and either the DATR or DSTR registers have no bits already set.

The register contents are only valid when either the DATR or DSTR register is non-zero and hence should be read prior to clearing these registers.

DEADD

Data Error Address Register (CSFR_Base + 901C_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
ERROR_ADDRESSES	[31:0]	rh	Error Address

5.12 Safety Features

The CPUs implements a number of safety concepts These are detailed in the following sections.

5.12.1 SRI Data Master Address Phase Error Injection

To allow the SRI address phase error detection system to be tested it is necessary to inject errors during the address phase of an SRI transaction. This is done by the TC1.6P or TC1.6E data master. The data master selectively inverts individual bits of the address phase of an SRI read or write phase ECC packet and is controlled using the SEGEN register. The SEGEN register contains three fields:- an enable (AE), an error descriptor (ADTYPE) and a bit flip (ADFLIP) field. When enabled for an address phase error the address ECC bits indicated by the flip field are inverted for the next SRI data read or write bus transaction performed by the DMI. Following the transaction the enable bit is cleared by hardware. This mechanism allows selected bits of the SRI address ECC to be corrupted for a single transaction. The SEGEN register is ENDINIT protected.

5.12.2 SRI Data Master Write Phase Error Injection

To allow the SRI data phase error detection system to be tested it is necessary to inject errors during the data phase of an SRI write transaction. This is done by the TC1.6P or TC1.6E data master. The data master selectively inverting individual bits of the SRI write phase ECC packet and is controlled using the SEGEN register. The SEGEN register contains three fields:- an enable (AE), an error descriptor (ADTYPE) and a bit flip (ADFLIP) field. When enabled for a data phase error the data phase ECC bits indicated by the flip field are inverted for the next SRI write bus transaction performed by the DMI. Following the transaction the enable bit is cleared by hardware. This mechanism allows selected bits of the SRI data ECC to be corrupted for a single transaction. The SEGEN register is ENDINIT protected.

5.12.3 SRI Data Slave Read Phase Error Injection

To allow the SRI data phase error detection system to be tested it is necessary to inject errors during the data phase of an SRI read transaction. This is done by the TC1.6P or TC1.6E slave. The data slave selectively inverts individual bits of the SRI read phase ECC packet and is controlled using the SEGEN register. The SEGEN register contains three fields:- an enable (AE), an error descriptor (ADTYPE) and a bit flip (ADFLIP) field. When enabled for a data phase error the data phase ECC bits indicated by the flip field are inverted for the next SRI read bus transaction performed by the DMI. Following the transaction the enable bit is cleared by hardware. This mechanism allows selected bits of the SRI data ECC to be corrupted for a single transaction. The SEGEN register is ENDINIT protected.

5.12.4 SRI Error Capture

The SRI master and slave interfaces of the CPU implement the standard SRI ECC system. Error information detected during SRI transactions at the SRI master and slave interfaces is captured in the PIETR and PIEAR, or DIETR and DIEAR registers. On detection of an error the error information is captured and the relevant IED bit is set. No further error information is captured until this bit is cleared by software.

Error conditions at the SRI program interfaces are notified to the Safety Monitor Module via the `pbus_err_o` output from the CPU.

Error conditions at the SRI data interfaces are notified to the Safety Monitor Module via the `dbus_err_o` output from the CPU.

If an error is detected at an SRI slave interface during a write operation then the erroneous data will not be written.

5.12.5 SRI Safe Data Master tag

In order to differentiate between data accesses from safe and regular tasks a new safe task identification bit is introduced into the PSW register (PSW.S).

An SRI data access performed by a task when the PSW.S bit is set uses the safe data master tag. When this bit is not set the regular data master tag is used for the access. There is only one program master tag. This is used for SRI program fetches by both safe and regular tasks.

The initial value of the PSW.S bit for interrupt handlers is defined by the SYSCON.IS bit. The initial value of the PSW.S bit for trap handlers is defined by the SYSCON.IT bit.

On a trap the save of the current context is performed using the data master tag defined by SYSCON.TS

On an interrupt the save of the current context is performed using the data master tag defined by SYSCON.IS

5.12.6 Safety Memory Protection

The CPUs each allow for the definition of eight, protected regions of local SRAM memory.

The protection applies only to write or read-modify-write accesses to local SRAM included in the DMI or PMI from the bus.

Read accesses are not protected.

The protection scheme is based on the use of SRI tags to identify the master attempting the access and allows for a six bit tag individually identifying up to 64 masters. 32 masters are supported in the current implementation.

Each region is defined using the registers, `SPROT_RGNLx (x=0-7)` to define the lower address of the region, `SPROT_RGNUx (x=0-7)` to define the upper address of the

CPU Subsystem

region and `SPROT_RGNACCENx` ($x=0-7$) to individually select the master tags permitted write access to the defined address range.

A write or read-modify-writes access is seen as valid if the master tag of the access is enabled in the `SPROT_RGNACCENx` register and the address of the access satisfies the following relationship:-

`SPROT_RGNLax <= address < SPROT_RGNUAx`

The `SPROT_RGNLax`, `SPROT_RGNUAx`, `SPROT_RGNACCENx` registers are protected by the `safety_endinit` signal.

After reset, the region address registers will be set to include the whole of the CPUs SRAM address space and write access by all masters will be enabled.

When altering protection settings, it should be noted that, due to access pipelining and resynchronization delays in the register block, a write to a memory address affected by the protection change occurring immediately after the register write initiating the change may, or may not, be affected by the changed settings.

Whenever a Safety Memory Protection violation occurs the event is notified to the Safety Management Unit via the `tc16_safe_prot_err_o` output signal from the CPU. The `PIETR/DIETR` and `PIEAR/DIEAR` registers are updated to aid localisation of the error

5.12.7 Safety Register Protection

The CPUs implement the standard memory protection scheme for peripheral registers using the `SPROT_ACCEN` register. This allows all CPU CSFR and SFR registers to be protected from corruption by untrusted masters. The `SPROT_ACCEN` register defines which masters may modify the SFR and CSFR registers via bus access through the SRI slave interface.

The `SPROT_ACCEN` register is protected by the `safety_endinit` signal.

In all cases the master is identified using the SRI tag of the access. See On Chip Bus chapter for the product's TAG ID to master peripheral mapping.

Whenever a Safety Register Protection violation occurs the event is notified to the Safety Management Unit via the `safe_prot_err_o` output signal from the CPU. The `PIETR` and `PIEAR` registers are updated to aid localisation of the error.

5.12.8 Lock Step Implementation

The TC1.6E and TC1.6P CPUs may be implemented as standard CPU (as described in the chapter) or as a checker CPU for use in lockstep comparison system. The checker CPU implementation has the following features.

- No Memories

CPU Subsystem

- All signals that would normally be fed to the memories become additional outputs from the checker CPU.
- All signals that would normally be fed from the memories become additional inputs to the checker CPU.
- All outputs from the checker CPU are inverted.

The lockstep system compares the following CPU interfaces.

- Interface to the interrupt router.
- Interface to the SCU.
- Interface to the DMI Data Scratch pad memory.
- Interface to the PMI Program Scratch memory.
- Interface to the PMI Program Cache memory.
- Interface to the PMI Program Tag memory.
- Master interface to the Peripheral bus.
- Master interface to the SRI bus for PMI accesses.
- Master interface to the SRI bus for DMI accesses.
- Slave accesses from the SRI bus for PMI and DMI accesses.
- The CPUs run in lockstep mode at the full specified frequency. Enabling lockstep does not require operational frequency to be reduced.
- The Checker CPU is implemented using inverse state storage "AntiCore".

5.12.9 MBIST usage recommendations

The following usage of the CPU SRAM MBIST system is recommended.

- When any one of the CPU local data memories (DSPR/DCACHE or DTAG) is in MBIST test mode (MEMTEST enabled), then other CPU local data memories will be not be accessible. It is therefore recommended to enable DSPR/DCACHE MBIST mode and DTAG memory MBIST mode together.
- When any one of the CPU local program memories (PSPR/PCACHE or PTAG) is in MBIST test mode (MEMTEST enabled), then other CPU local program memories will be not be accessible. It is therefore recommended to enable PSPR/PCACHE MBIST mode and PTAG memory MBIST mode together.
- For configurations that use dual MBIST controllers for a single logical memory (e.g TC29x DSPR/DCACHE for CPU1,2), MBIST tests run on one CPU will report errors to both controllers. It is therefore recommended to enable MBIST mode on both controllers together and that the error notification bits of both controllers are cleared at the end of testing.

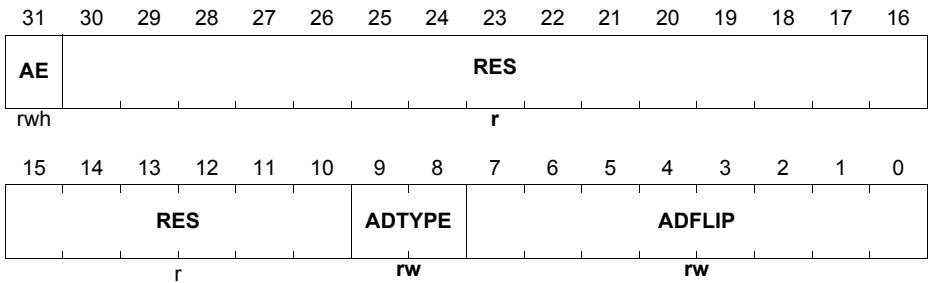
5.12.10 Registers Implementing Safety Features

SRI Error Generation Register

The SEGEN register controls the injection of SRI address phase errors from the DMI. This register is ENDINIT protected.

SEGEN

SRI Error Generation Register (CSFR_Base + 1030_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
ADFLIP	[7:0]	rw	Address ECC Bit Flip SRI address ECC Bits to be flipped on the next read or write transaction from the DMI when enabled by AE. 0 _B No Flip 1 _B Flip
ADTYPE	[9:8]	rw	Type of Error 00 _B Data Master Address Phase 01 _B Data Master Write Data 10 _B Data Slave Read Data 11 _B Reserved
RES	[30:10]	r	Reserved
AE	31	rwh	Activate Error Enable Enabled the selective inverting of SRI ECC packet bits defined by ADFLIP. This bit will be cleared by hardware after the next SRI read or write transaction from the DMI. 0 _B Not Enabled 1 _B Enabled

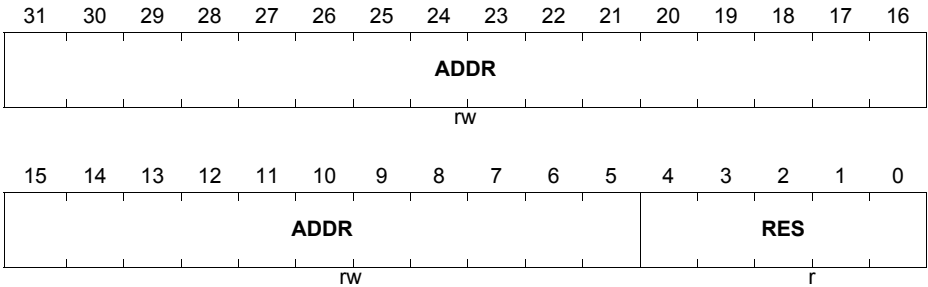
Safety Protection Region Lower Address Register

In conjunction with the associated SPROT_RGNUAx and SPROT_RGNACCENx registers, the SPROT_RGNLA register provides control of a memory protection region. SPROT_RGNLAx defines the lower address of a region of memory, SPROT_RGNUAx defines the upper address and the SPROT_RGNACCENx registers define the SRI tags allowed write access to the region. Address ranges can be set to be larger than the SRAM address space but only accesses to the SRAM are affected by these registers. The minimum resolution of the comparison logic is 32_D bytes so address bits 4_D down to 0_D are not used

SPROT_RGNLAx (x=0-7)

Safety Protection Region Lower Address Register

(SFR_Base + E000_H+x*10_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
RES	[4:0]	r	Reserved
ADDR	[31:5]	rw	Region Lower Address Bits 32 to 5 of the address which is the lower bound of the defined memory region

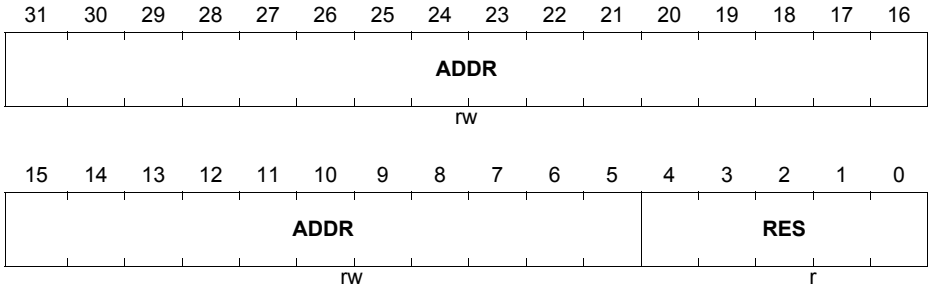
Safety protection Region Upper Address Register

In conjunction with the associated SPROT_RGNLAx and SPROT_RGNACCENx registers, the SPROT_RGNUAx register provides control of a memory protection region. SPROT_RGNUAx defines the upper address of the memory region

SPROT_RGNUAx (x=0-7)

Safety protection Region Upper Address Register

(SFR_Base + E004_H+x*10_H) Reset Value: FFFF FFE0_H



Field	Bits	Type	Description
RES	4:0	r	Reserved Unused bits will always read as 0 _B . Written value will be ignored
ADDR	31:5	rw	Region Upper Address Bits 31 to 5 of the address which is the upper bound of the defined memory region

Safety Region Access Enable Register A

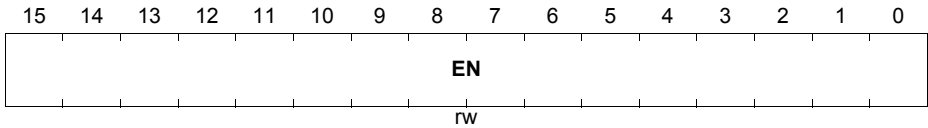
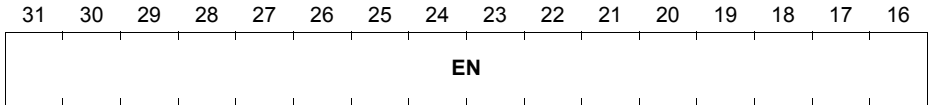
The Access Enable Register A controls write access for transactions to the protected memory region with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The Safety protection is prepared for an 6 bit TAG ID. The registers SPROT_RACCEN0 / SPROT_RACCEN1 are provide one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ... ,EN31 -> TAG ID 011111_B.

SPROT_RGNACCENAx (x=0-7)

Safety Protection Region Access Enable Register A

(SFR_Base+ E008_H+x*10_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
EN	[31:0]	rw	<p>Access Enable for Master TAG ID n (n=0-31)</p> <p>This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n</p> <p>0_B Write access will not be executed</p> <p>1_B Write access will be executed</p>

Safety Protection Region Access Enable Register B

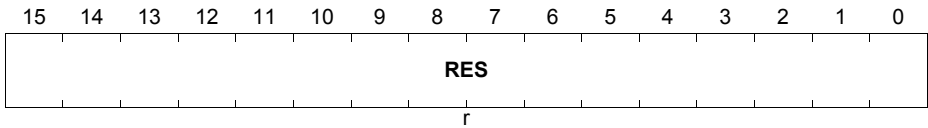
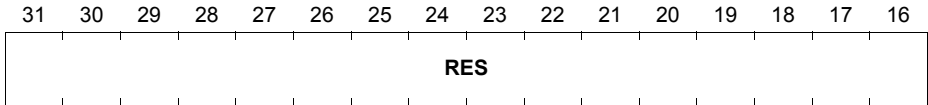
The Access Enable Register B controls write access for transactions to the protected memory region with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The CPU is prepared for an 6 bit TAG ID. The registers SPROT_RACCEN0 / SPROT_RACCEN1 are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to SPROT_ACCENB.ENx: EN0 -> TAG ID 100000_B, EN1 -> TAG ID 100001_B, ... ,EN31 -> TAG ID 111111_B.

SPROT_RGNACCENBx (x=0-7)

Safety Protection Region Access Enable Register B (SFR_Base + E00C_H+x*10_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RES	[31:0]	r	Reserved Read as 0; should be written with 0.

Safety Register Access Enable Register A

The Access Enable Register A controls write access for transactions to CSFR/SFR registers with the on chip bus master TAG ID 00000_B to 01111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The Safety protection is prepared for an 6 bit TAG ID. The registers SPROT_RACCEN0 / SPROT_RACCEN1 are provide one enable bit for each possible 6 bit TAG ID encoding.

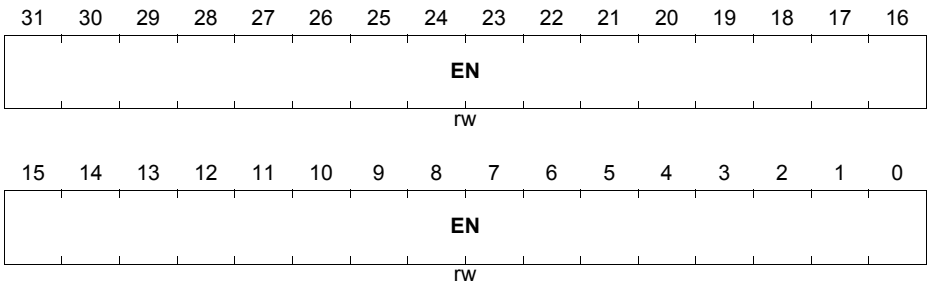
Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 00000_B, EN1 -> TAG ID 00000_B, ... ,EN31 -> TAG ID 01111_B.

SPROT_ACCENA

Safety Protection Register Access Enable Register A

(SFR_Base + E100_H)

Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
EN	[31:0]	rw	Access Enable for Master TAG ID n (n= 0-31) This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Safety Protection Register Access Enable Register B

The Access Enable Register B controls write access for transactions to CSFR/SFR registers with the on chip bus master TAG ID 10000_B to 11111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The CPU is prepared for an 6 bit TAG ID. The registers SPROT_RACCEN0 / SPROT_RACCEN1 are providing one enable bit for each possible 6 bit TAG ID encoding.

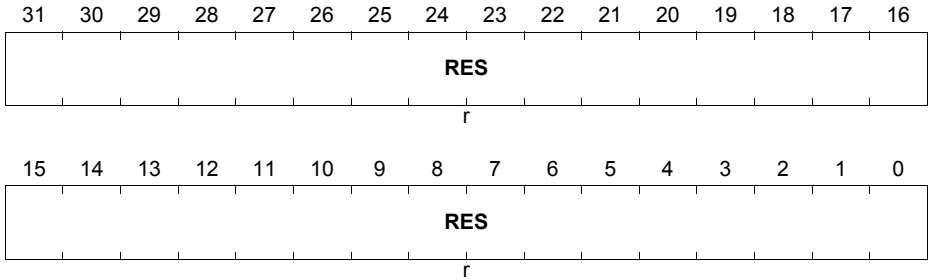
Mapping of TAG IDs to SPROT_ACCENB.ENx: EN0 -> TAG ID 10000_B, EN1 -> TAG ID 10000_B, ... ,EN31 -> TAG ID 11111_B.

SPROT_ACCENB

Safety Protection Region Access Enable Register B

(SFR_Base + E104_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RES	[31:0]	r	Reserved Read as 0; should be written with 0.

6 Lockstep Comparator Logic (LCL)

The Lockstep Comparator Logic module provides access to the control and self test functions of the processor lockstep comparators.

6.1 Feature List

An overview of the features implemented in the LCL follows:

- Monitoring the core comparators for the lockstep core and its shadow and flagging any detected differences
- Running background, continuous self test on the lockstep comparators to validate the correct operation of the logic.

6.2 Lockstep Control

The lockstep control function is enabled by the LSEN bitfield in a control register in the SCU. Each core capable of lockstep has its own instance of the control register. In this product, the CPU1 instance of the Tricore can be lockstepped so there is one register, LCLCON1 for CPU1.

These registers are only initialised by a cold power-on reset. In this initialisation state, all lockstepped processors in the system will have lockstep enabled. The lockstep function can only be disabled by the system initialisation software writing a 0_b to the LSEN bitfield. Application software cannot enable or disable the lockstep function.

The current mode of the lockstep logic can be monitored by reading the lockstep status bit, LS, in the associated LCLCON register.

Writes to the control registers will be subject to the protection mechanisms of the SCU.

6.3 Lockstep Monitoring

The lockstep monitoring function will compare the outputs from the master and checker cores and report that a failure has occurred to the Safety Management Unit (SMU) for appropriate action.

The monitoring function temporally separates the cores by inserting synchronisation delays to re-align the signals being compared. To achieve this, the checker core inputs and the master core outputs fed to the comparators are delayed by two clock cycles.

Lockstep Comparator Logic (LCL)

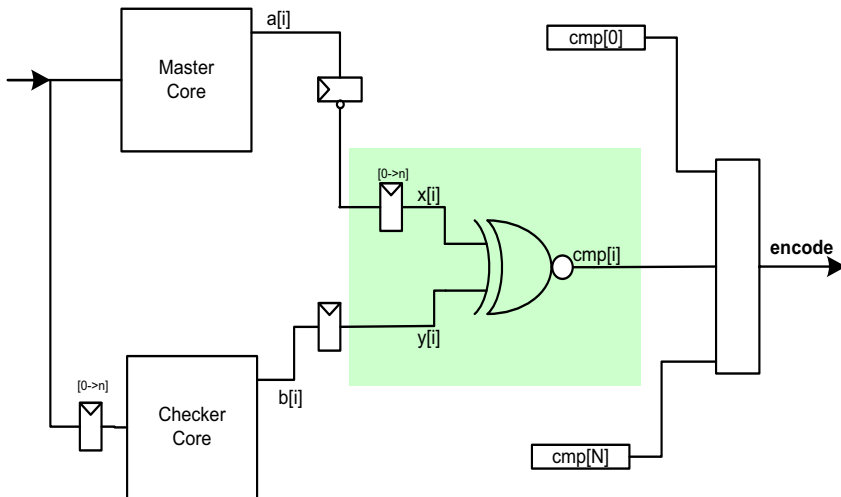


Figure 6-1 Node Comparator

Figure 6-1 above shows the equivalent circuit of an arbitrary node comparator, i , of 0 to N comparators. The comparator monitors two signals, a and b , which are connected to the same node in the master and checker cores. The signals can be synchronised in the relevant core using the core clock to minimise impact on the core timing.

After the optional synchronisation, the master core monitor point is inverted to reduce the risk of a common mode failure in the two monitored signals. Therefore, x is equivalent to a delayed by a n clock cycles, where $n=2$ in this implementation, to allow for the temporal shift between the master and checker cores and y is equivalent to b . If the nodes differ (i.e. x and y are the same), the CMP signal is set to 1_B . This will cause the **encode** signal to flag the failing node and the failure will be detected. In the event that multiple nodes fail, the encode output will be the minimum and maximum values of all the indices, i , of the failing nodes. This has no impact on normal functioning but allows the self test logic to detect a real failure occurring in the same clock cycle that a fault is injected for test purposes.

6.4 Lockstep Self Test

Each core capable of lockstep also has a continuously running background self test of the lockstep comparator.

The self test function will inject faults into both inputs of each of the monitored nodes and verify that the fault is correctly detected by the monitoring logic.

In the event of a self test failure being detected, the failure will be reported to the Safety Management Unit for an appropriate response.

Lockstep Comparator Logic (LCL)

An equivalent circuit of a node comparator showing the fault injection logic is shown in [Figure 6-2](#)

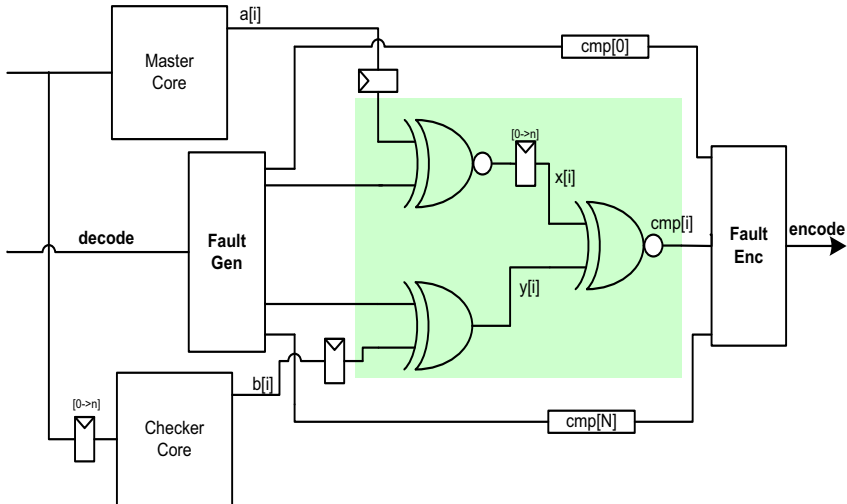


Figure 6-2 Node Comparator with Fault Injection

Faults are injected using the **Fault Gen** block. This will use a binary number, **decode**, to calculate which node is to be tested. **decode** will be generated from a free running, binary counter using gray code number representation (the **input counter**). The Self test circuit will test every node once every 8192 clock cycles. Alternate test cycles will inject faults into either the **a** or **b** side of the comparator node. The complete self test cycle will therefore repeat every 16384 clock cycles.

Injecting a fault into either side of the comparator node will cause that node to fail unless a real fault occurs in the same clock cycle in which case the two faults will cancel out.

The node failing is processed by the **Fault Enc** block to generate a binary representation of the failing node, **encode**. This contains two numbers, the node index of the first node found to be failing counting up from the lowest minimum node index and the node index of the first node found to be failing counting down from the maximum node index.

If the lockstep block is functioning correctly, both values in **encode**, will either be 0 if no failures have been detected or the number of the node which has had a fault induced by the self test logic.

The values in **encode** are checked against a second, independent binary counter (the **monitor counter**). The **monitor counter** is also compared against the value of the **input counter**. In the event that either of the values in **encode** or the **input counter** fails to match the value of the monitor counter, a failure condition will be flagged to the SMU.

Lockstep Comparator Logic (LCL)

With this implementation, any of the following conditions will cause a self test fail:

- an actual fault occurring on any of the **a** or **b** nodes
- a stuck at 0_B fault occurring on any of the **cmp** nodes
- a stuck at 1_B fault occurring on any of the **cmp** nodes
- a failure in the **Fault Gen** block causing an incorrect or no fault to be injected
- a failure in the **Fault Enc** block causing an incorrect detection or no fault to be detected
- a stuck at fault on **x** (assuming that an injected fault does not always coincide with a masking pulse on **b**)
- a stuck at fault on **y** (assuming that an injected does not always coincide with a masking pulse on **a**)
- an actual fault on any of the **a** or **b** nodes coinciding with an injected fault
- a soft error occurring on either the **input counter** or **monitor counter**.

6.5 Lockstep Failure Signalling Test

The lockstep comparator allows a failure to be injected into one of the comparator nodes to allow the signalling of failures to be verified. A failure can be injected by writing 1_B to the LCLT1 bitfield of the LCLTEST register. The failure will be injected for a single cycle of the SPB clock. It is not necessary to write a 0_B to clear the test.

6.6 Functional Redundancy

All registers in the lockstep block which are not capable of being directly monitored for correct operation by the self test function will be duplicated. In the event of the duplicated registers not storing the same state, an error will be flagged to the SMU.

6.7 Revision History

Deltas from Lockstep version 1.1 to version 1.2

- Bitfield names updated for LCLCON register. See [sections “Lockstep Monitoring” and “Lockstep Self Test”](#).
- Multiple failure detecton algorithm updates. See Lockstep Self Test section.

Deltas from Lockstep version 1.2 to 1.3

- Revision History Added

Deltas from Lockstep version 1.3 to 1.4

- Description of self test updated

Deltas from Lockstep version 1.4 to 1.5

- Control bits for self test and delay removed. Both functions will now be permanently enabled.
- Lockstep will now be on after reset and cannot be enabled once switched off

Deltas from Lockstep version 1.5 to 1.6

- Additional control bit added to allow injection of fault condition under software control.

Deltas from Lockstep version 1.6 to 1.7

- Specifying cold power-on reset / typo's.

Deltas from Lockstep version 1.7 to 1.8

- No functional change, updates to Requirements tags only.

Deltas from Lockstep version 1.8 to 1.9

- No functional change, correction of typos and support for TC23x configuration.

7 System Control Units

The System Control Cluster comprises various modules which each handle system control functions.

The System Control Cluster includes the following functional modules:

- Clock Control Unit (CCU)
- Reset Control Unit (RCU)
- Power Management Controller (PMC)
- System Control Unit (SCU), including ERU, LCL, WDT, EMS, NMI and OVC registers

7.1 Clocking and Clock Control Unit (CCU)

This section describes the TC26x B-Step clock system, its configuration and the principles upon which it is based.

The clock system itself is build up as chain composing out of different building blocks which allow different certain function parts for this complete chain.

Building blocks are:

- Basic clock generation (Clock Source)
- Clock speed upscaling
- Clock distribution
- Individual clock configuration

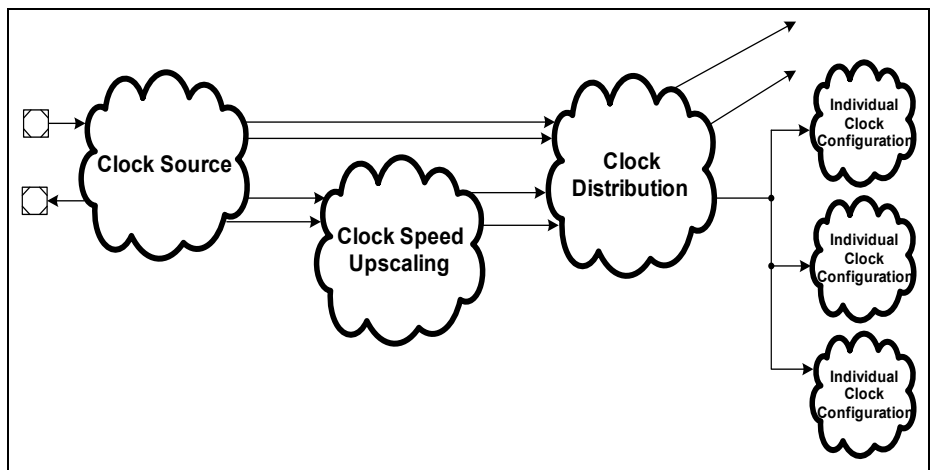


Figure 7-1 Clock Tree

Aside from the pure clock generation options there are several support functions which have been integrated in order to support easy and convenient controls.

In the following subsections the clock tree is explained from left to the right. Using this flow for the initial configuration is also recommended. Expert users can of course execute the configuration in a individual order.

Note: If a running system needs to be reconfigured not all parts of the clock tree need to be configured, only these parts that are required can be updated.

7.1.1 Clock Sources

There are several clock sources available which either source the complete device, a major or minor part, or only dedicated modules. This depends on the sources and the configuration.

Please note that several clock sources can be used in parallel inside the system but the main function of each peripheral is only related to one source at any time.

7.1.1.1 Oscillator Circuit (OSC)

The oscillator circuit, a Pierce oscillator, is designed to work with both an external crystal / ceramic resonator or an external stable clock source, consists of an inverting amplifier with XTAL1 as input, and XTAL2 as output with an integrated feedback resistor.

External Input Clock Mode

When using an external clock signal it must be connected to XTAL1. XTAL2 is left open (unconnected).

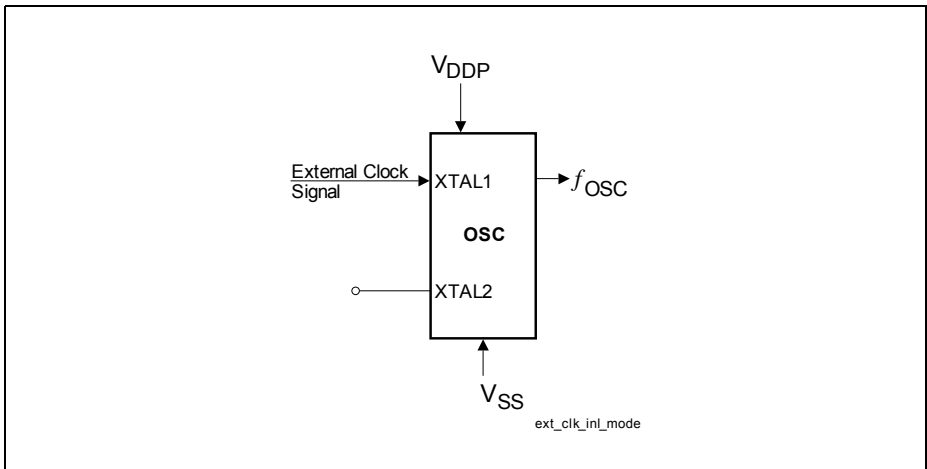


Figure 7-2 TC26x B-Step Direct Clock Input

When supplying the clock signal directly, not using an external crystal / ceramic resonator and bypassing the oscillator, the input frequency needs to be equal or greater than PLL VCO input frequency (the value is listed in the Data Sheet) if used in normal Mode.

External Crystal / Ceramic Resonator Mode

Figure 7-3 shows the recommended external circuitries for both operating modes, External Crystal / Ceramic Resonator Mode with and without external components.

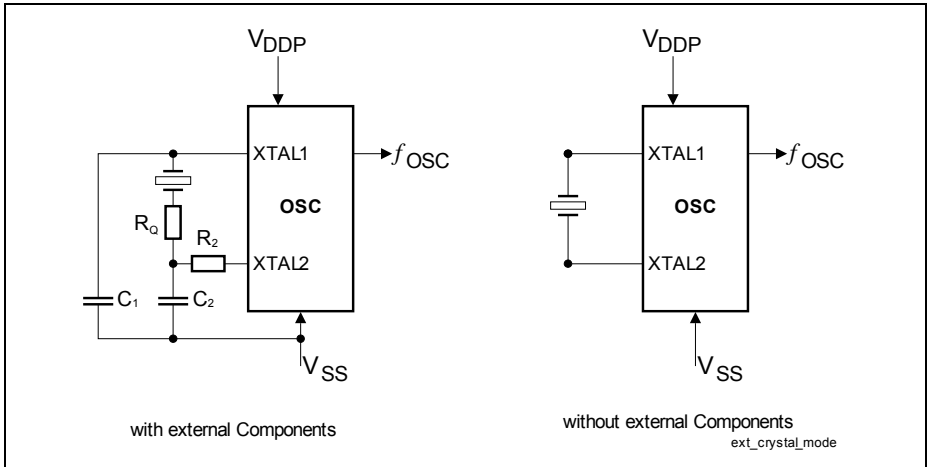


Figure 7-3 External Circuitry for Crystal / Ceramic Resonator operation

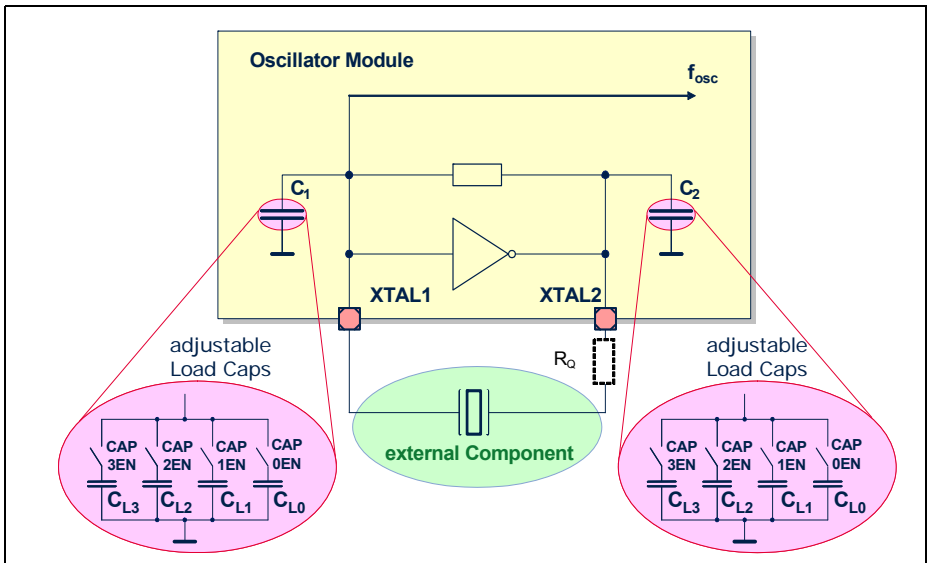


Figure 7-4 Circuitry for Crystal / Ceramic Resonator operation with internal CAPS

When using an external crystal / ceramic resonator, its frequency can be within the allowed range (the values are listed in the Data Sheet). An external oscillator load

circuitry can be used, connected to both pins, XTAL1 and XTAL2. Additionally are necessary, two load capacitances C_1 and C_2 (see [Figure 7-3](#)) or the internal loads can be used (see [Figure 7-4](#)), and depending on the crystal / ceramic resonator type, a series resistor R_2 to limit the current. A test resistor R_Q may be temporarily inserted to measure the oscillation allowance (negative resistance) of the oscillator circuitry. R_Q values are typically specified by the crystal / ceramic resonator vendor. The C_1 and C_2 values shown in the Data Sheet can be used as starting points for the negative resistance evaluation and for non-productive systems. The exact values and related operating range are dependent on the crystal / ceramic resonator frequency and have to be determined and optimized together with the crystal / ceramic resonator vendor using the negative resistance method. Oscillation measurement with the final target system is strongly recommended to verify the input amplitude at XTAL1 and to determine the actual oscillation allowance (margin negative resistance) for the oscillator-crystal / ceramic resonator system.

The oscillator can also be used in combination with a ceramic resonator. The final circuitry must be also verified by the resonator vendor.

Oscillator Circuit Control Register
OSCCON
OSC Control Register

 (010_H)

 Reset Value: 0000 0X1X_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			CAP 3EN	CAP 2EN	CAP 1EN	CAP 0EN	APR EN	0			OSCV _{VAL}				
r			rw	rw	rw	rw	rw	r			rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			X1D EN	X1D	0	PLL HV	SH BY	MODE		GAINSEL	OSC RES	PLL LV	0		
r			rw	rh	r	rh	rw	rw		rw	w	rh	r		

Field	Bits	Type	Description
PLLLV	1	rh	Oscillator for PLL Valid Low Status Bit This bit indicates if the frequency output of OSC is usable for the VCO part of the PLL. This is checked by the Oscillator Watchdog of the PLL. 0 _B The OSC frequency is not usable. Frequency f_{REF} is too low. 1 _B The OSC frequency is usable <i>Note: The default value depends on the status of the clock received from pin XTAL1.</i>
OSCRE_S	2	w	Oscillator Watchdog Reset 0 _B The Oscillator Watchdog of the PLL is not cleared and remains active 1 _B The Oscillator Watchdog of the PLL is cleared and restarted <i>Note: Always read as zero.</i>

Field	Bits	Type	Description
GAINSEL	[4:3]	rw	<p>Oscillator Gain Selection</p> <p>In Normal Mode this value should not be changed from the reset value 11_B.</p> <p>00_B The gain control is configured for frequencies from 4 MHz to 8 MHz</p> <p>01_B The gain control is configured for frequencies from 4 MHz to 16 MHz</p> <p>10_B The gain control is configured for frequencies from 4 MHz to 20 MHz</p> <p>11_B The gain control is configured for frequencies from 4 MHz to 25 MHz</p> <p><i>Note: Default value after reset is 11_B.</i></p>
MODE	[6:5]	rw	<p>Oscillator Mode</p> <p>This bit field defines which mode can be used and if the oscillator entered the Power-Saving Mode or not.</p> <p>00_B External Crystal / Ceramic Resonator Mode and External Input Clock Mode. The oscillator Power-Saving Mode is not entered.</p> <p>01_B OSC is disabled. The oscillator Power-Saving Mode is not entered.</p> <p>10_B External Input Clock Mode and the oscillator Power-Saving Mode is entered</p> <p>11_B OSC is disabled. The oscillator Power-Saving Mode is entered.</p> <p><i>Note: Default value after reset is 00_B.</i></p>
SHBY	7	rw	<p>Shaper Bypass</p> <p>0_B The shaper is not bypassed</p> <p>1_B The shaper is bypassed</p> <p><i>Note: Default value after reset is 0_B.</i></p>
PLLHV	8	rh	<p>Oscillator for PLL Valid High Status Bit</p> <p>This bit indicates if the frequency output of OSC is usable for the VCO part of the PLL. This is checked by the Oscillator Watchdog of the PLL.</p> <p>0_B The OSC frequency is not usable. Frequency f_{OSC} is too high.</p> <p>1_B The OSC frequency is usable</p> <p><i>Note: The default value depends on the status of the clock received from pin XTAL1.</i></p>

Field	Bits	Type	Description
X1D	10	rh	XTAL1 Data Value This bit monitors the value (level) of pin XTAL1. If XTAL1 is not used as clock input it can be used as GPI pin. This bit is only updated if X1DEN is set. <i>Note: The default value depends on the status of the clock received from pin XTAL1.</i>
X1DEN	11	rw	XTAL1 Data Enable 0 _B Bit X1D is not updated 1 _B Bit X1D can be updated <i>Note: Default value after reset is 0_B.</i>
OSCVL	[20:16]	rw	OSC Frequency Value This bit field defines the divider value that generates the reference clock that is supervised by the oscillator watchdog. f_{OSC} is divided by OSCVAL + 1 in order to generate f_{OSCREF} .
APREN	23	rw	Amplitude Regulation Enable 0 _B Amplitude Regulation is disabled 1 _B Amplitude Regulation is enabled
CAP0EN	24	rw	Capacitance 0 Enable 0 _B Capacitance C_{L0} is disabled 1 _B Capacitance C_{L0} is enabled <i>Note: Total capacitance for each XTAL1 and XTAL2 is the sum of the enabled capacitance 0 to 3.</i>
CAP1EN	25	rw	Capacitance 1 Enable 0 _B Capacitance C_{L1} is disabled 1 _B Capacitance C_{L1} is enabled <i>Note: Total capacitance for each XTAL1 and XTAL2 is the sum of the enabled capacitance 0 to 3.</i>
CAP2EN	26	rw	Capacitance 2 Enable 0 _B Capacitance C_{L2} is disabled 1 _B Capacitance C_{L2} is enabled <i>Note: Total capacitance for each XTAL1 and XTAL2 is the sum of the enabled capacitance 0 to 3.</i>

Field	Bits	Type	Description
CAP3EN	27	rw	Capacitance 3 Enable 0_B Capacitance C_{L3} is disabled 1_B Capacitance C_{L3} is enabled <i>Note: Total capacitance for each XTAL1 and XTAL2 is the sum of the enabled capacitance 0 to 3.</i>
0	0, 9, [15:12] , [22:21] , [31:28]	r	Reserved Read as 0; should be written with 0.

Configuration of the Oscillator

A configuration of the oscillator is always required before an external crystal / ceramic resonator could be used as clock source. For this start-up configuration two options are supported:

- Configuration via the SSW
- Configuration after the execution of the SSW

Configuration via SSW

This option is enabled when bit FLASH0_PROCOND.OSCCFG is set. In this mode the control information for the register OSCCON is loaded from FLASH0_PROCOND by the SSW. Therefore the required information needs to be stored with the UCB_DFlash at offset 00_H.

In this mode OSCCON.MODE and OSCCON.CAPxEN together with bit OSCCON.APREN are controllable.

Note: A control for OSCCON.GAINSEL is not required as it is recommended always to keep the default configuration of 11_B.

If the oscillator was disabled it must be enabled by setting bit field OSCCON.MODE = 00_B.

If the integrated capacitors should be used this can be enabled via the bits OSCCON.CAPxEN. If OSCCON[27:24] ≠ 0000_B then bit OSCCON.APREN need to be set too. Please note that the Amplitude Regulation needs to be enabled always when integrated caps are used. Enabling the Amplitude Regulation does not require a change for OSCCON.GAINSEL, which should be left at 11_B.

Configuration after SSW

After the optional configuration in the SSW there is always an option to configure the oscillator in the application software.

This is done via register OSCCON. The register itself is Safety ENDINIT protected.

In general the same rules apply for configuration as described in the section before.

Miscellaneous Oscillator Features

Support for two additional features which are related to the oscillator are also located in the OSCCON register.

XTAL1 as General Purpose Input Pin

For the rare case that XTAL1 / 2 are not used for clocking purpose XTAL1 could be used as an input pad instead. Setting bit OSCCON.X1DEN enables this function and the input value can be read thereafter via bit OSCCON.XD1.

Please note that this defines only a basic input function and supports no output features for XTAL1 or XTAL2.

Note: For XTAL2 no input function is supported.

Note: If XTAL1 is used for clock functions bit OSCCON.X1DEN should be cleared, as otherwise bit OSCCON.X1D would toggle with the input clock frequency.

Oscillator Watchdog

In combination with the PLL, a monitoring function is implemented. This feature is defined to detect severe malfunctions of an external crystal / ceramic resonator. Detectable errors are loss of clock input or a much too high input frequency.

The oscillator watchdog monitors the incoming clock frequency f_{OSC} from OSC. A stable and defined input frequency is a mandatory requirement for operation in both Prescaler Mode and Normal Mode of the PLLs. For operation in Freerunning Mode no f_{OSC} input frequency is required. Therefore this mode is selected automatically after each System Reset. In addition for the Normal Mode it is required that the input frequency f_{OSC} is in a certain frequency range to obtain a stable master clock from the VCO part.

The expected input frequency is selected via the bit field OSCCON.OSCVAL. The OSC_WDT checks for frequencies which are too low or too high.

The frequency that is monitored is f_{OSCREF} which is derived from f_{OSC} .

(7.1)

$$f_{OSCREF} = \frac{f_{OSC}}{OSCVAL + 1}$$

The divider value OSCCON.OSCVAL has to be selected in a way that f_{OSCREf} is 2.5 MHz.

Note: f_{OSCREf} has to be within the range of 2 MHz to 3 MHz and should be as close as possible to 2.5 MHz.

Before configuring the OSC_WDT function all the SMU Oscillator Watchdog alarm response options should be disabled in order to avoid unintended traps. Thereafter the value of OSCCON.OSCVAL can be changed. Then the OSC_WDT should be reset by setting OSCCON.OSCRES. This requests the start of OSC_WDT monitoring with the new configuration. When the expected positive monitoring results of OSCCON.PLLLV and / or OSCCON.PLLHV are set the input frequency is within the expected range. As setting OSCCON.OSCRES clears both bits OSCCON.PLLLV and OSCCON.PLLHV all both status flags will be set. Therefore both flags should be cleared before the SMU alarm responses are enabled again. The trap disabling-clearing-enabling sequence should also be used if only bit OSCCON.OSCRES is set without any modification of OSCCON.OSCVAL.

The oscillator clock is selected as source for the watchdog by configuring CCUCON1.INSEL = 01_B.

7.1.1.2 Back-up Clock

A back-up clock source is available as alternate clock source. This clock source provides a stable but reliable clock source that can be used as clock for the system. It provides less accuracy than an external crystal or ceramic resonator. The back-up clock can not be enabled or disabled or anyhow else be controlled. Therefore no control bits beside the selection itself as source (CCUCON0.CLKSEL = 00_B as clock source for the clock distribution and CCUCON1.INSEL = 00_B as clock source for the PLL / PLL_ERAY) are available.

7.1.2 Clock Speed Upscaling

Typical CPU operating speeds are about 10 times higher than the speed of the used crystal as clock source. Therefore an upscaling of the clock frequency is required.

For the upscaling up to two Phase Lock Loop (PLLs) are provided.

7.1.2.1 Phase-Locked Loop (PLL) Module

The PLL can convert a low-frequency external clock signal to a high-speed internal clock for maximum performance. It allows the use of input and output frequencies of a wide range by varying the different divider factors.

The PLL also has fail-safe logic that detects degenerate external clock behavior such as abnormal frequency deviations or a total loss of the external clock. It can execute emergency actions if it loses its lock on the external clock.

Features

- VCO lock detection
- 4-bit input divider **P**: (divide by $PDIV+1$)
- 7-bit feedback divider **N**: (multiply by $NDIV+1$)
- 7-bit output divider **K1 or K2**: (divide by either by $K1DIV+1$ or $K2DIV+1$)
- 7-bit output divider **K3**: (divide by $K3DIV+1$)
- Oscillator Watchdog
 - Detecting too low input frequencies
 - Detecting too high input frequencies
- Different operating modes
 - Prescaler Mode
 - Freerunning Mode
 - Normal Mode
- VCO Power Down
- Glitchless switching between both K-Dividers
- Glitchless switching between Normal Mode and Prescaler Mode
- Frequency Modulation
- Jitter reduction for modulation jitter

PLL Functional Description

The following figure shows the PLL block structure.

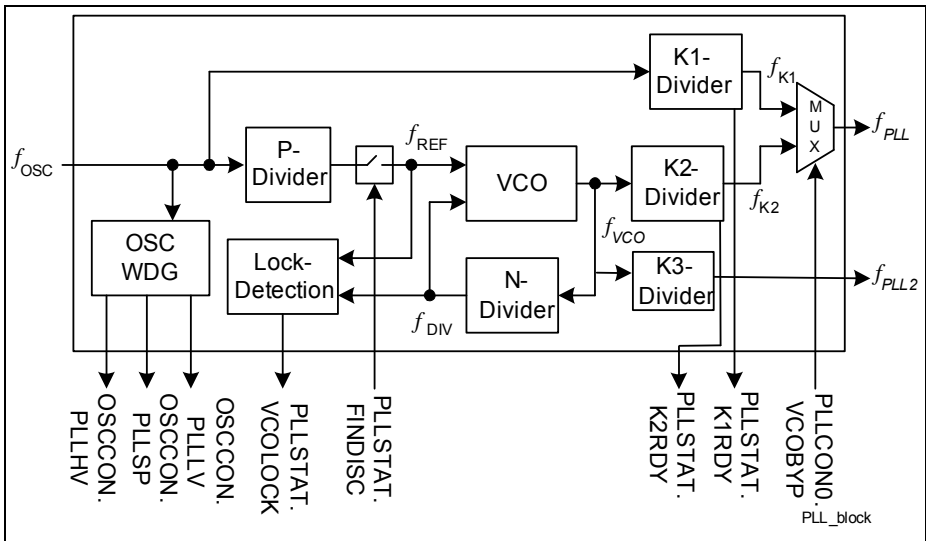


Figure 7-5 PLL Block Diagram

Mode Control

The PLL clock f_{PLL} and f_{PLL2} are generated from f_{OSC} in one of following software selectable modes:

- Normal Mode
- Prescaler Mode
- Freerunning Mode

Normal Mode

In Normal Mode the input frequency f_{OSC} is divided down by a factor P, multiplied by a factor N and then divided down by a factor K2 or K3.

The output frequency is given by

$$f_{PLL} = \frac{N}{P \cdot K2} \cdot f_{OSC} \quad (7.2)$$

$$f_{PLL2} = \frac{N}{P \cdot K3} \cdot f_{OSC} \quad (7.3)$$

Prescaler Mode

In Prescaler Mode the reference frequency f_{OSC} is only divided down by a factor K1. The output frequency is given by

$$f_{PLL} = \frac{f_{OSC}}{K1} \quad (7.4)$$

Freerunning Mode

In Freerunning Mode the base frequency output of the Voltage Controlled Oscillator (VCO) $f_{PLLBASE}$ is only divided down by a factor K2 or K3.

The output frequency is given by

$$f_{PLL} = \frac{f_{PLLBASE}}{K2} \quad (7.5)$$

$$f_{PLL2} = \frac{f_{PLLBASE}}{K3} \quad (7.6)$$

Configuration and Operation of the Freerunning Mode

In Freerunning Mode, the PLL is running at its VCO base frequency and f_{PLL} / f_{PLL2} is derived from f_{VCO} only by the K2 / K3-Divider.

The Freerunning Mode is entered after each System Reset.

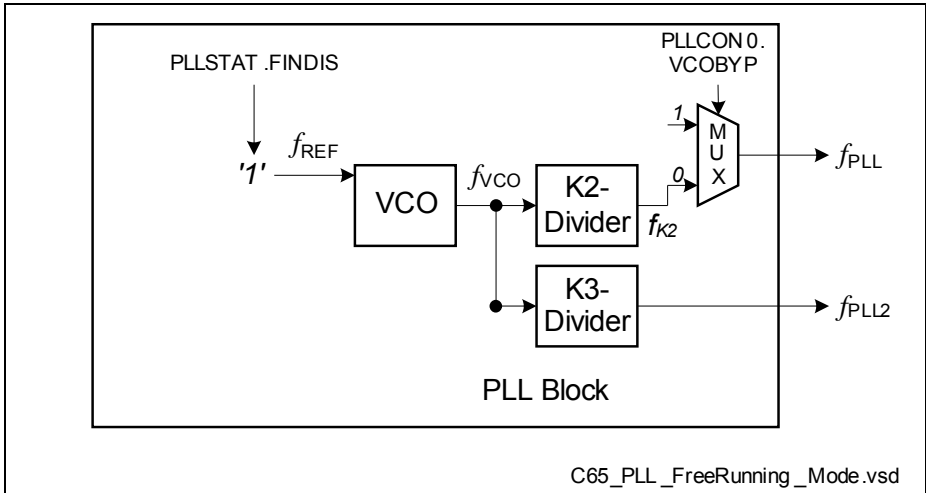


Figure 7-6 PLL Free-Running Mode Diagram

The output frequency is given by

$$f_{PLL} = \frac{f_{PLLBASE}}{K2} \quad (7.7)$$

$$f_{PLL2} = \frac{f_{PLLBASE}}{K3} \quad (7.8)$$

The Freerunning Mode is selected by the following settings

- PLLCON0.VCOBYP = 0
- PLLCON0.SETFINDIS = 1

The Freerunning Mode is entered when

- PLLSTAT.FINDIS = 1
- AND
- PLLSTAT.VCOBYST = 0

Operation on the Freerunning Mode does not require an input clock frequency of f_{OSC} . The Freerunning Mode is automatically entered on a PLL VCO Loss-of-Lock event if bit PLLCON0.OSCDISDIS is cleared. This mechanism allows a fail-safe operation of the PLL as in emergency cases still a clock is available.

The frequency of the Freerunning Mode $f_{PLLBASE}$ is listed in the Data Sheet.

Note: Changing the system operation frequency by changing the value of the K2-Divider has a direct effect on the power consumption of the device. Therefore this has to be changed carefully to avoid excessive current steps.

Depending on the selected divider value of the K2-Divider the duty cycle of the clock is selected. This can have an impact for the operation with an external communication interface.

Configuration and Operation of the Prescaler Mode

In Prescaler Mode, the PLL is running at the external frequency f_{OSC} and f_{PLL} is derived from f_{OSC} only by the K1-Divider.

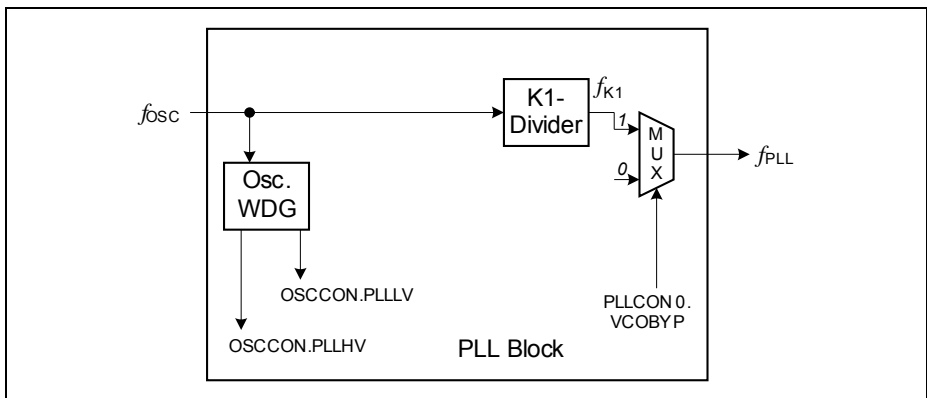


Figure 7-7 PLL Prescaler Mode Diagram

The output frequency is given by:

$$f_{PLL} = \frac{f_{OSC}}{K1} \quad (7.9)$$

The Prescaler Mode is selected by the following settings

- PLLCON0.VCOBYP = 1

The Prescaler Mode is entered when the following requirement is invalid:

- PLLSTAT.VCOBYST = 1

Operation on the Prescaler Mode does require an input clock frequency of f_{OSC} . Therefore it is recommended to check and monitor if an input frequency f_{OSC} is available at all by checking OSCCON.PLLLV. For a better monitoring also the upper frequency can be monitored via OSCCON.PLLHV.

For the Prescaler Mode there are no requirements regarding the frequency of f_{OSC} .

The system operation frequency is controlled in the Prescaler Mode by the value of the K1-Divider. When the value of PLLCON1.K1DIV was changed the next update of this value should not be done before bit PLLSTAT.K1RDY is set.

Note: Changing the system operation frequency by changing the value of the K1-Divider has a direct coupling to the power consumption of the device. Therefore this has to be done carefully.

Depending on the selected divider value of the K1-Divider the duty cycle of the clock is selected. This can have an impact for the operation with an external communication interface. The duty cycles values for the different K1-divider values are defined in the Data Sheet.

The Prescaler Mode is requested from the Freerunning or Normal Mode by setting bit PLLCON.VCOBYP. The Prescaler Mode is entered when the status bit PLLSTAT.VCOBYST is set. Before the Prescaler Mode is requested the K1-Divider should be configured with a value generating a PLL output frequency f_{PLL} that matches the one generated by the Freerunning or Normal Mode as much as possible. In this way the frequency change resulting out of the mode change is reduced to a minimum.

The Prescaler Mode is requested to be left by clearing bit PLLCON.VCOBYP. The Prescaler Mode is left when the status bit PLLSTAT.VCOBYST is cleared.

Configuration and Operation of the Normal Mode

In Normal Mode, the PLL is running at the external frequency f_{OSC} divided down by a factor P, multiplied by a factor N and then divided down by a factor K2 / K3 resulting in $f_{\text{PLL}} / f_{\text{PLL}2}$.

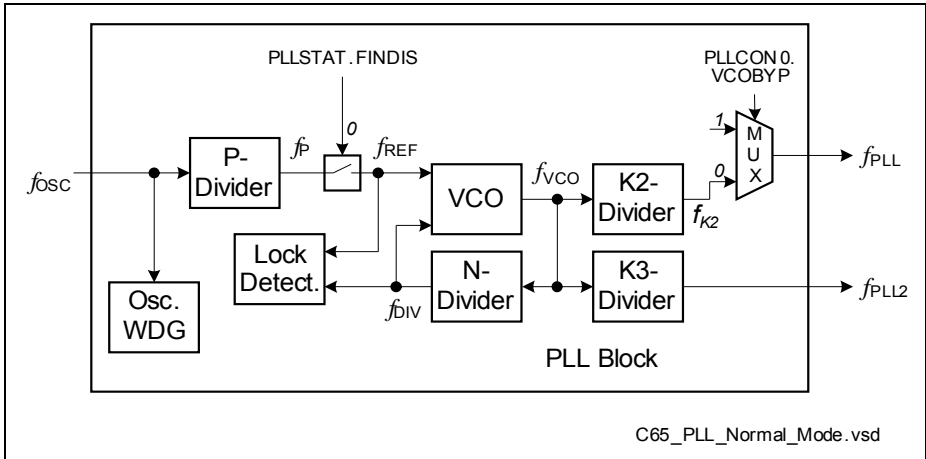


Figure 7-8 PLL Normal Mode Diagram

The output frequency is given by:

$$f_{\text{PLL}} = \frac{N}{P \cdot K_2} \cdot f_{\text{OSC}} \quad (7.10)$$

$$f_{\text{PLL2}} = \frac{N}{P \cdot K_3} \cdot f_{\text{OSC}} \quad (7.11)$$

The Normal Mode is selected by the following settings

- PLLCON0.VCOBYP = 0
- PLLCON0.CLRFINDIS = 1

The Normal Mode is entered when the following requirements are all together valid:

- PLLSTAT.FINDIS = 0
- PLLSTAT.VCOBYST = 0
- PLLSTAT.VCOLOCK = 1
- OSCCON.PLLLV = 1
- OSCCON.PLLHV = 1

Operation on the Normal Mode does require an input clock frequency of f_{OSC} . Therefore it is recommended to check and monitor if an input frequency f_{OSC} is available at all by checking OSCCON.PLLLV. For a better monitoring also the upper frequency can be monitored via OSCCON.PLLHV.

The system operation frequency is controlled in the Normal Mode by the values of the three dividers: P, N, and K2 / K3. A modification of the two dividers P and N has a direct influence to the VCO frequency and could lead to a loss of the VCO Lock status. A modification of the K2 / K3-divider has no impact on the VCO Lock status but still changes the PLL output frequency.

Note: Changing the system operation frequency by changing the value of the K2-Divider has a direct coupling to the power consumption of the device. Therefore this has to be done carefully.

When the frequency of the Normal Mode should be modified or entered the following sequence should be followed:

First the Prescaler Mode should be configured and entered. For more details see the Prescaler Mode.

The SMU alarm generation for the VCO Loss of Lock should be disabled.

While the Prescaler Mode is used the Normal Mode can be configured and checked for a positive VCO Lock status. The first target frequency of the Normal Mode should be selected in a way that it matches or is only slightly higher as the one used in the Prescaler Mode. This avoids big changes in the system operation frequency and therefore power consumption when switching later from Prescaler Mode to Normal Mode. The P and N divider should be selected in the following way:

- Selecting P and N in a way that f_{VCO} is in the lower area of its allowed values leads to a slightly reduced power consumption but to a slightly increased jitter
- Selecting P and N in a way that f_{VCO} is in the upper area of its allowed values leads to a slightly increased power consumption but to a slightly reduced jitter

After the P, N, and K2 / K3 dividers have been updated on the first configuration, the indication of the VCO Lock status should be checked (PLLSTAT.VCOLOCK = 1).

Note: When configuring the PLL for the first time after a system reset it is recommended to disconnect the input clock f_{OSC} before configuring P and / or N and to connect f_{OSC} before checking for the lock status.

Note: It is recommended to reset the VCO Lock detection (PLLCON0.RESLD = 1) after the new values of the dividers are configured to get a defined VCO lock check time.

Note: Resetting the lock detection (PLLCON0.RESLD = 1) while PLLCON0.OSCDISCDIS = 0 when the PLL was already locked (PLLSTAT.VCOLOCK = 1) can lead to a disconnect of the input clock. In this case PLLCON0.OSCDISCDIS should be set before lock detection is reset. When the lock is thereafter is set again PLLCON0.OSCDISCDIS could be cleared again.

When this happens the switch from Prescaler Mode to Normal Mode can be done. Normal Mode is requested by clearing PLLCON.VCOBYP. The Normal Mode is entered when the status bit PLLSTAT.VCOBYST is cleared.

Now the Normal Mode is entered. The SMU status flag for the system PLL VCO Loss-of-Lock event should be cleared and then enabled again. The intended PLL output target frequency can now be configured by changing only the K2 / K3-Divider.

Depending on the selected divider value of the K2-Divider the duty cycle of the clock is selected. This can have an impact for the operation with an external communication interface.

This can result in multiple changes of the K2-Divider to avoid too big frequency changes. Between the update of two K2-Divider values 6 cycles of f_{PLL} should be waited.

PLL VCO Lock Detection

The PLL has a lock detection that supervises the VCO part of the PLL in order to differentiate between stable and instable VCO circuit behavior. The lock detector marks the VCO circuit and therefore the output f_{VCO} of the VCO as instable if the two inputs f_{REF} and f_{DIV} differ too much. Changes in one or both input frequencies below a level are not marked by a loss of lock because the VCO can handle such small changes without any problem for the system.

PLL VCO Loss-of-Lock Event

The PLL may become unlocked, caused by a break of the crystal / ceramic resonator or the external clock line. In such a case, an SMU alarm event is generated.

Additionally, the OSC clock input f_{OSC} is disconnected from the PLL VCO to avoid unstable operation due to noise or sporadic clock pulses coming from the oscillator circuit. Without a clock input f_{OSC} , the PLL gradually slows down to its VCO base frequency and remains there. This automatic feature can be disabled by setting bit PLLCON0.OSCDISCDIS. If this bit is set the OSC clock remains connected to the VCO.

VCO Power Down Mode

The PLL offers a VCO Power Down Mode. This mode can be entered to save power within the PLL. The VCO Power Down Mode is entered by setting bit PLLCON0.VCOPWD. While the PLL is in VCO Power Down Mode only the Prescaler Mode is operable. Please note that selecting the VCO Power Down Mode does not automatically switch to the Prescaler Mode. So before the VCO Power Down Mode is entered the Prescaler Mode must be active.

PLL Power Down Mode

The PLL offers a Power Down Mode. This mode can be entered to save power if the PLL is not needed at all. The Power Down Mode is entered by setting bit PLLCON0.PLLPWD. While the PLL is in Power Down Mode no PLL output frequency is generated.

Frequency Modulation

If the PLL operates in Normal Mode the output frequency f_{PLL} can additionally be modified by a low-frequency modulation. f_{VCO} is randomly modulated.

The modulation is enabled via bit PLLCON0.MODEN. The modulation itself variate the VCO frequency randomly with the range of the configured modulation amplitude. The modulation amplitude is selected via PLLCON2.MODCFG[9:0].

(7.12)

$$\text{MA} = \frac{f_{\text{MV}}}{f_{\text{VCO}} \times 2} \times \text{MODCFG}[9, 0]$$

The modulation performed in a way that the resulting accumulated jitter added by the modulation stays below J_{MOD} . The modulation itself is monitored with f_{REF} and therefore the P-divider should be configured with the smallest possible value.

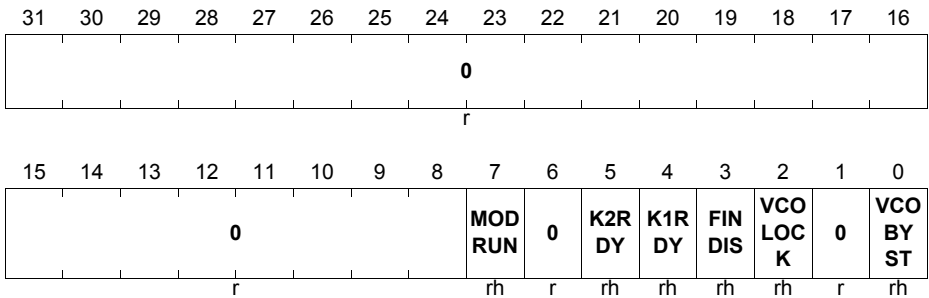
PLL Registers

PLL registers can be accessed by all CPUs in the system. However, it is suggested that only one CPU is used to control the clocks. As CPU0 is the active and available CPU immediately after a reset, this is the logical choice.

These registers control the setting of the PLL.

PLLSTAT

PLL Status Register (014_H) **Reset Value: 0000 0038_H**



Field	Bits	Type	Description
VCOBYST	0	rh	VCO Bypass Status 0 _B Freerunning / Normal Mode is entered 1 _B Prescaler Mode is entered
VCLOCK	2	rh	PLL VCO Lock Status 0 _B The frequency difference of f_{REF} and f_{DIV} is greater than allowed. The VCO part of the PLL can not lock on a target frequency. 1 _B The frequency difference of f_{REF} and f_{DIV} is small enough to enable a stable VCO operation. <i>Note: In case of a loss of VCO lock the f_{VCO} is keep on the previous constant frequency.</i>

Field	Bits	Type	Description
FINDIS	3	rh	<p>Input Clock Disconnect Select Status</p> <p>0_B The input clock from the oscillator is connected to the VCO part</p> <p>1_B The input clock from the oscillator is disconnected from the VCO part</p> <p><i>Note: This bit can be set by setting bit PLLCON0.SETFINDIS.</i></p> <p><i>Note: This bit can be cleared by setting bit PLLCON0.CLRFINDIS.</i></p>
K1RDY	4	rh	<p>K1 Divider Ready Status</p> <p>This bit indicates if the K1-divider operates on the configured value or not. this is of interest if the values is changed.</p> <p>0_B K1-Divider is not ready to operate with the new value</p> <p>1_B K1-Divider is ready to operate with the new value</p>
K2RDY	5	rh	<p>K2 Divider Ready Status</p> <p>This bit indicates if the K2-divider operates on the configured value or not. this is of interest if the values is changed.</p> <p>0_B K2-Divider is not ready to operate with the new value</p> <p>1_B K2-Divider is ready to operate with the new value</p>
MODRUN	7	rh	<p>Modulation Run</p> <p>This bit indicates if the frequency modulation of the PLL is activated or not.</p> <p>0_B Frequency modulation is not active</p> <p>1_B Frequency modulation is active</p>
0	1, 6, [31:8]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

PLLCON0
PLL Configuration 0 Register
(018_H)
Reset Value: 0001 C600_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			PDIV				0				RES LD	0	PLL PWD		
r			rw				r				w	r	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDIV						0	0	OSC DISC DIS	CLR FIN DIS	SET FIN DIS	0	MOD EN	VCO PWD	VCO BYP	
rw						rw	r	rw	w	w	rw	rw	rw	rw	

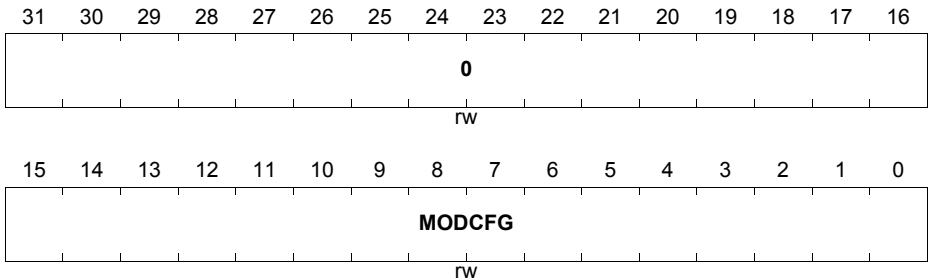
Field	Bits	Type	Description
VCOBYP	0	rw	VCO Bypass 0 _B Normal operation, VCO is not bypassed 1 _B Prescaler Mode; VCO is bypassed
VCOPWD	1	rw	VCO Power Saving Mode 0 _B Normal behavior 1 _B The VCO is put into a Power Saving Mode and can no longer be used. Only Prescaler Mode are active if previously selected.
MODEN	2	rw	Modulation Enable This bit controls the activation of the frequency modulation of the PLL. 0 _B Frequency modulation is not activated 1 _B Frequency modulation is activated
SETFINDIS	4	w	Set Status Bit PLLSTAT.FINDIS 0 _B Bit PLLSTAT.FINDIS is left unchanged 1 _B Bit PLLSTAT.FINDIS is set. The input clock from the oscillator is disconnected from the VCO part.
CLRFINDIS	5	w	Clear Status Bit PLLSTAT.FINDIS 0 _B Bit PLLSTAT.FINDIS is left unchanged 1 _B Bit PLLSTAT.FINDIS is cleared. The input clock from the oscillator is connected to the VCO part.

Field	Bits	Type	Description
OSCDISCDIS	6	rw	Oscillator Disconnect Disable This bit is used to disable the control PLLSTAT.FINDIS in a PLL loss-of-lock case. 0 _B In case of a PLL loss-of-lock bit PLLSTAT.FINDIS is set 1 _B In case of a PLL loss-of-lock bit PLLSTAT.FINDIS is cleared
NDIV	[15:9]	rw	N-Divider Value The value the N-Divider operates is NDIV+1.
PLLPWD	16	rw	PLL Power Saving Mode 0 _B The complete PLL block is put into a Power Saving Mode and can no longer be used. 1 _B Normal behavior
RESLD	18	w	Restart VCO Lock Detection Setting this bit will clear bit PLLSTAT.VCOLOCK and restart the VCO lock detection. Reading this bit returns always a zero.
PDIV	[27:24]	rw	P-Divider Value The value the P-Divider operates is PDIV+1.
0	3, 8	rw	Reserved Have to be written with 0.
0	7, 17, [23:19], [31:28]	r	Reserved Read as 0; should be written with 0.

PLLCON1
PLL Configuration 1 Register
(01C_H)
Reset Value: 0002 020F_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								K1DIV							
r								rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	K3DIV						0	K2DIV							
r	rw						r	rw							

Field	Bits	Type	Description
K2DIV	[6:0]	rw	K2-Divider Value The value the K2-Divider operates is K2DIV+1.
K3DIV	[14:8]	rw	K3-Divider Value The value the K3-Divider operates is K3DIV+1.
K1DIV	[22:16]	rw	K1-Divider Value The value the K1-Divider operates is K1DIV+1.
0	7, 15, [31:23]	r	Reserved Read as 0; should be written with 0.

PLLCON2
PLL Configuration 2 Register
(020_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
MODCFG	[15:0]	rw	Modulation Configuration This bit field defines the modulation. MODCFG[9:0] defines the modulation amplitude. Bits MODCFG[9:5] are treated as integer part and bits MODCFG[4:0] as fractional part. Bits MODCFG[15:10] have to be configured with the following setting: 0x111101 _B .
0	[31:16]	rw	Reserved Have to be written with 0.

7.1.2.2 ERAY Phase-Locked Loop (PLL_ERAY) Module

The PLL_ERAY can convert a low-frequency external clock signal to a high-speed internal clock for maximum performance. It can execute emergency actions if it loses its lock on the external clock.

This module is a phase locked loop for integer frequency synthesis. It allows the use of input and output frequencies of a wide range by varying the different divider factors.

Features

- VCO lock detection
- 4-bit input divider **P**: (divide by PDIV+1)
- 5-bit feedback divider **N**: (multiply by NDIV+1)
- 7-bit output divider **K1 or K2**: (divide by either by K1DIV+1 or K2DIV+1)
- 7-bit output divider **K3**: (divide by K3DIV+1)
- Different operating modes
 - Prescaler Mode
 - Freerunning Mode
 - Normal Mode
- VCO Power Down (Sleep Mode)
- Glitchless switching between both K-Dividers
- Glitchless switching between Normal Mode and Prescaler Mode

PLL_ERAY Functional Description

The following figure shows the PLL_ERAY block structure.

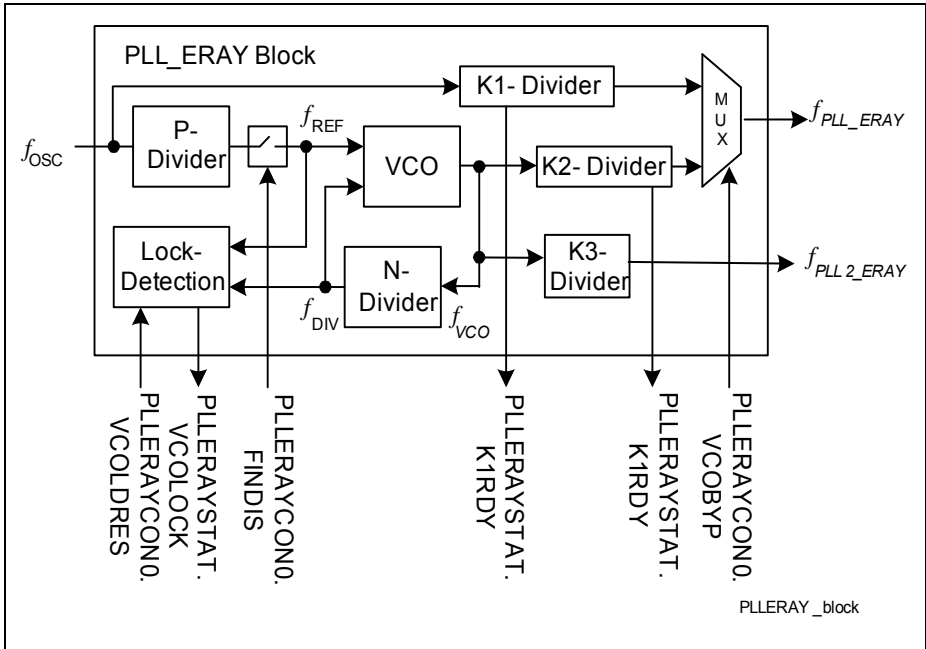


Figure 7-9 PLL_ERAY Block Diagram

Clock Source Control

The PLL_ERAY clocks f_{PLL_ERAY} / f_{PLL2_ERAY} are generated from f_{OSC} in one of three software selectable modes:

- Normal Mode
- Prescaler Mode
- Freerunning Mode

Normal Mode

In Normal Mode the input frequency f_{OSC} is divided down by a factor P, multiplied by a factor N and then divided down by a factor K2 / K3.

The output frequency is given by

(7.13)

$$f_{\text{PLL_ERAY}} = \frac{N}{P \times K2} \cdot f_{\text{OSC}}$$

(7.14)

$$f_{\text{PLL2_ERAY}} = \frac{N}{P \times K3} \cdot f_{\text{OSC}}$$

Prescaler Mode

In Prescaler Mode the reference frequency f_{OSC} is only divided down by a factor K1.

The output frequency is given by

(7.15)

$$f_{\text{PLL_ERAY}} = \frac{f_{\text{OSC}}}{K1}$$

Freerunning Mode

In Freerunning Mode the base frequency output of the Voltage Controlled Oscillator (VCO) $f_{\text{PLLBASEERAY}}$ is only divided down by a factor K2 / K3.

The output frequency is given by

(7.16)

$$f_{\text{PLL_ERAY}} = \frac{f_{\text{PLLBASEERAY}}}{K2}$$

(7.17)

$$f_{\text{PLL2_ERAY}} = \frac{f_{\text{PLLBASEERAY}}}{K3}$$

Configuration and Operation of the Freerunning Mode

In Freerunning Mode, the PLL_ERAY is running at its VCO base frequency and $f_{\text{PLL_ERAY}} / f_{\text{PLL2_ERAY}}$ are derived from f_{VCO} only by the K2 / K3-Divider.

The Freerunning Mode is entered after each System Reset.

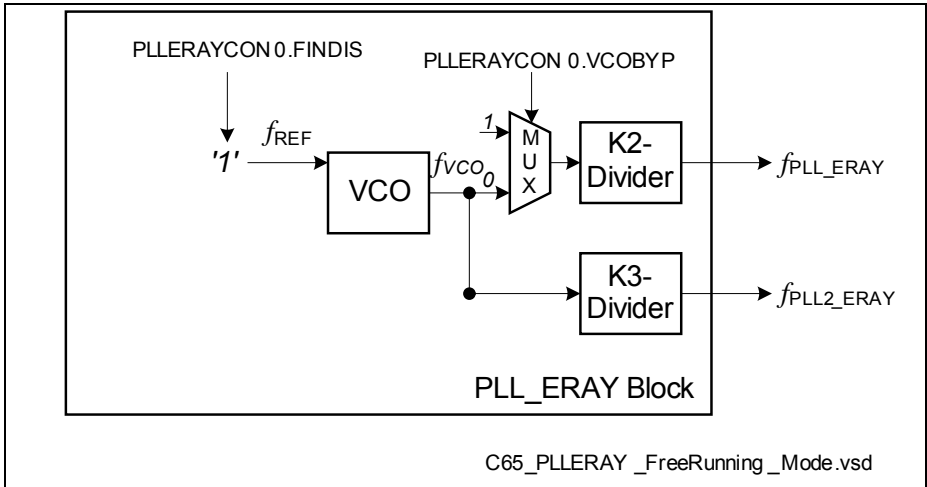


Figure 7-10 PLL_ERAY Free-Running Mode Diagram

The output frequency is given by

$$f_{\text{PLLERAY}} = \frac{f_{\text{PLLBASEERAY}}}{K2} \quad (7.18)$$

$$f_{\text{PLL2ERAY}} = \frac{f_{\text{PLLBASEERAY}}}{K3} \quad (7.19)$$

The Freerunning Mode is selected by the following settings

- PLLERAYCON0.VCOBYP = 0
- PLLERAYCON0.SETFINDIS = 1

The Freerunning Mode is entered when

- PLLERAYSTAT.FINDIS = 1
- AND
- PLLERAYSTAT.VCOBYST = 0

Operation on the Freerunning Mode does not require an input clock frequency of f_{OSC} . The Freerunning Mode is automatically entered on a PLL_ERAY VCO Loss-of-Lock event if bit PLLERAYCON0.OSCDISDIS is cleared. This mechanism allows a fail-safe operation of the PLL_ERAY as in emergency cases still a clock is available.

The frequency of the Freerunning Mode $f_{\text{PLLBASEERAY}}$ is listed in the Data Sheet.

Depending on the selected divider value of the K2-Divider the duty cycle of the clock is selected. This can have an impact for the operation with an external communication interface.

Configuration and Operation of the Prescaler Mode

In Prescaler Mode, the PLL_ERAY is running at the external frequency f_{OSC} and f_{PLL_ERAY} is derived from f_{OSC} only by the K1-Divider.

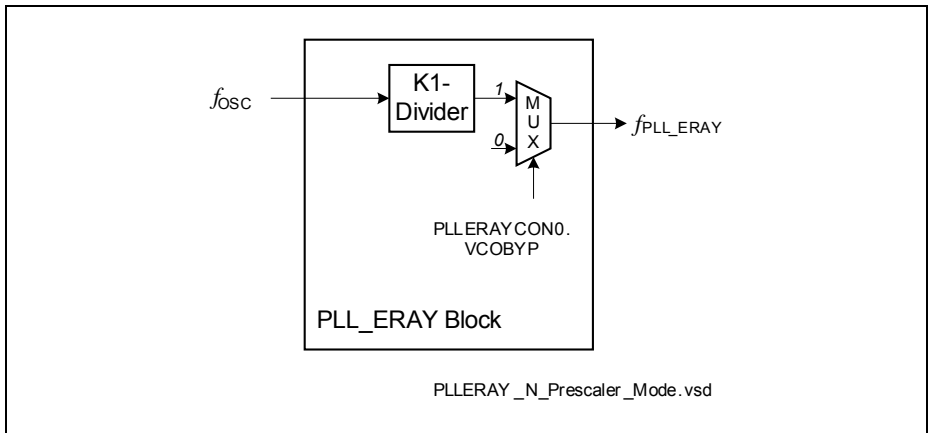


Figure 7-11 PLL_ERAY Prescaler Mode Diagram

The output frequency is given by:

$$f_{PLL_ERAY} = \frac{f_{OSC}}{K1} \tag{7.20}$$

The Prescaler Mode is selected by the following settings

- PLLERYCON0.VCOBYP = 1

The Prescaler Mode is entered when the following requirement is invalid:

- PLLERYSTAT.VCOBYST = 1

Operation on the Prescaler Mode does require an input clock frequency of f_{OSC} . Therefore it is recommended to check and monitor if an input frequency f_{OSC} is available at all by checking OSCCON.PLLLV. For a better monitoring also the upper frequency can be monitored via OSCCON.PLLHV.

For the Prescaler Mode there are no requirements regarding the frequency of f_{OSC} .

The system operation frequency is controlled in the Prescaler Mode by the value of the K1-Divider. When the value of PLLERAYCON1.K1DIV was changed the next update of this value should not be done before bit PLLERAYSTAT.K1RDY is set.

Depending on the selected divider value of the K1-Divider the duty cycle of the clock is selected. This can have an impact for the operation with an external communication interface. The duty cycles values for the different K1-divider values are defined in the Data Sheet.

The Prescaler Mode is requested from the Freerunning or Normal Mode by setting bit PLLERAYCON.VCOBYP. The Prescaler Mode is entered when the status bit PLLERAYSTAT.VCOBYST is set. Before the Prescaler Mode is requested the K1-Divider should be configured with a value generating a PLL_ERAY output frequency f_{PLL_ERAY} that matches the one generated by the Freerunning or Normal Mode as much as possible. In this way the frequency change resulting out of the mode change is reduced to a minimum.

The Prescaler Mode is requested to be left by clearing bit PLLERAYCON.VCOBYP. The Prescaler Mode is left when the status bit PLLERAYSTAT.VCOBYST is cleared.

Configuration and Operation of the Normal Mode

In Normal Mode, the PLL is running at the external frequency f_{OSC} divided down by a factor P, multiplied by a factor N and then divided down by a factor K2 / K3 resulting in $f_{PLL_ERAY} / f_{PLL2_ERAY}$.

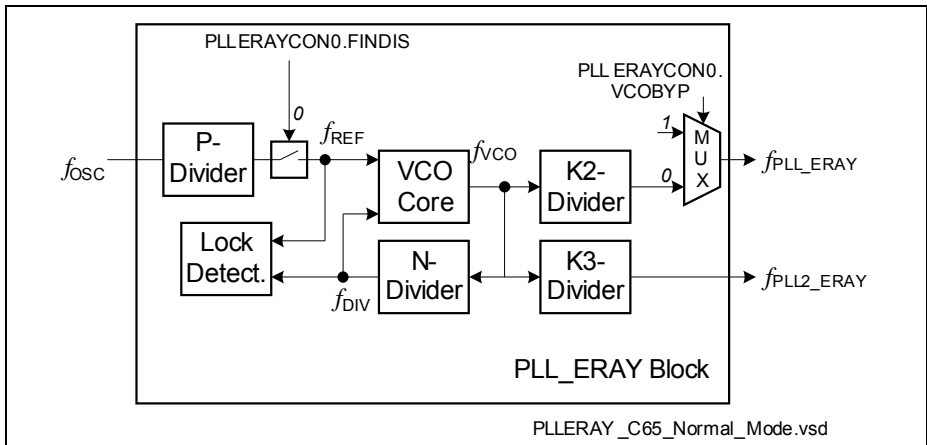


Figure 7-12 PLL_ERAY Normal Mode Diagram

The output frequency is given by:

(7.21)

$$f_{\text{PLL1ERAY}} = \frac{N}{P \times K2} \cdot f_{\text{OSC}}$$

(7.22)

$$f_{\text{PLL2ERAY}} = \frac{N}{P \times K3} \cdot f_{\text{OSC}}$$

The Normal Mode is selected by the following settings

- PLLERAYCON0.VCOBYP = 0
- PLLERAYCON0.CLRFINDIS = 1

The Normal Mode is entered when the following requirements are all together valid:

- PLLERAYSTAT.FINDIS = 0
- PLLERAYSTAT.VCOBYST = 0
- PLLERAYSTAT.VCOLOCK = 1
- OSCCON.PLLLV = 1
- OSCCON.PLLHV = 1

Operation on the Normal Mode does require an input clock frequency of f_{OSC} . Therefore it is recommended to check and monitor if an input frequency f_{OSC} is available at all by checking OSCCON.PLLLV. For a better monitoring also the upper frequency can be monitored via OSCCON.PLLHV.

The system operation frequency is controlled in the Normal Mode by the values of the three dividers: P, N and K2 / K3. A modification of the two dividers P and N has a direct influence to the VCO frequency and could lead to a loss of the VCO Lock status. A modification of the K2 / K3-divider has no impact on the VCO Lock status but still changes the PLL_ERAY output frequency.

When the frequency of the Normal Mode should be modified or entered the following sequence should be followed:

First the Prescaler Mode should be configured and entered. For more details see the Prescaler Mode.

The SMU alarm generation for the VCO Loss of Lock should be disabled.

While the Prescaler Mode is used the Normal Mode can be configured and checked for a positive VCO Lock status. The first target frequency of the Normal Mode should be selected in a way that it matches or is only slightly higher as the one used in the Prescaler Mode. This avoids big changes in the system operation frequency and therefore power consumption when switching later from Prescaler Mode to Normal Mode. The N divider should be selected in the following way:

- Selecting P and N in a way that f_{VCO} is in the lower area of its allowed values leads to a slightly reduced power consumption but to a slightly increased jitter
- Selecting P and N in a way that f_{VCO} is in the upper area of its allowed values leads to a slightly increased power consumption but to a slightly reduced jitter

After the P, N and K2 / K3 dividers are updated for the first configuration the indication of the VCO Lock status should be await (PLLERYSTAT.VCOLOCK = 1).

Note: When configuring the PLL the input clock f_{OSC} should be disconnected before configuring P and / or N and connected before checking for the lock status.

Note: It is recommended to reset the VCO Lock detection (PLLERYCON0.RESLD = 1) after the new values of the dividers are configured to get a defined VCO lock check time.

Note: Resetting the lock detection (PLLERYCON0.RESLD = 1) while PLLERYCON0.OSCDISCDIS = 0 when the PLL_ERAY was already locked (PLLERYSTAT.VCOLOCK = 1) can lead to a disconnect of the input clock. In this case PLLERYCON0.OSCDISCDIS should be set before lock detection is reset. If the lock is subsequently set again PLLERYCON0.OSCDISCDIS could be cleared again.

When this happens the switch from Prescaler Mode to Normal Mode can be done. Normal Mode is requested by clearing PLLERYCON.VCOBYP. The Normal Mode is entered when the status bit PLLERYSTAT.VCOBYST is cleared.

Now the Normal Mode is entered. The SMU alarm flag for the PLL_ERAY VCO Loss-of-Lock event should be cleared and then enabled again. The intended PLL_ERAY output target frequency can not be configured by changing only the K2-Divider.

Depending on the selected divider value of the K2-Divider the duty cycle of the clock is selected. This can have an impact for the operation with an external communication interface. This can result in multiple changes of the K2-Divider to avoid too big frequency changes. Between the update of two K2-Divider values 6 cycles of f_{PLL_ERAY} should be waited.

PLL_ERAY VCO Lock Detection

The PLL_ERAY has a lock detection that supervises the VCO part of the PLL_ERAY in order to differentiate between stable and instable VCO circuit behavior. The lock detector marks the VCO circuit and therefore the output f_{VCO} of the VCO as instable if the two inputs f_{REF} and f_{DIV} differ too much. Changes in one or both input frequencies below a level are not marked by a loss of lock because the VCO can handle such small changes without any problem for the system.

PLL_ERAY VCO Loss-of-Lock Event

The PLL_ERAY may become unlocked, caused by a break of the crystal / ceramic resonator or the external clock line. In such a case, an SMU alarm event is generated.

Additionally, the OSC clock input f_{OSC} is disconnected from the PLL_ERAY VCO to avoid unstable operation due to noise or sporadic clock pulses coming from the oscillator circuit. Without a clock input f_{OSC} , the PLL_ERAY gradually slows down to its VCO base frequency and remains there. This automatic feature can be disabled by setting bit PLLERAYCON0.OSCDISCDIS. If this bit is cleared the OSC clock remains connected to the VCO.

VCO Power Down Mode

The PLL_ERAY offers a VCO Power Down Mode. This mode can be entered to save power within the PLL_ERAY. The VCO Power Down Mode is entered by setting bit PLLCON0.VCOPWD. While the PLL_ERAY is in VCO Power Down Mode only the Prescaler Mode is operable. Please note that selecting the VCO Power Down Mode does not automatically switch to the Prescaler Mode. So before the VCO Power Down Mode is entered, the Prescaler Mode must be active.

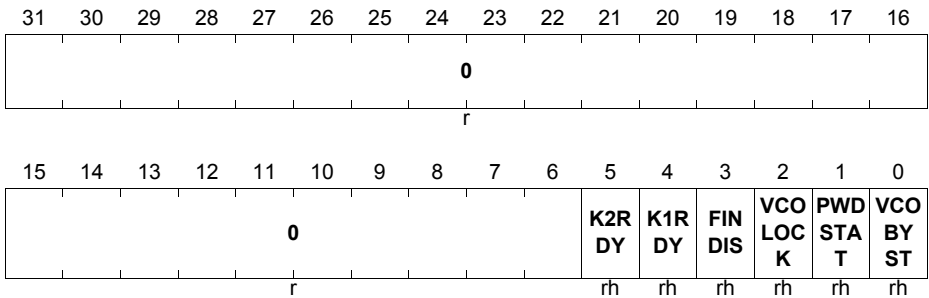
PLL_ERAY Registers

PLL_ERAY registers may be accessed by any CPUs in the system. However, it is suggested that only one CPU is used to control the clocks. As CPU0 is the active and available CPU after each reset this is the logical choice.

These registers controls the setting of the PLL_ERAY.

PLLERAYSTAT

PLL_ERAY Status Register (024_H) **Reset Value: 0000 0038_H**



Field	Bits	Type	Description
VCOBYST	0	rh	VCO Bypass Status 0 _B Freerunning / Normal Mode is entered 1 _B Prescaler Mode is entered
PWDSTAT	1	rh	PLL_ERAY Power-saving Mode Status 0 _B PLL_ERAY Power-saving Mode was not entered 1 _B PLL_ERAY Power-saving Mode was entered

Field	Bits	Type	Description
VCOLOCK	2	rh	<p>PLL VCO Lock Status</p> <p>0_B The frequency difference of f_{REF} and f_{DIV} is greater than allowed. The VCO part of the PLL_ERAY can not lock on a target frequency.</p> <p>1_B The frequency difference of f_{REF} and f_{DIV} is small enough to enable a stable VCO operation.</p> <p><i>Note: In case of a loss of VCO lock the f_{VCO} goes to the upper boundary of the VCO frequency if the reference clock input is greater than expected.</i></p> <p><i>Note: In case of a loss of VCO lock the f_{VCO} goes to the lower boundary of the VCO frequency if the reference clock input is lower than expected.</i></p>
FINDIS	3	rh	<p>Input Clock Disconnect Select Status</p> <p>0_B The input clock from the oscillator is connected to the VCO part</p> <p>1_B The input clock from the oscillator is disconnected from the VCO part</p> <p><i>Note: This bit can be set by setting bit PLLERAYCON0.SETFINDIS.</i></p> <p><i>Note: This bit can be cleared by setting bit PLLERAYCON0.CLRFINDIS.</i></p>
K1RDY	4	rh	<p>K1 Divider Ready Status</p> <p>This bit indicates if the K1-divider operates on the configured value or not. this is of interest if the values is changed.</p> <p>0_B K1-Divider is not ready to operate with the new value</p> <p>1_B K1-Divider is ready to operate with the new value</p>

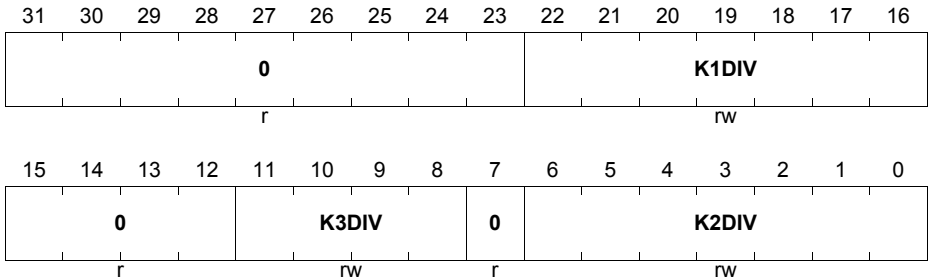
Field	Bits	Type	Description
K2RDY	5	rh	K2 Divider Ready Status This bit indicates if the K2-divider operates on the configured value or not. this is of interest if the values is changed. 0_B K2-Divider is not ready to operate with the new value 1_B K2-Divider is ready to operate with the new value
0	[31:6]	r	Reserved Read as 0; should be written with 0.

PLLERYCON0
PLL_ERAY Configuration 0 Register (028_H)
Reset Value: 0001 2E00_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			PDIV					0				RES LD	0	PLL PWD	
r			rw					r				w	r	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	NDIV				0		OSC DISC DIS	CLR FIN DIS	SET FIN DIS	0		VCO PWD	VCO BYP	
r	rw	rw				r		rw	w	w	r		rw	rw	

Field	Bits	Type	Description
VCOBYP	0	rw	VCO Bypass 0_B Normal operation, VCO is not bypassed 1_B Prescaler Mode; VCO is bypassed
VCOPWD	1	rw	VCO Power Saving Mode 0_B Normal behavior 1_B The VCO is put into a Power Saving Mode and can no longer be used.
SETFINDIS	4	w	Set Status Bit PLLERAYSTAT.FINDIS 0_B Bit PLLERAYSTAT.FINDIS is left unchanged 1_B Bit PLLERAYSTAT.FINDIS is set. The input clock from the oscillator is disconnected from the VCO part.

Field	Bits	Type	Description
CLRFINDIS	5	w	Clear Status Bit PLLERAYSTAT.FINDIS 0 _B Bit PLLERAYSTAT.FINDIS is left unchanged 1 _B Bit PLLERAYSTAT.FINDIS is cleared. The input clock from the oscillator is connected to the VCO part.
OSCDISCDIS	6	rw	Oscillator Disconnect Disable This bit is used to disable the control PLLERAYSTAT.FINDIS in a PLL_ERAY loss-of-lock case. 0 _B In case of a PLL loss-of-lock bit PLLERAYSTAT.FINDIS is set 1 _B In case of a PLL loss-of-lock bit PLLERAYSTAT.FINDIS is cleared
NDIV	[13:9]	rw	N-Divider Value The value the N-Divider operates is (NDIV+1).
PLLPWD	16	rw	PLL Power Saving Mode 0 _B The complete PLL_ERAY block is put into a Power Saving Mode and can no longer be used. Only the Bypass Mode is active if previously selected. 1 _B Normal behavior
RESLD	18	w	Restart VCO Lock Detection Setting this bit will clear bit PLLERAYSTAT.VCOLOCK and restart the VCO lock detection. Reading this bit returns always a zero.
PDIV	[27:24]	rw	P-Divider Value The value the P-Divider operates is PDIV+1.
0	14	rw	Reserved Should be written with 0.
0	[3:2], [8:7], 15, 17, [23:19], [31:28]]	r	Reserved Read as 0; should be written with 0.

PLLERAYCON1
PLL_ERAY Configuration 1 Register (02C_H)
Reset Value: 000F 020F_H


Field	Bits	Type	Description
K2DIV	[6:0]	rw	K2-Divider Value The value the K2-Divider operates is K2DIV+1.
K3DIV	[11:8]	rw	K3-Divider Value The value the K1-Divider operates is K3DIV+1.
K1DIV	[22:16]	rw	K1-Divider Value The value the K1-Divider operates is K1DIV+1.
0	7, [15:12], [31:23]	r	Reserved Read as 0; should be written with 0.

7.1.3 Clock Distribution

Using the first two parts of the Clock System all clocks the system rely on for operation are defined. Now these different clocks need to be distributed to the single modules, CPUs, and blocks in a way that these IPs can operate in best way in terms performance and power consumption.

For the clock distribution the system is split into several sub-clock domains where the clock speed could be configured individually but there are also several limitation for each sub-clock domain derived out of the internal interfaces.

Each sub-clock domain defines a logical unit from clocking perspective point of view.

The clock distribution is done via the Clock Control Unit (CCU). The CCU receives the clocks that are created by the two PLLs ($f_{PLL/PLL2}$ and f_{PLL_ERAY/PLL_ERAY2}), the back-up clock f_{Back} , and f_{OSC0} . These clock are either forwarded directly or divided in order to supply the sub-clock domains.

The following figure shows the structure of the TC26x B-Step clock system that shows the different sub-clock domains together with their clock inputs.

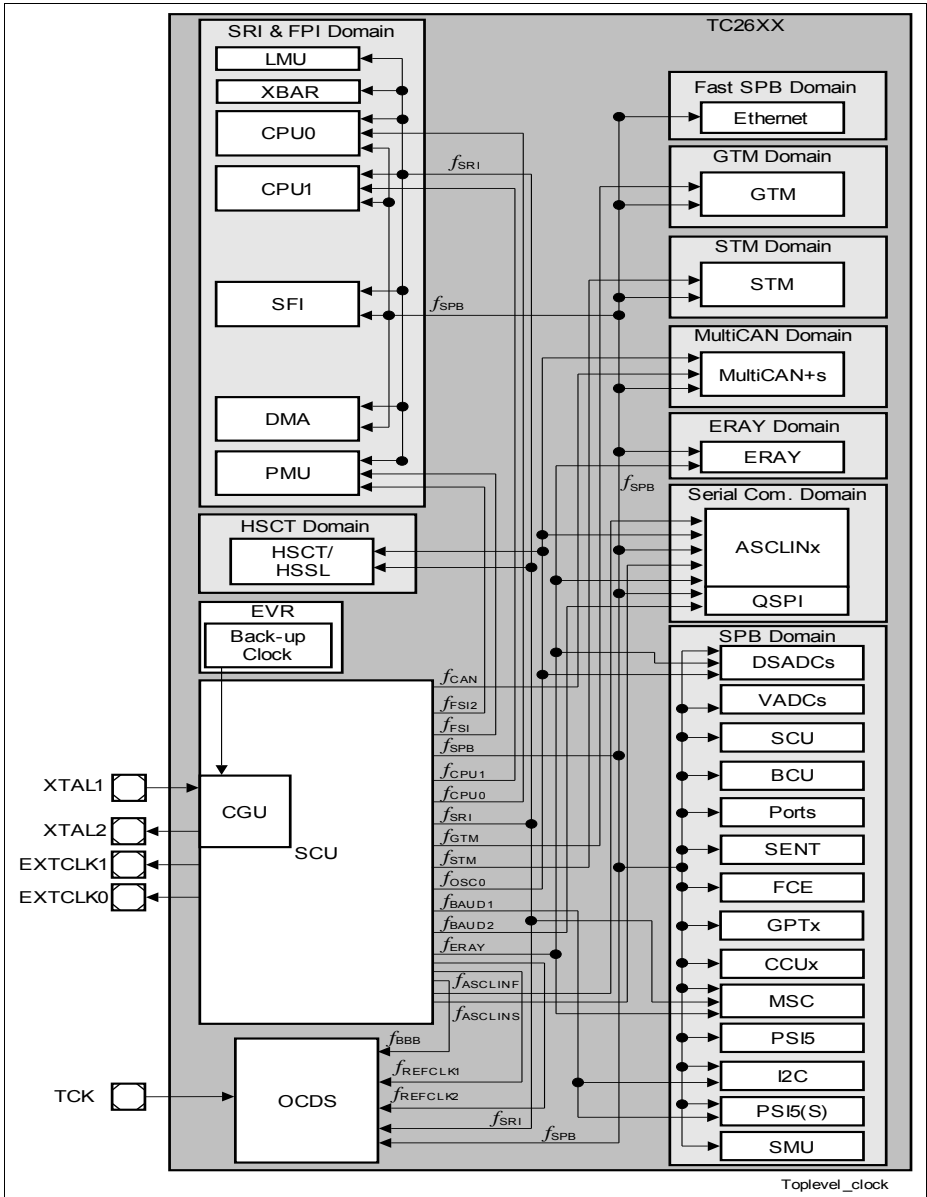


Figure 7-13 TC26x B-Step Clocking System

7.1.3.1 Clock Control Unit

The Clock Control Unit (CCU) receives the clocks that are created by the two PLLs ($f_{PLL/PLL2}$ and $f_{PLL_ERAY/PLL2_ERAY}$), the back-up clock f_{Back} , and f_{OSC0} .

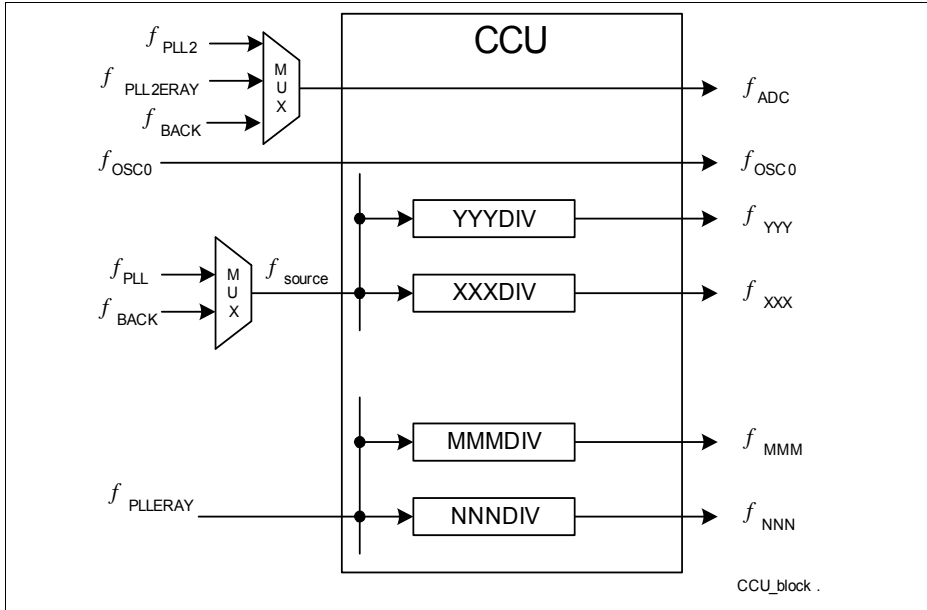


Figure 7-14 Clock Control Unit Overview

For most clocks a linear divider is provided to adapt the clock frequency to the application requirement. This divider is controlled by the bit field XXXDIV.

For the CPU clocks, more sophisticated dividers are implemented. This allows a better precision for the control in clock frequency changes of individual CPUs. This mechanism allows the adaptation of the frequency change to a specific current limit defined by the application.

There is also a fixed reference clock REFCLK1/2 for the debug block, which divides the master clock f_{PLL}/f_{PLL_ERAY} by 24. This allows the OCDS to generate timestamps independently of the selected SRI and SPB clock speeds.

Basic Clock System Mechanisms

The clocking system offers a lot of options and a high amount of flexibility. Behind this complex clocking system a set of basic ideas is implemented in order to allow a configuration as close as possible to the given application needs.

The heart of the system is a synchronous block including all modules except the ERAY. The heartbeat here is defined by f_{MAX} . f_{MAX} is defined as the highest frequency used inside the system for any clock derived out of f_{SOURCE} .

For different IPs (CPUs, modules or module clusters) options are included to influence the application execution / performance via the clock control.

Following is a brief overview of the different clocks:

- System buses
 - f_{SRI} defines the operating performance of the SRI-Bus and therefore the data exchange rate between all connected masters and slaves
 - f_{SPB} defines the operating performance of the SPB-Bus and therefore the data exchange rate between all connected masters and slaves and the interrupt system
- CPU controls
 - f_{CPU0} defines the execution speed of CPU0
 - f_{CPU1} defines the execution speed of CPU1
- PMU controls
 - f_{FSI2} defines the execution speed of the PFlash for reading operations
 - f_{FSI} defines the execution speed for all other flash operations
- Peripheral options
 - f_{STM} defines a basic frequency for the STMs independent of the rest of the system (beside the limitations listed in [Table 7-2](#)). This allows the STMs to operate on a constant frequency.
 - f_{GTM} defines a basic frequency for the GTM independent of the rest of the system (beside the limitations listed in [Table 7-2](#)). This allows the GTM to operate on a constant frequency.
 - f_{BAUD1} defines a basic frequency for ‘slow’ communication interfaces independent of the rest of the system (beside the limitations listed in [Table 7-2](#)). This allows the IIC to operate on a constant baudrate (frequency).
 - f_{BAUD2} defines a basic frequency for ‘fast’ communication interfaces independent of the rest of the system (beside the limitations listed in [Table 7-2](#)). This allows the QSPIs and PSI5S to operate on a constant baudrate (frequency).
 - f_{CAN} defines a basic frequency for the MultiCAN independent of the rest of the system (beside the limitations listed in [Table 7-2](#)). This allows the MultiCAN to operate on a constant baudrate (frequency).
 - $f_{ASCLINF} / ASCLINS$ defines a basic frequency for the ASCLINs independent of the rest of the system (beside the limitations listed in [Table 7-2](#)). This allows the ASCLINs to operate on a constant baudrate (frequency).
- Debug system
 - As debugging should be non-intrusive to the application system, a separate clock f_{BBB} for dedicated debug resources is available. This allows debugging (trace generation) during changes of the other clock configurations. Please note that f_{BBB} needs to be faster than or equal to f_{SPB} for debug.

In addition several peripherals offer the option to operate on other clock sources. If frequency modulation is used, some peripherals should switch to a non-modulated clock. Due to its unique requirements the ERAY module is implemented as an asynchronous domain with its own independent clock source.

Table 7-1 CCU Clock Options

CCU Clock Output	Clock Source			
	PLL	PLL_ERAY	Back-up	OSC_XTAL
f_{MAX}	✓	–	Default	–
f_{SRI}	✓	–	Default	–
f_{CPU0}	f_{SRI}	–	–	–
f_{CPU1}	f_{SRI}	–	–	–
f_{SPB}	✓	–	Default	–
f_{FSI}	f_{SRI}	–	–	–
f_{FSI2}	f_{SRI}	–	–	–
$f_{REFCLK1/2}$	✓	✓	Default	–
f_{BBB}	✓	–	Default	–
f_{ERAY}	–	Default	–	–
f_{GTM}	✓	–	Default	–
f_{STM}	✓	–	Default	–
f_{BAUD1}	✓	–	Default	–
f_{BAUD2}	✓	–	Default	–
f_{CAN}	✓	–	Default	–
$f_{ASCLINF}$	✓	–	Default	–
$f_{ASCLINS}$	✓	–	Default	–

The complete system is based on a single clock (f_{SOURCE}) in order to achieve the performance advantage of a synchronous system. Therefore certain limitations have to be considered when configuring the clock control options.

Clock Divider Limitations

For the clock dividers which control the different sub-clock domains / modules the allowed values are limited and the following ratios have to be observed. The ratios are defined in the following way: clock A = f_{AAA} ; clock B = f_{BBB} the allowed ratio is 1 : n where n has a defined range of values. Clock A is always faster or equal to clock B in this definition with f_{AAA} [MHz] = n * f_{BBB} [MHz].

In addition the divider values are limited by the resulting maximum frequencies. These allowed max. frequencies are defined in the Target Data Sheet / ACDC Target Specification.

Table 7-2 CCU allowed Clock Ratios

Clock A	Clock B	Allowed Ratios ¹⁾	Recommended Default
f_{SRI}	f_{SPB}	1 : n; n = 1, 2, 3, 4, 5, 6	n = 2
f_{SRI}	f_{FSI}	1 : n; n = 1, 2	n = 2
f_{SRI}	f_{FSI2}	1 : n; n = 1, 2	n = 1
f_{FSI2}	f_{FSI}	1 : n; n = 1, 2	n = 2
f_{GTM}	f_{SPB}	1 : n ²⁾ ; n = 1, 2, 3, 4, 5,...	n = 1
f_{SPB}	f_{GTM}	1 : n ²⁾ ; n = 1, 2, 3, 4, 5,...	n = 1
f_{SPB}	f_{STM}	1 : n ³⁾ ; n = 1, 2, 3, 4, 5,...	n = 1
f_{STM}	f_{SPB}	1 : n ³⁾ ; n = 1, 2, 3, 4, 5,...	n = 1
f_{SRI}	f_{BBB} ⁴⁾	1 : n; n = 1, 2	n = 2
f_{BAUD1}	f_{SPB}	1 : n ⁵⁾ ; n = 1, 2, 3, 4, 5,...	n = 1
f_{SPB}	f_{BAUD1}	1 : n ⁵⁾ ; n = 1, 2, 3, 4, 5,...	n = 1
f_{BAUD2}	f_{SPB}	1 : n ⁶⁾ ; n = 1, 2, 3, 4, 5,...	n = 2
f_{SPB}	f_{BAUD2}	1 : n ⁶⁾ ; n = 1, 2, 3, 4, 5,...	-
f_{CAN}	f_{SPB}	1 : n ⁷⁾ ; n = 1, 2, 3, 4, 5,...	n = 1
f_{SPB}	f_{CAN}	1 : n ⁷⁾ ; n = 1, 2, 3, 4, 5,...	-
$f_{ASCLINF}$	f_{SPB}	1 : n ⁸⁾ ; n = 1, 2, 3, 4, 5,...	n = 2
f_{SPB}	$f_{ASCLINF}$	1 : n ⁸⁾ ; n = 1, 2, 3, 4, 5,...	-
$f_{ASCLINS}$	f_{SPB}	1 : n ⁹⁾ ; n = 1, 2, 3, 4, 5,...	n = 1
f_{SPB}	$f_{ASCLINS}$	1 : n ⁹⁾ ; n = 1, 2, 3, 4, 5,...	-

1) Configuring the registers CCUCON0, CCUCON1 and CCUCON2 to values that result in a violation of the allowed ratios are not allowed even if a single configuration entry for a bit field is with the allowed register bit field description.

2) f_{GTM} can be faster, slower, or equal to f_{SPB}

3) f_{STM} can be faster, slower, or equal to f_{SPB}

4) f_{BBB} has to be faster, or equal to f_{SPB}

5) f_{BAUD1} can be faster, slower, or equal to f_{SPB}

6) f_{BAUD2} can be faster, slower, or equal to f_{SPB}

7) f_{CAN} can be faster, slower, or equal to f_{SPB}

8) $f_{ASCLINF}$ can be faster, slower, or equal to f_{SPB}

9) $f_{ASCLINS}$ can be faster, slower, or equal to f_{SPB}

Note: Recommended values did not necessarily reflect the default configuration after a reset event. Instead they should give a hint how to configure the system for an optimal performance configuration.

CCU Registers

CCU registers may be accessed by any CPUs in the system. However, it is suggested that only one CPU is used to control the clocks. As CPU0 is the active and available CPU after a reset, this is the logical choice.

Note: The relation in between the described CCUCONx.yyyDIV bit fields is described in the table [CCU allowed Clock Ratios](#).

CCUCON0

CCU Clock Control Register 0 (030_H) **Reset Value: 0112 0148_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	UP	CLKSEL	0	FSIDIV	0	FSI2DIV					SPBDIV				
rh	w	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LPDIV				SRIDIV				BAUD2DIV				BAUD1DIV			
rw				rw				rw				rw			

Field	Bits	Type	Function
BAUD1DIV	[3:0]	rw	<p>Baud1 Divider Reload Value</p> <p>The resulting Baud1 frequency is configured to $f_{\text{BAUD1}} = f_{\text{source}} / \text{BAUD1DIV}$ for the allowed configurations. For BAUD1DIV = 0000_B the clock is shut off.</p> <p>f_{source} could be configured either to f_{PLL} (CLKSEL = 01_B) or f_{BACK} (CLKSEL = 00_B)</p> <p>0000_B. f_{BAUD1} is stopped</p> <p>0001_B. $f_{\text{BAUD1}} = f_{\text{source}}$</p> <p>0010_B. $f_{\text{BAUD1}} = f_{\text{source}}/2$</p> <p>0011_B. $f_{\text{BAUD1}} = f_{\text{source}}/3$</p> <p>0100_B. $f_{\text{BAUD1}} = f_{\text{source}}/4$</p> <p>0101_B. $f_{\text{BAUD1}} = f_{\text{source}}/5$</p> <p>0110_B. $f_{\text{BAUD1}} = f_{\text{source}}/6$</p> <p>0111_B Reserved, do not use this combination</p> <p>1000_B. $f_{\text{BAUD1}} = f_{\text{source}}/8$</p> <p>1001_B Reserved, do not use this combination</p> <p>1010_B. $f_{\text{BAUD1}} = f_{\text{source}}/10$</p> <p>1011_B Reserved, do not use this combination</p> <p>1100_B. $f_{\text{BAUD1}} = f_{\text{source}}/12$</p> <p>1101_B Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>1111_B. $f_{\text{BAUD1}} = f_{\text{source}}/15$</p>

Field	Bits	Type	Function
BAUD2DIV	[7:4]	rw	<p>Baud2 Divider Reload Value</p> <p>The resulting Baud2 frequency is configured to $f_{\text{BAUD2}} = f_{\text{source}} / \text{BAUD2DIV}$ for the allowed configurations. For BAUD2DIV = 0000_B the clock is shut off.</p> <p>f_{source} could be configured either to f_{PLL} (CLKSEL = 01_B) or f_{BACK} (CLKSEL = 00_B)</p> <p>0000_B f_{BAUD2} is stopped</p> <p>0001_B $f_{\text{BAUD2}} = f_{\text{source}}$</p> <p>0010_B $f_{\text{BAUD2}} = f_{\text{source}}/2$</p> <p>0011_B $f_{\text{BAUD2}} = f_{\text{source}}/3$</p> <p>0100_B $f_{\text{BAUD2}} = f_{\text{source}}/4$</p> <p>0101_B $f_{\text{BAUD2}} = f_{\text{source}}/5$</p> <p>0110_B $f_{\text{BAUD2}} = f_{\text{source}}/6$</p> <p>0111_B Reserved, do not use this combination</p> <p>1000_B $f_{\text{BAUD2}} = f_{\text{source}}/8$</p> <p>1001_B Reserved, do not use this combination</p> <p>1010_B $f_{\text{BAUD2}} = f_{\text{source}}/10$</p> <p>1011_B Reserved, do not use this combination</p> <p>1100_B $f_{\text{BAUD2}} = f_{\text{source}}/12$</p> <p>1101_B Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>1111_B $f_{\text{BAUD2}} = f_{\text{source}}/15$</p>

Field	Bits	Type	Function
SRIDIV	[11:8]	rw	<p>SRI Divider Reload Value</p> <p>The resulting SRI frequency is configured to $f_{SRI} = f_{source} / SRIDIV$ for the allowed configurations. For SRIDIV = 0000_B the clock is shut off. f_{source} could be configured either to f_{PLL} (CLKSEL = 01_B) or f_{BACK} (CLKSEL = 00_B)</p> <p>0000_B f_{SRI} is stopped</p> <p>0001_B $f_{SRI} = f_{source}$</p> <p>0010_B $f_{SRI} = f_{source}/2$</p> <p>0011_B $f_{SRI} = f_{source}/3$</p> <p>0100_B $f_{SRI} = f_{source}/4$</p> <p>0101_B $f_{SRI} = f_{source}/5$</p> <p>0110_B $f_{SRI} = f_{source}/6$</p> <p>0111_B Reserved, do not use this combination</p> <p>1000_B $f_{SRI} = f_{source}/8$</p> <p>1001_B Reserved, do not use this combination</p> <p>1010_B $f_{SRI} = f_{source}/10$</p> <p>1011_B Reserved, do not use this combination</p> <p>1100_B $f_{SRI} = f_{source}/12$</p> <p>1101_B Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>1111_B $f_{SRI} = f_{source}/15$</p>
LPDIV	[15:12]	rw	<p>Low Power Divider Reload Value</p> <p>0000_B f_{MAX}, f_{SRI}, f_{SPB}, and f_{BBB} are controlled by the dedicated CCUCONx bit fields</p> <p>0001_B f_{SRI}, f_{SPB}, and $f_{BBB} = f_{source}/30$; $f_{MAX} = f_{source}/15$</p> <p>0010_B f_{SRI}, f_{SPB}, and $f_{BBB} = f_{source}/60$; $f_{MAX} = f_{source}/30$</p> <p>0011_B f_{SRI}, f_{SPB}, and $f_{BBB} = f_{source}/120$; $f_{MAX} = f_{source}/60$</p> <p>0100_B f_{SRI}, f_{SPB}, and $f_{BBB} = f_{source}/240$; $f_{MAX} = f_{source}/120$</p> <p>0101_B Reserved, do not use this combination</p> <p>0110_B Reserved, do not use this combination</p> <p>0111_B Reserved, do not use this combination</p> <p>1000_B Reserved, do not use this combination</p> <p>1001_B Reserved, do not use this combination</p> <p>1010_B Reserved, do not use this combination</p> <p>1011_B Reserved, do not use this combination</p> <p>1100_B Reserved, do not use this combination</p> <p>1101_B Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>1111_B Reserved, do not use this combination</p>

Field	Bits	Type	Function
SPBDIV	[19:16]	rw	SPB Divider Reload Value 0000 _B Reserved, do not use this combination 0001 _B Reserved, do not use this combination 0010 _B $f_{SPB} = f_{source}/2$ 0011 _B $f_{SPB} = f_{source}/3$ 0100 _B $f_{SPB} = f_{source}/4$ 0101 _B $f_{SPB} = f_{source}/5$ 0110 _B $f_{SPB} = f_{source}/6$ 0111 _B Reserved, do not use this combination; resulting in $f_{SPB} = f_{source}/6$ 1000 _B $f_{SPB} = f_{source}/8$ 1001 _B Reserved, do not use this combination; resulting in $f_{SPB} = f_{source}/8$ 1010 _B $f_{SPB} = f_{source}/10$ 1011 _B Reserved, do not use this combination; resulting in $f_{SPB} = f_{source}/10$ 1100 _B $f_{SPB} = f_{source}/12$ 1101 _B Reserved, do not use this combination; resulting in $f_{SPB} = f_{source}/12$ 1110 _B Reserved, do not use this combination; resulting in $f_{SPB} = f_{source}/12$ 1111 _B $f_{SPB} = f_{source}/15$ f_{source} could be configured either to f_{PLL} (CLKSEL = 01 _B) or f_{BACK} (CLKSEL = 00 _B)
FSI2DIV	[21:20]	rw	FSI2 Divider Reload Value 00 _B f_{FSI2} is shut off 01 _B $f_{FSI2} = f_{SRI}$ 10 _B $f_{FSI2} = f_{SRI} / 2$ for SRIDIV = 0001 _B or 0010 _B , else $f_{FSI2} = f_{SRI}$ 11 _B $f_{FSI2} = f_{SRI} / 3$ for SRIDIV = 0001 _B or 0010 _B , else $f_{FSI2} = f_{SRI}$
FSIDIV	[25:24]	rw	FSI Divider Reload Value 00 _B f_{FSI} is shut off 01 _B $f_{FSI} = f_{SRI}$ 10 _B $f_{FSI} = f_{SRI} / 2$ for SRIDIV = 0001 _B or 0010 _B , else $f_{FSI} = f_{SRI}$ 11 _B $f_{FSI} = f_{SRI} / 3$ for SRIDIV = 0001 _B or 0010 _B , else $f_{FSI} = f_{SRI}$

Field	Bits	Type	Function
CLKSEL	[29:28]	rw	Clock Selection This bit field defines the clock source that is used for the clock generation of f_{SOURCE} . 00 _B back-up clock is used as clock source f_{source} 01 _B f_{PLL} is used as clock source f_{source} 10 _B Reserved, do not use this combination 11 _B Reserved, do not use this combination
UP	30	w	Update Request This bit defined the update for the CCU. Please note that setting this bit request a CCU update based on all three registers, so only one UP bit have to be set. 0 _B No action 1 _B A new complete parameter set is transferred to the CCU. All three registers CCUCON0, 1 and 5 content is taken by CCU. This bit always read as zero.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and can not be updated
0	[23:22] , [27:26]	rw	Reserved Should be written with 0.

Note: The relation in between the described CCUCONx.yyyDIV bit fields is described in the table [CCU allowed Clock Ratios](#).

CCUCON1
CCU Clock Control Register 1

 (034_H)

 Reset Value: 0000 2211_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	UP	INSEL		ASCLINSDIV				ASCLINFDIV				ETHDIV			
rh	w	rw		rw				rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GTMDIV				STMDIV				ERAYDIV				CANDIV			
rw				rw				rw				rw			

Field	Bits	Type	Function
CANDIV	[3:0]	rw	<p>MultiCAN Divider Reload Value</p> <p>The resulting MultiCAN frequency is configured to $f_{CAN} = f_{source} / CANDIV$ for the allowed configurations. For $CANDIV = 0000_B$ the clock is shut off. f_{source} could be configured either to f_{PLL} ($CLKSEL = 01_B$) or f_{BACK} ($CLKSEL = 00_B$)</p> <p>$0000_B f_{CAN}$ is stopped</p> <p>$0001_B f_{CAN} = f_{source}$</p> <p>$0010_B f_{CAN} = f_{source}/2$</p> <p>$0011_B f_{CAN} = f_{source}/3$</p> <p>$0100_B f_{CAN} = f_{source}/4$</p> <p>$0101_B f_{CAN} = f_{source}/5$</p> <p>$0110_B f_{CAN} = f_{source}/6$</p> <p>$0111_B$ Reserved, do not use this combination</p> <p>$1000_B f_{CAN} = f_{source}/8$</p> <p>$1001_B$ Reserved, do not use this combination</p> <p>$1010_B f_{CAN} = f_{source}/10$</p> <p>$1011_B$ Reserved, do not use this combination;</p> <p>$1100_B f_{CAN} = f_{source}/12$</p> <p>$1101_B$ Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>$1111_B f_{CAN} = f_{source}/15$</p>

Field	Bits	Type	Function
ERAYDIV	[7:4]	rw	<p>ERAY Divider Reload Value</p> <p>The resulting ERAY frequency is configured to $f_{ERAY} = f_{PLLERAY} / \text{ERAYDIV}$ for the allowed configurations. For ERAYDIV = 0000_B the clock is shut off.</p> <p>0000_B f_{ERAY} is stopped</p> <p>0001_B $f_{ERAY} = f_{PLLERAY}$</p> <p>0010_B $f_{ERAY} = f_{PLLERAY}/2$</p> <p>0011_B $f_{ERAY} = f_{PLLERAY}/3$</p> <p>0100_B $f_{ERAY} = f_{PLLERAY}/4$</p> <p>0101_B $f_{ERAY} = f_{PLLERAY}/5$</p> <p>0110_B $f_{ERAY} = f_{PLLERAY}/6$</p> <p>0111_B Reserved, do not use this combination</p> <p>1000_B $f_{ERAY} = f_{PLLERAY}/8$</p> <p>1001_B Reserved, do not use this combination</p> <p>1010_B $f_{ERAY} = f_{PLLERAY}/10$</p> <p>1011_B Reserved, do not use this combination</p> <p>1100_B $f_{ERAY} = f_{PLLERAY}/12$</p> <p>1101_B Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>1111_B $f_{ERAY} = f_{PLLERAY}/15$</p> <p>This bit field is not link to the update handshake defined by bits UP and LCK. An update of this bit field has no influence to the other clocks in the system. Always a new value (different number) is written to this bit field an update is done. If the same value is written again no update is started.</p>

Field	Bits	Type	Function
STMDIV	[11:8]	rw	<p>STM Divider Reload Value</p> <p>The resulting STM frequency is configured to $f_{STM} = f_{source} / \text{STMDIV}$ for the allowed configurations. For STMDIV = 0000_B the clock is shut off. f_{source} could be configured either to f_{PLL} (CLKSEL = 01_B) or f_{BACK} (CLKSEL = 00_B)</p> <p>0000_B f_{STM} is stopped</p> <p>0001_B $f_{STM} = f_{source}$</p> <p>0010_B $f_{STM} = f_{source}/2$</p> <p>0011_B $f_{STM} = f_{source}/3$</p> <p>0100_B $f_{STM} = f_{source}/4$</p> <p>0101_B $f_{STM} = f_{source}/5$</p> <p>0110_B $f_{STM} = f_{source}/6$</p> <p>0111_B Reserved, do not use this combination</p> <p>1000_B $f_{STM} = f_{source}/8$</p> <p>1001_B Reserved, do not use this combination</p> <p>1010_B $f_{STM} = f_{source}/10$</p> <p>1011_B Reserved, do not use this combination</p> <p>1100_B $f_{STM} = f_{source}/12$</p> <p>1101_B Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>1111_B $f_{STM} = f_{source}/15$</p>

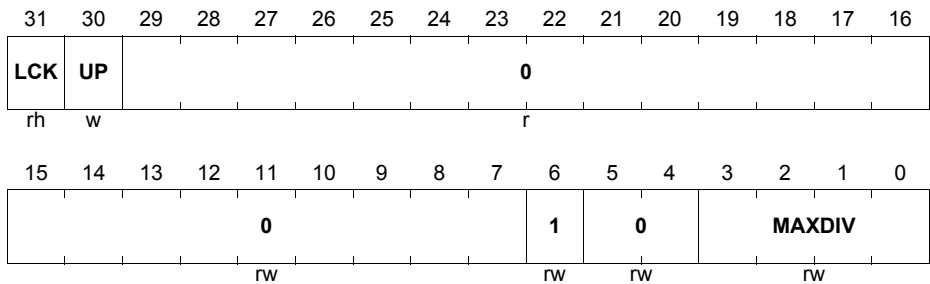
Field	Bits	Type	Function
GTMDIV	[15:12]	rw	<p>GTM Divider Reload Value</p> <p>The resulting GTM frequency is configured to $f_{\text{GTM}} = f_{\text{source}} / \text{GTMDIV}$ for the allowed configurations. For GTMDIV = 0000_B the clock is shut off. f_{source} could be configured either to f_{PLL} (CLKSEL = 01_B) or f_{BACK} (CLKSEL = 00_B)</p> <p>0000_B f_{GTM} is stopped</p> <p>0001_B $f_{\text{GTM}} = f_{\text{source}}$</p> <p>0010_B $f_{\text{GTM}} = f_{\text{source}}/2$</p> <p>0011_B $f_{\text{GTM}} = f_{\text{source}}/3$</p> <p>0100_B $f_{\text{GTM}} = f_{\text{source}}/4$</p> <p>0101_B $f_{\text{GTM}} = f_{\text{source}}/5$</p> <p>0110_B $f_{\text{GTM}} = f_{\text{source}}/6$</p> <p>0111_B Reserved, do not use this combination</p> <p>1000_B $f_{\text{GTM}} = f_{\text{source}}/8$</p> <p>1001_B Reserved, do not use this combination</p> <p>1010_B $f_{\text{GTM}} = f_{\text{source}}/10$</p> <p>1011_B Reserved, do not use this combination</p> <p>1100_B $f_{\text{GTM}} = f_{\text{source}}/12$</p> <p>1101_B Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>1111_B $f_{\text{GTM}} = f_{\text{source}}/15$</p>

Field	Bits	Type	Function
ETHDIV	[19:16]	rw	<p>Ethernet Divider Reload Value</p> <p>The resulting Ethernet frequency is configured to $f_{ETH} = f_{PLL\text{ERAY}} / \text{ETHDIV} * 4$ for the allowed configurations. For ETHDIV = 0000_B the clock is shut off.</p> <p>0000_B f_{ETH} is stopped</p> <p>0001_B $f_{ETH} = f_{PLL\text{ERAY}}/4$</p> <p>0010_B $f_{ETH} = f_{PLL\text{ERAY}}/8$</p> <p>0011_B $f_{ETH} = f_{PLL\text{ERAY}}/12$</p> <p>0100_B $f_{ETH} = f_{PLL\text{ERAY}}/16$</p> <p>0101_B $f_{ETH} = f_{PLL\text{ERAY}}/20$</p> <p>0110_B $f_{ETH} = f_{PLL\text{ERAY}}/24$</p> <p>0111_B Reserved, do not use this combination</p> <p>1000_B $f_{ETH} = f_{PLL\text{ERAY}}/32$</p> <p>1001_B Reserved, do not use this combination</p> <p>1010_B $f_{ETH} = f_{PLL\text{ERAY}}/40$</p> <p>1011_B Reserved, do not use this combination</p> <p>1100_B $f_{ETH} = f_{PLL\text{ERAY}}/48$</p> <p>1101_B Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>1111_B $f_{ETH} = f_{PLL\text{ERAY}}/60$</p> <p>This bit field is not link to the update handshake defined by bits UP and LCK. An update of this bit field has no influence to the other clocks in the system. Always a new value (different number) is written to this bit field an update is done. If the same value is written again no update is started.</p> <p>A 50% duty cycle clock is generated for f_{ETH}.</p>

Field	Bits	Type	Function
ASCLINFDIV	[23:20]	rw	<p>ASCLIN Fast Divider Reload Value</p> <p>The resulting ASCLIN frequency is configured to $f_{ASCLIN} = f_{source} / ASCLINFDIV$ for the allowed configurations. For ASCLINFDIV = 0000_B the clock is shut off.</p> <p>f_{source} could be configured either to f_{PLL} (CLKSEL = 01_B) or f_{BACK} (CLKSEL = 00_B)</p> <p>0000_B f_{ASCLIN} is stopped</p> <p>0001_B $f_{ASCLIN} = f_{source}$</p> <p>0010_B $f_{ASCLIN} = f_{source}/2$</p> <p>0011_B $f_{ASCLIN} = f_{source}/3$</p> <p>0100_B $f_{ASCLIN} = f_{source}/4$</p> <p>0101_B $f_{ASCLIN} = f_{source}/5$</p> <p>0110_B $f_{ASCLIN} = f_{source}/6$</p> <p>0111_B Reserved, do not use this combination</p> <p>1000_B $f_{ASCLIN} = f_{source}/8$</p> <p>1001_B Reserved, do not use this combination</p> <p>1010_B $f_{ASCLIN} = f_{source}/10$</p> <p>1011_B Reserved, do not use this combination</p> <p>1100_B $f_{ASCLIN} = f_{source}/12$</p> <p>1101_B Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>1111_B $f_{ASCLIN} = f_{source}/15$</p>

Field	Bits	Type	Function
ASCLNSDIV	[27:24]	rw	<p>ASCLIN Slow Divider Reload Value</p> <p>The resulting ASCLINS frequency is configured to $f_{ASCLINS} = f_{source} / ASCLNSDIV$ for the allowed configurations. For ASCLNSDIV = 0000_B the clock is shut off.</p> <p>f_{source} could be configured either to f_{PLL} (CLKSEL = 01_B) or f_{BACK} (CLKSEL = 00_B)</p> <p>0000_B $f_{ASCLINS}$ is stopped</p> <p>0001_B $f_{ASCLINS} = f_{source}$</p> <p>0010_B $f_{ASCLINS} = f_{source}/2$</p> <p>0011_B $f_{ASCLINS} = f_{source}/3$</p> <p>0100_B $f_{ASCLINS} = f_{source}/4$</p> <p>0101_B $f_{ASCLINS} = f_{source}/5$</p> <p>0110_B $f_{ASCLINS} = f_{source}/6$</p> <p>0111_B Reserved, do not use this combination</p> <p>1000_B $f_{ASCLINS} = f_{source}/8$</p> <p>1001_B Reserved, do not use this combination</p> <p>1010_B $f_{ASCLINS} = f_{source}/10$</p> <p>1011_B Reserved, do not use this combination</p> <p>1100_B $f_{ASCLINS} = f_{source}/12$</p> <p>1101_B Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>1111_B $f_{ASCLINS} = f_{source}/15$</p>
INSEL	[29:28]	rw	<p>Input Selection</p> <p>This bit field defines as clock source for the two PLLs (PLL and PLL_ERAY).</p> <p>00_B back-up clock is used as clock source</p> <p>01_B f_{OSC0} is used as clock source</p> <p>10_B Reserved, do not use this combination</p> <p>11_B Reserved, do not use this combination</p>
UP	30	w	<p>Update Request</p> <p>This bit defined the update for the CCU. Please note that setting this bit request a CCU update based on all three registers, so only one UP bit have to be set.</p> <p>0_B No action</p> <p>1_B A new complete parameter set is transferred to the CCU. All three registers CCUCON0, 1 and 5 content is taken by CCU.</p> <p>This bit always read as zero.</p>

Field	Bits	Type	Function
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and can not be updated

CCUCON5
CCU Clock Control Register 5
(04C_H)
Reset Value: 0000 0041_H


Field	Bits	Type	Function
MAXDIV	[3:0]	rw	<p>Max Divider Reload Value</p> <p>The resulting frequency is configured to $f_{MAX} = f_{source} / MAXDIV$. For $MAXDIV = 0000_B$ $f_{MAX} = f_{source}$ f_{source} could be configured either to f_{PLL} (CCUCON0.CLKSEL = 01_B) or f_{BACK} (CCUCON0.CLKSEL = 00_B)</p> <p>0000_B $f_{MAX} = f_{source}$ 0001_B $f_{MAX} = f_{source}$ 0010_B $f_{MAX} = f_{source}/2$ 0011_B $f_{MAX} = f_{source}/3$ 0100_B $f_{MAX} = f_{source}/4$ 0101_B $f_{MAX} = f_{source}/5$ 0110_B $f_{MAX} = f_{source}/6$ 0111_B Reserved, do not use this combination 1000_B $f_{MAX} = f_{source}/8$ 1001_B Reserved, do not use this combination 1010_B $f_{MAX} = f_{source}/10$ 1011_B Reserved, do not use this combination 1100_B $f_{MAX} = f_{source}/12$ 1101_B Reserved, do not use this combination 1110_B Reserved, do not use this combination 1111_B $f_{MAX} = f_{source}/15$</p> <p><i>Note: For power saving it's recommended to configure MAXDIV always with the same value as CCUCON0.SRIDIV. Otherwise MAXDIV could be set always to 0001_B.</i></p>
1	6	rw	<p>Reserved</p> <p>Should be written with 1.</p>
0	[5:4], [15:7]	rw	<p>Reserved</p> <p>Should be written with 0.</p>
0	[29:16]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

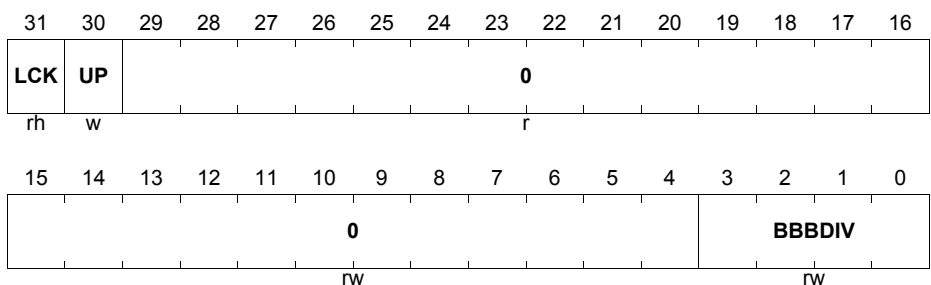
Field	Bits	Type	Function
UP	30	w	<p>Update Request</p> <p>This bit defined the update for the CCU. Please note that setting this bit request a CCU update based on all three registers, so only one UP bit have to be set.</p> <p>0_B No action 1_B A new complete parameter set is transferred to the CCU. All three registers CCUCON0, 1 and 5 content is taken by CCU.</p> <p>This bit always read as zero.</p>
LCK	31	rh	<p>Lock Status</p> <p>This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p>0_B The register is unlocked and can be updated 1_B The register is locked and can not be updated</p>

Note: The relation in between the described CCUCONx.yyyDIV bit fields is described in the table [CCU allowed Clock Ratios](#).

CCUCON2
CCU Clock Control Register 2

(040_H)

Reset Value: 0000 0002_H

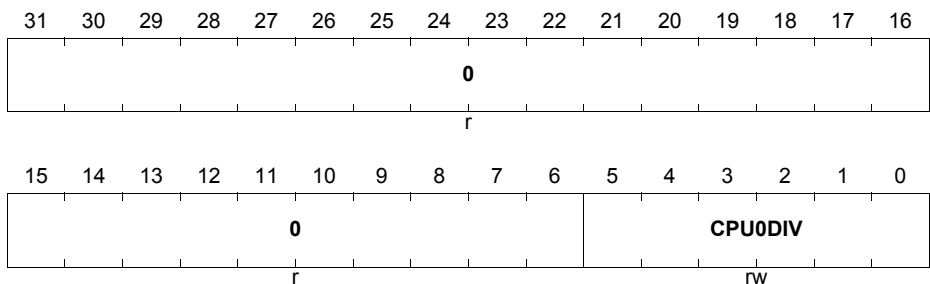


Field	Bits	Type	Function
BBBDIV	[3:0]	rw	<p>BBB Divider Reload Value</p> <p>The resulting BBB frequency is configured to $f_{\text{BBB}} = f_{\text{source}} / \text{BBBDIV}$ for all allowed configurations. For $\text{BBBDIV} = 0000_{\text{B}}$ the clock is shut off. f_{source} could be configured either to f_{PLL} ($\text{CLKSEL} = 01_{\text{B}}$) or f_{BACK} ($\text{CLKSEL} = 00_{\text{B}}$)</p> <p>$0000_{\text{B}}$ f_{BBB} is stopped</p> <p>0001_{B} $f_{\text{BBB}} = f_{\text{source}}$</p> <p>$0010_{\text{B}}$ $f_{\text{BBB}} = f_{\text{source}}/2$</p> <p>$0011_{\text{B}}$ $f_{\text{BBB}} = f_{\text{source}}/3$</p> <p>$0100_{\text{B}}$ $f_{\text{BBB}} = f_{\text{source}}/4$</p> <p>$0101_{\text{B}}$ $f_{\text{BBB}} = f_{\text{source}}/5$</p> <p>$0110_{\text{B}}$ $f_{\text{BBB}} = f_{\text{source}}/6$</p> <p>$0111_{\text{B}}$ Reserved, do not use this combination; resulting in $f_{\text{BBB}} = f_{\text{source}}/6$</p> <p>$1000_{\text{B}}$ $f_{\text{BBB}} = f_{\text{source}}/8$</p> <p>$1001_{\text{B}}$ Reserved, do not use this combination; resulting in $f_{\text{BBB}} = f_{\text{source}}/8$</p> <p>$1010_{\text{B}}$ $f_{\text{BBB}} = f_{\text{source}}/10$</p> <p>$1011_{\text{B}}$ Reserved, do not use this combination; resulting in $f_{\text{BBB}} = f_{\text{source}}/10$</p> <p>$1100_{\text{B}}$ $f_{\text{BBB}} = f_{\text{source}}/12$</p> <p>$1101_{\text{B}}$ Reserved, do not use this combination; resulting in $f_{\text{BBB}} = f_{\text{source}}/12$</p> <p>$1110_{\text{B}}$ Reserved, do not use this combination; resulting in $f_{\text{BBB}} = f_{\text{source}}/12$</p> <p>$1111_{\text{B}}$ $f_{\text{BBB}} = f_{\text{source}}/15$</p>
UP	30	w	<p>Update Request</p> <p>This bit defined the update for the CCU. Please note that setting this bit request a CCU update based on this register.</p> <p>0_{B} No action</p> <p>1_{B} A new complete parameter set is transferred to the CCU.</p> <p>This bit always read as zero.</p>

Field	Bits	Type	Function
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and can not be updated
0	[15:4]	rw	Reserved Should be written with 0.
0	[29:16]	r	Reserved Read as 0; should be written with 0.

CCUCON6
CCU Clock Control Register 6

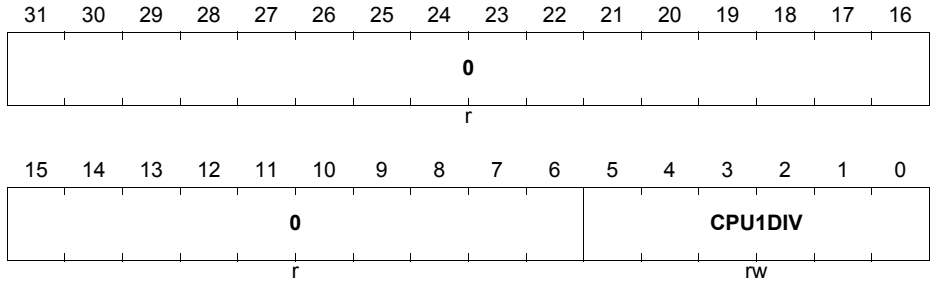
 (080_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Function
CPU0DIV	[5:0]	rw	CPU0 Divider Reload Value The resulting CPU0 frequency (performance) is configured to $f_{CPU0} = f_{SRI} * (64 - CPU0DIV) / 64$. For CPU0DIV = 000000 _B , $f_{CPU0} = f_{SRI}$.
0	[31:6]	r	Reserved Read as 0; should be written with 0.

CCUCON7
CCU Clock Control Register 7

 (084_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Function
CPU1DIV	[5:0]	rw	CPU1 Divider Reload Value The resulting CPU1 frequency (performance) is configured to $f_{CPU1} = f_{SRI} * (64 - CPU1DIV) / 64$. For CPU1DIV = 000000 _B $f_{CPU1} = f_{SRI}$.
0	[31:6]	r	Reserved Read as 0; should be written with 0.

7.1.4 Individual Clock Generation

Each module provides some individual clock control options. Most modules provide a Clock Control (CLC) Register for this purpose. Some modules provide in addition more options to control the clocks. For more details please see the dedicated module chapters.

Modules can have in addition other registers controlling sub-clock domains e.g. for baud rate generation. These registers are always defined in the module chapters.

7.1.4.1 Clock Control Register CLC

All CLC registers have basically the same bit and bit field layout. However, not all CLC register functions are implemented for each peripheral module. **Table 7-3** defines in detail which bits and bit fields of the CLC registers are implemented for each clock control register.

The following functions for the module are associated with the CLC register:

- Peripheral clock static on/off control
- Module clock behavior in Sleep Mode

Module Enable/Disable Control

If a module is not used at all by an application, it can be completely shut off by setting bit DISR in its CLC register. For peripheral modules with a run mode clock divider field RMC, a second option to completely switch off the module is to set bit field RMC to 00_H. This also disables the module's operation.

The status bit DISS always indicates whether a module is currently switched off (DISS = 1) or switched on (DISS = 0).

Write operations to the non CLC registers of disabled modules are not allowed. However, the CLC of a disabled module can be written. An attempt to write to any of the other writable registers of a disabled module except CLC will cause the corresponding Bus Control Unit (BCU) to generate a bus error.

A read operation of registers of a disabled module is allowed and does not generate a bus error.

When a disabled module is switched on by writing an appropriate value to its MOD_CLC register (DISR = 0 and RMC (if implemented) > 0), status bit DISS changes from 1 to 0. During the phase in which the module becomes active, any write access to corresponding module registers (while DISS is still set) will generate a bus error. When enabling a disabled module, application software should read back the CLC register once, to check that DISS is cleared, before writing to any module register (including the CLC register).

Sleep Mode Control

The EDIS bit in the CLC register controls whether or not a module is stopped during Sleep Mode. If EDIS is 0, a Sleep Mode request can be recognized by the module and, when received, its clock is shut off.

If EDIS is set to 1, a Sleep Mode request is disregarded by the module and the module continues its operation.

Entering Disabled Mode

Software can request that a peripheral unit be put into Disabled Mode by setting DISR. A module will also be put into Disabled Mode if the Sleep Mode is requested and the module is configured to allow Sleep Mode.

In Secure Shut-off Mode, a module first finishes any operation in progress, then proceeds with an orderly shut down. When all sub-components of the module are ready to be shut down, the module's clock control unit turns off the clock. The status bit DISS is updated by the peripheral unit accordingly.

Module Clock Divider Control

Peripheral modules of the TC26x B-Step can have a RMC control bit field in their CLC registers. This Run Mode Clock control bit field makes it possible to slow down the CLC clock via a programmable clock divider circuit.

A value of 00_H in RMC disables the clock signals to these modules (CLC clock is switched off). If RMC is not equal to 00_H, the clock for a module register (f_{CLC}) accesses is generated as

(7.23)

$$f_{CLC} = \frac{f_{SPB}}{RMC}$$

where RMC is the content of its CLC register RMC field with a range of 1 to 255. If RMC is not available in a CLC register, the CLC clock frequency f_{CLC} is always equal to the frequency of f_{SPB} .

Note: The number of module clock cycles (wait states) that are required for a "destructive read" access (means: flags/bits are set/cleared by a read access) to a module register of a peripheral unit depends on the selected CLC clock frequency.

Therefore, a slower CLC clock (selected via bit field RMC in the CLC register) may result in a longer read cycle access time on the SPB for peripheral units with "destructive read" access.

Module Clock Register Implementations

Table 7-3 shows which of the CLC register bits/bit fields are implemented for each peripheral module in the TC26x B-Step.

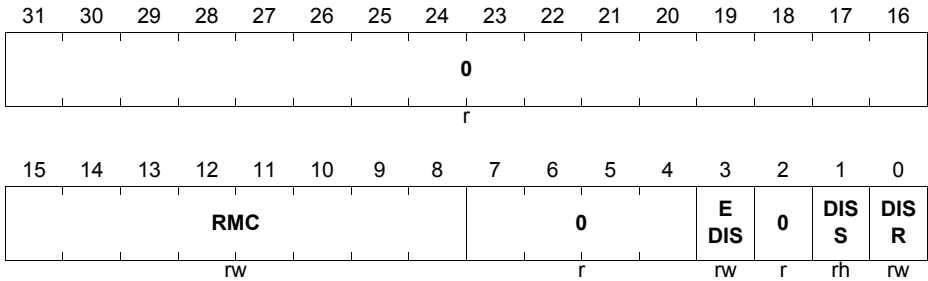
Table 7-3 Clock Generation Implementation of the TC26x B-Step Peripheral Modules

Module	DISR Bit 0	DISS Bit 1	EDIS Bit 3	RMC
VADC	✓	✓	✓	–
DSADC	✓	✓	✓	–
ASCLINx	✓	✓	✓	–
QSPIx	✓	✓	✓	–
GPT12x	✓	✓	✓	–
MultiCAN	✓	✓	✓	–
DMA	✓	✓	✓	–
MSCx	✓	✓	✓	–
STMx	✓	✓	✓	–
ERAY	✓	✓	✓	8-bit
FCE	✓	✓	–	–
LMU	✓	✓	–	–
GTM	✓	✓	–	–
CCU6x	✓	✓	✓	–
SENT	✓	✓	✓	–
CAN	✓	✓	✓	–
I2C	✓	✓	✓	8-bit
HSSL	✓	✓	✓	–
HSCT	✓	✓	✓	–
Ethernet	✓	✓	–	–
IOM	✓	✓	✓	8-bit
PSI5	✓	✓	–	–
PSI5S	✓	✓	–	–

Module CLC Register
MOD_CLC
Clock Control Register

 (00_H)

Reset Value: Module-specific



Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module. 0 _B Module disable is not requested 1 _B Module disable is requested
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module 0 _B Module is enabled 1 _B Module is disabled If the RMC field is implemented and if it is 0, DISS is set automatically.
EDIS	3	rw	Sleep Mode Enable Control Used for module Sleep Mode control. 0 _B Sleep Mode request is regarded. Module is enabled to go into Sleep Mode. 1 _B Sleep Mode request is disregarded: Sleep Mode cannot be entered on a request.
RMC	[15:8]	rw	8-Bit Clock Divider Value in RUN Mode This is a maximum 8-bit divider value for clock f_{SPB} . If RMC is set to 0 the module is disabled.
0	2, [7:4], [31:16]	r	Reserved Read as 0; should be written with 0.

7.1.5 Clock Monitors

For safety reasons clock monitors are available. For the following safety relevant clocks in the system monitors are present:

 f_{PLL}
 f_{PLL_ERAY}
 f_{SRI}
 f_{SPB}
 f_{GTM}
 f_{STM}

Each of these clocks is monitored by its own counter.

As reference clock the back-up clock is used as diverse clock source.

The divider has to be set in a way that the monitored frequency is one of the possible frequencies of [Table 7-4](#).

Operating the Clock Monitors

The clock monitors are implemented in a way that safety system requirements are supported for a flexible system configuration of the clock system. So it is possible to adapt the monitor configuration to the application configuration of the clock system.

For each clock monitor two bit fields are available, either in register CCUCON3 or CCUCON4 depending on the monitor.

For each monitor a divider bit field and a select bit field are available for the configuration.

The CCUCONy.SELXXX bit field defines one target frequency out of four predefined options.

Bit field CCUCONy.DIVXXX is used to divide the real clock frequency down to the selected target frequency.

Example: if the SPB is configured to operate at 100 MHz and needs to be monitored, CCUCON3.SBPDIV should be written with 0x14 and CCUCON3.SBPSEL with 00_B. This selects a target monitoring frequency of 5 MHz and divides the SPB clock by 20 resulting also in 5 MHz. If the SPB is configured to operate at 20 MHz and needs to be monitored, CCUCON3.SBPDIV should be written with 0x04 and CCUCON3.SBPSEL with 00_B.

The four target frequencies are selected in a way that all normally used frequency in a system can be covered. This means that frequencies like 66 MHz, 120 MHz, 130 MHz, 133 MHz, ... can be monitored.

Goal of the clock monitoring is to detect and signal clock malfunctions before these errors lead to not longer working system. Therefore the back-up clock is trimmed automatically and operated on f_{BACKT} . This setup guarantees that error are detected before the system could not react anymore

Table 7-4 Target trimmed Check limits

Target Frequency	LOWER value	UPPER value	SELXXX	Error can be detected for min. deviation	Error is detected for min. deviation
7.5 MHz	0x24	0x27	11 _B	-1.23% +1.56%	-8.56% +9.44%
6.6 MHz	0x20	0x23	10 _B	-0.9% +2.38%	-8.59% +11.11%
6 MHz	0x1C	0x1F	01 _B	-3.23% +1.56%	-11.13% +10.11%
5 MHz	0x17	0x1A	00 _B	-3.9% +2.83%	-12.41% +12.11%

An error is detected if the reference counter has an overflow and the monitor counter of the dedicated clock is either below the lower limit (clock too slow) or greater as the upper limit (clock too fast). Errors are signaled to the SMU where the further reaction can be configured.

In addition there is a fast upper limit checking for the SPB frequency available. This limit checking checks only for high SPB compared to the maximum allowed value. A violation generates a trigger.

An error is detected when the reference counter generates an overflow AND:

- Counter_value - UPPER > 0 for the upper limit
- OR
- LOWER - Counter_value > 0 for the upper limit

Note: This feature is supported by the Infineon safety driver [saflib] and there is no additional customer software required.

7.1.5.1 Clock Monitor Registers

These registers can be accessed by all CPUs in the system. Anyway it is suggested that only one CPU is used to control the clocks. As CPU0 is the active and available CPU after each reset this is the best choice.

CCUCON3

CCU Clock Control Register 3

 (044_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	UP	SLC K	0				SRISEL		SRIDIV						
rh	w	rw	r				rw		rw						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLL _{ERAY} SEL		PLL _{ERAY} DIV						PLLSEL		PLLDIV					
rw		rw						rw		rw					

Field	Bits	Type	Function
PLLDIV	[5:0]	rw	PLL Divider Value The resulting monitoring frequency is configured to $f_{\text{PLLMON}} = f_{\text{PLL}} / \text{PLLDIV}$. For PLLDIV = 0000 _B the monitoring is disabled.
PLLSEL	[7:6]	rw	PLL Target Monitoring Frequency Selection 00 _B 5 MHz is selected as target monitoring frequency 01 _B 6 MHz is selected as target monitoring frequency 10 _B 6.6 MHz is selected as target monitoring frequency 11 _B 7.5 MHz is selected as target monitoring frequency
PLL _{ERAY} DIV	[13:8]	rw	PLL_{ERAY} Divider Value The resulting monitoring frequency is configured to $f_{\text{PLLERAYMON}} = f_{\text{PLLERAYERAYDIV}$. For PLL _{ERAY} DIV = 0000 _B the monitoring is disabled.

Field	Bits	Type	Function
PLLERAYSEL	[15:14]	rw	<p>PLL_ERAY Target Monitoring Frequency Selection</p> <p>00_B 5 MHz is selected as target monitoring frequency</p> <p>01_B 6 MHz is selected as target monitoring frequency</p> <p>10_B 6.6̄ MHz is selected as target monitoring frequency</p> <p>11_B 7.5 MHz is selected as target monitoring frequency</p> <p>This bit field is not link to the update handshake defined by bits UP and LCK. Always a new value (different number) is written to this bit field an update is done. If the same value is written again no update is started.</p>
SRIDIV	[21:16]	rw	<p>SRI Divider Value</p> <p>The resulting monitoring frequency is configured to $f_{SRIMON} = f_{SRI} / SRIDIV$. For SRIDIV = 0000_B the monitoring is disabled.</p>
SRISEL	[23:22]	rw	<p>SRI Target Monitoring Frequency Selection</p> <p>00_B 5 MHz is selected as target monitoring frequency</p> <p>01_B 6 MHz is selected as target monitoring frequency</p> <p>10_B 6.6̄ MHz is selected as target monitoring frequency</p> <p>11_B 7.5 MHz is selected as target monitoring frequency</p>
SLCK	29	r	<p>Security Lock</p> <p>Bit is implemented but not functional in this device</p>

Field	Bits	Type	Function
UP	30	w	Update Request This bit triggers the update for the CCU, beside CCUCON3.PLLERAY. Please note that setting this bit request a CCU update based on both registers (CCUCON3 and CCUCON4), so only one UP bit have to be set. 0_B No action 1_B A new complete parameter set is transferred to the CCU. Both registers CCUCON3 and 4 content is taken by CCU. This bit always read as zero.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0_B The register is unlocked and can be updated 1_B The register is locked and can not be updated
0	[28:24]	r	Reserved Read as 0; should be written with 0.

CCUCON4
CCU Clock Control Register 4
(048_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	UP	SLC K	0				STMSEL		STMDIV						
rh	w	rw	r				rw		rw						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GTMSEL		GTMDIV				SPBSEL		SPBDIV							
rw		rw				rw		rw							

Field	Bits	Type	Function
SPBDIV	[5:0]	rw	SPB Divider Value The resulting monitoring frequency is configured to $f_{\text{SPBMON}} = f_{\text{SPB}} / \text{SPBDIV}$. For SPBDIV = 0000 _B the monitoring is disabled.
SPBSEL	[7:6]	rw	SPB Target Monitoring Frequency Selection 00 _B 5 MHz is selected as target monitoring frequency 01 _B 6 MHz is selected as target monitoring frequency 10 _B 6.6 MHz is selected as target monitoring frequency 11 _B 7.5 MHz is selected as target monitoring frequency
GTMDIV	[13:8]	rw	GTM Divider Value The resulting monitoring frequency is configured to $f_{\text{GTMMON}} = f_{\text{GTM}} / \text{GTMDIV}$. For GTMDIV = 0000 _B the monitoring is disabled.
GTMSSEL	[15:14]	rw	GTM Target Monitoring Frequency Selection 00 _B 5 MHz is selected as target monitoring frequency 01 _B 6 MHz is selected as target monitoring frequency 10 _B 6.6 MHz is selected as target monitoring frequency 11 _B 7.5 MHz is selected as target monitoring frequency
STMDIV	[21:16]	rw	STM Divider Value The resulting monitoring frequency is configured to $f_{\text{STMMON}} = f_{\text{STM}} / \text{STMDIV}$. For STMDIV = 0000 _B the monitoring is disabled.

Field	Bits	Type	Function
STMSEL	[23:22]	rw	STM Target Monitoring Frequency Selection 00 _B 5 MHz is selected as target monitoring frequency 01 _B 6 MHz is selected as target monitoring frequency 10 _B 6.6 MHz is selected as target monitoring frequency 11 _B 7.5 MHz is selected as target monitoring frequency
SLCK	29	rw	Security Lock Functionality not available in this device.
UP	30	w	Update Request This bit triggers the update for the CCU, beside CCUCON3.PLLERAY. Please note that setting this bit request a CCU update based on both registers (CCUCON3 and CCUCON4), so only one UP bit have to be set. 0 _B No action 1 _B A new complete parameter set is transferred to the CCU. Both registers CCUCON3 and 4 content is taken by CCU. This bit always read as zero.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and can not be updated
0	[28:24]	r	Reserved Read as 0; should be written with 0.

7.1.6 Clock Emergency Behavior

In case of a clock error the CCU switches to the back-up clock f_{BACK} as clock source. The only exception here is the ERAY clock f_{ERAY} that stays operating on the PLL_ERAY. A clock error is defined by a loss of lock event of the PLL (not PLL_ERAY) while operating in Normal Mode or by an error of the PLL clock monitor.

Note: After a clock emergency bit field CCUCON0.CLKSEL has to be re-written following a reconfiguration of the clock system (see also [Page 7-88](#)) when the root cause for the clock error disappears.

Note: After a clock emergency bit field PLLCON0.SETFINDIS / PLLERAYCON0.SETFINDIS has to be set following by PLLCON0.CLRFINDIS / PLLERAYCON0.CLRFINDIS clear before a reconfiguration of the clock system (see also [Page 7-88](#)) when the root cause for the clock error disappears.

7.1.7 External Clock Output

Two external clock outputs are provided via pins EXTCLK0 and EXTCLK1. These external clocks can be enabled/disabled via bits EXTCON.EN0 for EXTCLK0 and EXTCON.EN1 for EXTCLK1. Each of the clocks that defines a clock domain can individually be selected to be seen at pins EXTCLK0 or EXTCLK1, this is configured via bit field EXTCON.SEL0/1. Changing the content of bit field EXTCON.SEL0/1 can lead to spikes at pins EXTCLK0/1.

7.1.7.1 Programmable Frequency Output for EXTCLK0

This section describes the external clock generation using the Fractional Divider.

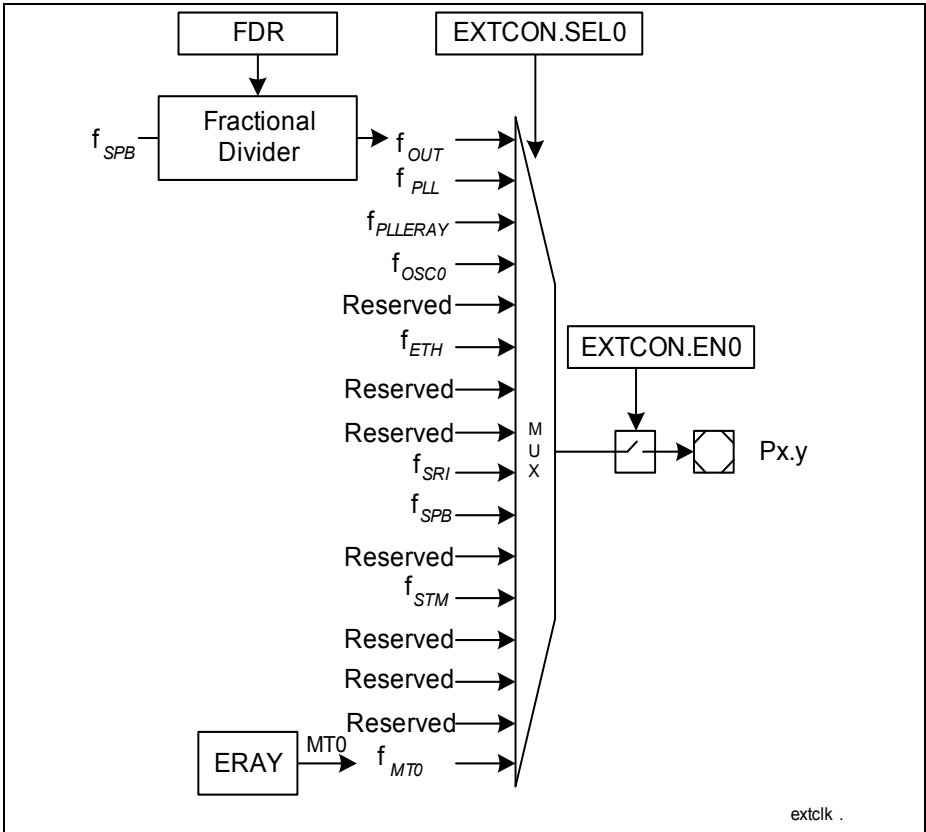


Figure 7-15 EXTCLK0 Generation

Overview

The fractional divider makes it possible to generate a external clock from the SPB clock using a programmable divider. The fractional divider divides the input clock f_{SPB} either by the factor $1/n$ or by a fraction of $n/1024$ for any value of n from 0 to 1023. This clock is thereafter divided additionally by a factor of two to guarantee a 50% duty cycle and outputs the clock, f_{OUT} . The fractional divider is controlled by the FDR register. [Figure 7-16](#) shows the fractional divider block diagram.

The adder logic of the fractional divider can be configured for two operating modes:

- Reload counter (addition of +1), generating an output clock pulse on counter overflow
- Adder that adds a STEP value to the RESULT value and generates an output clock pulse on counter overflow

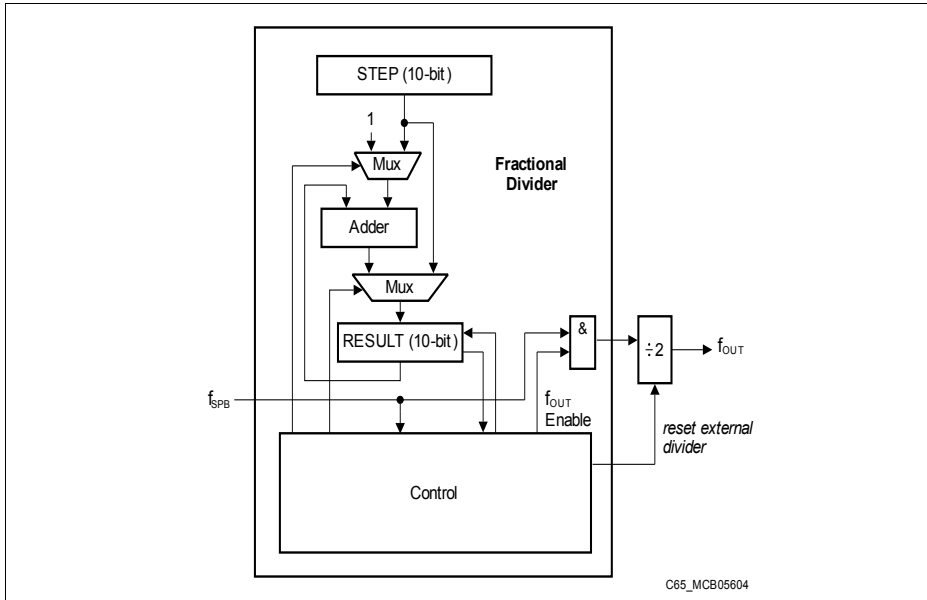


Figure 7-16 Fractional Divider Block Diagram

The adder logic of the fractional divider can be configured for two operating modes:

- **Normal Mode:** Reload counter (RESULT = RESULT + 1), generating an output clock pulse on counter overflow.
- **Fractional Divider Mode:** Adder that adds a STEP value to the RESULT value and generates an output clock pulse on counter overflow.

Fractional Divider Operating Modes

The fractional divider has two operating modes:

- Normal Divider Mode
- Fractional Divider Mode

Normal Divider Mode

In Normal Divider Mode (FDR.DM = 01_B), the fractional divider behaves as a reload counter (addition of +1) that generates an output clock pulse on the transition from 3FF_H to 000_H. FDR.RESULT represents the counter value and FDR.STEP determines the reload value.

The output frequencies in Normal Divider Mode are defined according to the following formulas:

$$f_{\text{OUT}} = \frac{f_{\text{SPB}} \times \frac{1}{n}}{2}, \text{ with } n = 1024 - \text{STEP} \quad (7.24)$$

In order to get $f_{\text{OUT}} = f_{\text{SPB}}/2$ STEP must be programmed with $3FF_{\text{H}}$.

Fractional Divider Mode

When the Fractional Divider Mode is selected ($\text{FDR.DM} = 10_{\text{B}}$), the output is derived from the input clock f_{SPB} by division of a fraction of $n/1024$ for any value of n from 0 to 1023 followed by the division of two. In general, the Fractional Divider Mode makes it possible to program the average output clock frequency with a higher accuracy than in Normal Divider Mode.

In Fractional Divider Mode, a pulse is generated depending on the result of the addition $\text{FDR.RESULT} + \text{FDR.STEP}$. If the addition leads to an overflow over $3FF_{\text{H}}$, a pulse is generated for the divider by two. Note that in Fractional Divider Mode the clock f_{OUT} can have a maximum period jitter of one f_{SPB} clock period.

The output frequencies in Fractional Divider Mode are defined according to the following formulas:

$$f_{\text{OUT}} = \frac{f_{\text{SPB}} \times \frac{n}{1024}}{2}, \text{ with } n = \text{FDR.STEP} = 0-1023 \quad (7.25)$$

7.1.7.2 Programmable Frequency Output for EXTCLK1

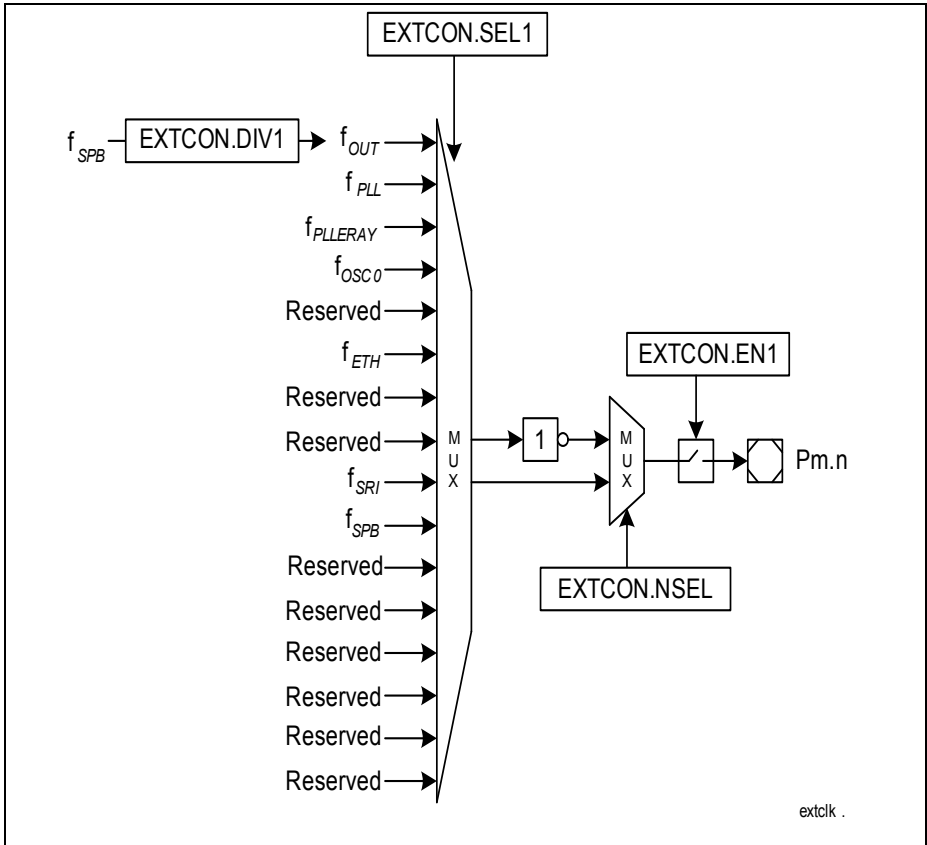


Figure 7-17 EXTCLK1 Generation

Clock f_{OUT} is generated via a counter, so the output frequency can be selected in small steps.

f_{OUT} always provides complete output periods.

Register EXTCON provides control over the output generation (frequency, activation).

7.1.7.3 Clock Output Control Register

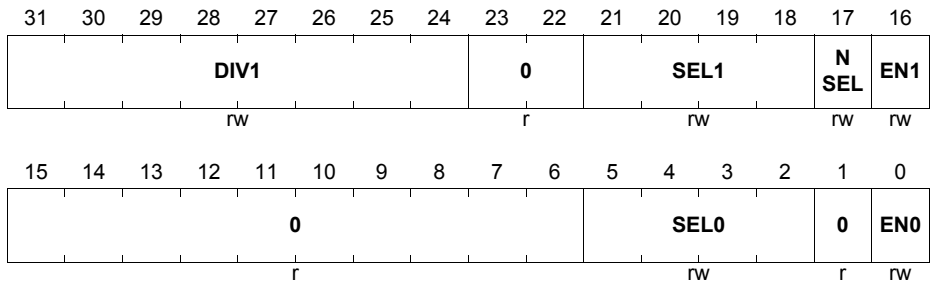
These registers can be accessed by all CPUs in the system. Anyway it is suggested that only one CPU is used to control the clocks. As CPU0 is the active and available CPU after each reset this is the best choice.

EXTCON

External Clock Control Register

(03C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
EN0	0	rw	External Clock Enable for EXTCLK0 0 _B No external clock is provided 1 _B The configured external clock is provided

Field	Bits	Type	Description
SEL0	[5:2]	rw	<p>External Clock Select for EXTCLK0</p> <p>This bit field defines the clock source that is selected as output for pin EXTCLK0.</p> <p>0000_B.f_{OUT} is selected for the external clock signal</p> <p>0001_B.f_{PLL} is selected for the external clock signal</p> <p>0010_B.$f_{PLLERAY}$ is selected for the external clock signal</p> <p>0011_B.f_{OSCO} is selected for the external clock signal</p> <p>0100_B Reserved, do not use this combination</p> <p>0101_B.f_{ETH} is selected for the external clock signal</p> <p>0110_B Reserved, do not use this combination</p> <p>0111_B Reserved, do not use this combination</p> <p>1000_B.f_{SRI} is selected for the external clock signal</p> <p>1001_B.f_{SPB} is selected for the external clock signal</p> <p>1010_B Reserved, do not use this combination</p> <p>1011_B.f_{STM} is selected for the external clock signal</p> <p>1100_B Reserved, do not use this combination</p> <p>1101_B Reserved, do not use this combination</p> <p>1110_B Reserved, do not use this combination</p> <p>1111_B.f_{MTO} from the ERAY module is selected for the external clock signal</p>
EN1	16	rw	<p>External Clock Enable for EXTCLK1</p> <p>0_B No external clock is provided</p> <p>1_B The configured external clock is provided</p>
NSEL	17	rw	<p>Negation Selection</p> <p>0_B The external clock is inverted</p> <p>1_B The external clock is not inverted</p>

Field	Bits	Type	Description
SEL1	[21:18]	rw	External Clock Select for EXTCLK1 This bit field defines the clock source that is selected as output for pin EXTCLK1. 0000 _B f_{OUT} is selected for the external clock signal 0001 _B f_{PLL} is selected for the external clock signal 0010 _B $f_{PLL\text{ERAY}}$ is selected for the external clock signal 0011 _B f_{OSCO} is selected for the external clock signal 0100 _B Reserved, do not use this combination 0101 _B f_{ETH} is selected for the external clock signal 0110 _B Reserved, do not use this combination 0111 _B Reserved, do not use this combination 1000 _B f_{SRI} is selected for the external clock signal 1001 _B f_{SPB} is selected for the external clock signal 1010 _B Reserved, do not use this combination 1011 _B Reserved, do not use this combination 1100 _B Reserved, do not use this combination 1101 _B Reserved, do not use this combination 1110 _B Reserved, do not use this combination 1111 _B Reserved, do not use this combination
DIV1	[31:24]	rw	External Clock Divider for EXTCLK1 This value defines the reload value of the divider that generates f_{OUT} out of f_{SPB} ($f_{OUT} = f_{SPB}/(DIV1+1)$). The divider itself is cleared each time bit EN1 is cleared.
0	1, [15:6], [23:22]	r	Reserved Read as 0; should be written with 0.

FDR
Fractional Divider Register
(038_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIS CLK		0				RESULT									
rwh		r				rh									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DM		0				STEP									
rw		r				rw									

Field	Bits	Type	Description
STEP	[9:0]	rw	<p>Step Value</p> <p>In Normal Divider Mode, STEP contains the reload value for RESULT.</p> <p>In Fractional Divider Mode, this bit field determines the 10-bit value that is added to RESULT with each input clock cycle.</p>
DM	[15:14]	rw	<p>Divider Mode</p> <p>This bit fields determines the functionality of the fractional divider block.</p> <p>00_B Fractional divider is switched off; no output clock is generated. The Reset External Divider signal is 1. RESULT is not updated (default after System Reset).</p> <p>01_B Normal Divider Mode selected.</p> <p>10_B Fractional Divider Mode selected.</p> <p>11_B Fractional divider is switched off; no output clock is generated. RESULT is not updated.</p>
RESULT	[25:16]	rh	<p>Result Value</p> <p>In Normal Divider Mode, RESULT acts as reload counter (addition +1).</p> <p>In Fractional Divider Mode, this bit field contains the result of the addition RESULT + STEP.</p> <p>If DM is written with 01_B or 10_B, RESULT is loaded with 3FF_H.</p>
DISCLK	31	rwh	<p>Disable Clock</p> <p>0_B Clock generation of f_{OUT} is enabled according to the setting of bit field DM.</p> <p>1_B Fractional divider is stopped. No change except when writing bit field DM.</p>
0	[13:10], [30:26]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

7.1.8 Clock Generation Unit

The Clock Generation Unit (CGU) allows a very flexible clock generation for the TC26x B-Step. During user program execution the frequency can be programmed for an optimal ratio between performance and power consumption.

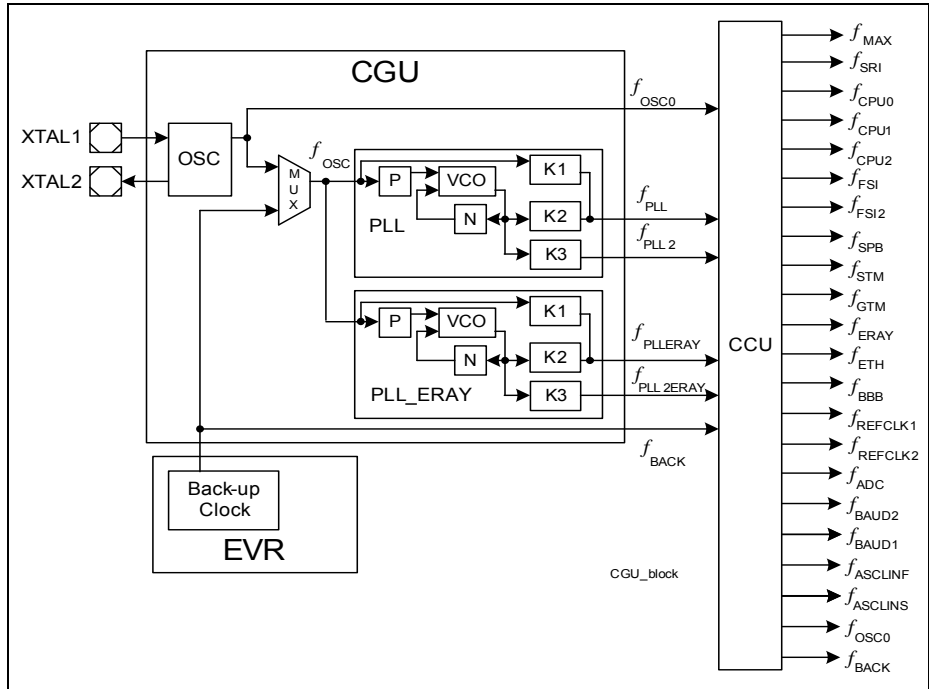


Figure 7-18 Clock Generation Unit Block Diagram

The CGU includes the clock source generation, the clock upscaling beside the clock distribution.

7.1.8.1 Example Sequence

The following sequence give an example for the clock ramp-up.

Goal for normal target setting is that system clock is based on PLL with external crystal.

- After power-on and system reset:
 - the system operates on the back-up clock
 - the oscillator (OSC) needs to be enabled via software or was enabled by firmware via BMI settings
- Fast clocks (SRI-Bus and CPUs) run with approximately 100 MHz

- Peripherals and SPB-Bus run with approximately 50 MHz
- Step 1: If f_{OSCO} is to be used but is not running, enable the oscillator to wait until it is providing a stable clock
- Step 2: Initialize the PLL to target f_{VCO} and f_{PLL} frequency
 - Select PLL input clock via CCUCON1.INSEL
 - Disconnect input clock from VCO; set PLLCON0.SETFINDIS
 - Select P, K2 / K3, and N divider for final target VCO and PLL frequency
 - 1) Select P as small as possible
 - 2) Select N (VCO) as high as possible
 - Connect input clock to VCO; set PLLCON0.CLRFINDIS
 - Configure resulting K2
- Step 3: Wait for PLL lock to be set
- Step 4: Configure CCUCON0, CCUCON1 and CCUCON5 to first target setting
 - Hint: both registers CCUCON0 and CCUCON1 can be reprogrammed without changing the clocking. Only setting the UP bit in one of the two registers transfers the values for both registers and lead to changes in the clock system.
 - Register CCUCON2 has its own UP bit
 - Bit fields CCUCON1.ERAY and CCUCON1.ETH are excluded from the update via UP in CCUCON0 or CCUCON1 and are always updated when a new value is programmed
- Step 5: Switch CCU input clock f_{SOURCE} to PLL via CCUCON0.CLKSEL
- Step 6: After setting CCU f_{SOURCE} to f_{PLL} , frequency has to be increased step by step to target frequency

Important hints for steps 4 and 5;

- Select CCUCON1 and CCUCON0 register settings in a way that the current consumption of the 1.3 V domain does not change more than the desired target value of the application, e.g. 50 mA.
- After every frequency programming step a wait time is recommended until supply ripple caused by supply current transient is faded away.

Note: The wait time between frequency steps depends on the supply and block concept.

- Continue these CCUCON1 and CCUCON0 update steps until the target system frequency is reached.

7.1.9 CCU Register Address

Table 7-5 Registers Address Spaces - CCU Registers

Module	Base Address	End Address	Note
SCU	F003 6000 _H	F003 63FF _H	-

7.1.10 CCU Kernel Registers

This section describes the CCU kernel registers of the SCU module. Most of CCU kernel register names described in this section will be referenced in other parts of the TC26x B-Step User's Manual by the module name prefix "SCU_".

CCU Kernel Register Overview

Table 7-6 Register Overview of CCU Registers (Offset from SCU Base)

Short Name	Long Name	Offset Addr. 1)	Access Mode		Reset	Description See
			Read	Write		
OSCCON	OSC Control Register	010 _H	U, SV	SV, SE, P	System Reset	Page 7-7
PLLSTAT	PLL Status Register	014 _H	U, SV	BE	System Reset	Page 7-23
PLLCON0	PLL Configuration 0 Register	018 _H	U, SV	SV, SE, P	System Reset	Page 7-25
PLLCON1	PLL Configuration 1 Register	01C _H	U, SV	SV, SE, P	System Reset	Page 7-26
PLLCON2	PLL Configuration 2 Register	020 _H	U, SV	SV, SE, P	System Reset	Page 7-27
PLLERAYSTAT	PLL_ERAY Status Register	024 _H	U, SV	BE	System Reset	Page 7-37
PLLERAYCON0	PLL_ERAY Configuration 0 Register	028 _H	U, SV	SV, SE, P	System Reset	Page 7-39
PLLERAYCON1	PLL_ERAY Configuration 1 Register	02C _H	U, SV	SV, SE, P	System Reset	Page 7-41

Table 7-6 Register Overview of CCU Registers (Offset from SCU Base)

Short Name	Long Name	Offset Addr. 1)	Access Mode		Reset	Description See
			Read	Write		
CCUCON0	CCU Control Register 0	030 _H	U, SV	SV, SE, P	System Reset	Page 7-49
CCUCON1	CCU Control Register 1	034 _H	U, SV	SV, SE, P	System Reset	Page 7-55
FDR	Fractional Divider Register	038 _H	U, SV	SV, SE, P	System Reset	Page 7-86
EXTCON	External Clock Control Register	03C _H	U, SV	SV, SE, P	System Reset	Page 7-84
CCUCON2	CCU Control Register 2	040 _H	U, SV	SV, SE, P	System Reset	Page 7-64
CCUCON3	CCU Control Register 3	044 _H	U, SV	SV, SE, P	System Reset	Page 7-74
CCUCON4	CCU Control Register 4	048 _H	U, SV	SV, SE, P	System Reset	Page 7-76
CCUCON5	CCU Control Register 5	04C _H	U, SV	SV, SE, P	System Reset	Page 7-62
CCUCON6	CCU Control Register 6	080 _H	U, SV	SV, SE, P	System Reset	Page 7-66
CCUCON7	CCU Control Register 7	084 _H	U, SV	SV, SE, P	System Reset	Page 7-67
–	Reserved	244 _H - 3F4 _H	BE	BE	–	–

1) The absolute register address is calculated as follows:
Module Base Address + Offset Address (shown in this column)

7.2 Reset Control Unit (RCU)

The Reset Control Unit (RCU) of the TC26x B-Step contains the following functional sub-blocks:

- Basic Reset Operation (see [Section 7.2.1](#))
- External Reset sources and indications (see [Section 7.2.2](#))
- Software Boot Support (see [Section 7.2.3](#))
- NMI Trap Generation (see [Section 7.2.4](#))
- RCU register overview table (see [Table 7-17](#))

7.2.1 Reset Operation

This section describes the conditions under which the TC26x B-Step will be reset and the reset operation configuration and control.

7.2.1.1 Overview

The following reset request triggers are available:

- Supply monitor (SWD) triggers a power-on reset (cold reset)
- 1.3V EVR monitor triggers a power-on reset (cold reset)
- 3.3V EVR monitor triggers a power-on reset (cold reset)
- Standby EVR (STBYR) monitor triggers a power-on reset (cold reset)
- External active low hardware “power-on” reset request trigger; $\overline{\text{PORST}}$ (can be either a warm reset or to extend a cold reset)
- External System Request reset trigger pins; $\overline{\text{ESR0}}$ and ESR1 (warm reset)
- Safety Management Unit (SMU) alarm reset request trigger, (warm reset)
- Software reset (SW), (warm reset)
- System Timer (STMx) trigger (warm reset)
- Resets via the JTAG interface
- JTAG resets (special reset)
- Software triggered module reset
-
-
-

Note: The JTAG resets are described in the OCDS chapter.

7.2.1.2 Reset Types

The following list describes the different reset types.

- Power-on Reset :
This reset results in initialization of the complete system into a defined state. A Power-on Reset also generates a Debug Reset and a System Reset (and therefore also an Application Reset).
- System Reset :
This reset leads to a initialization into a defined state of the complete system but without a reset of the power subsystem, debug subsystem or reset configuration registers.
A System Reset also generates an Application Reset.
- Debug Reset :
This reset leads to a initialization into a defined state of the complete debug system.
- Application Reset:
This reset leads to a initialization into a defined state of the complete application system with the following parts: all peripherals, the CPUs and parts of the SCU.

- **Module Resets:**
Module resets result in individual modules being initialized into a defined state without any impact on the rest of the system.

7.2.1.3 Reset Sources Overview

The connection of the reset sources and the activated reset signals/domains are shown in [Table 7-7](#).

Table 7-7 Effect of Reset Triggers

Reset Request Trigger	Application Reset	Debug Reset	System Reset
PORST	Activated	Activated	Activated
STBYR	Activated	Activated	Activated
SWD	Activated	Activated	Activated
EVR13	Activated	Activated	Activated
EVR33	Activated	Activated	Activated
ESR0	Configurable	Not Activated	Configurable
ESR1	Configurable	Not Activated	Configurable
SMU	Configurable	Not Activated	Configurable
STM0	Configurable	Not Activated	Configurable
STM1	Configurable	Not Activated	Configurable
SW	Configurable	Not Activated	Configurable
Cerberus RSTCL0¹⁾	Activated	Not Activated	Activated
Cerberus RSTCL1²⁾	Not Activated	Activated	Not Activated
Cerberus RSTCL3³⁾	Activated	Not Activated	Not Activated
Module Resets⁴⁾	Not Activated	Not Activated	Not Activated

1) Cerberus resets may be triggered by CBS_OCCTRL or CBS_OJCONF

2) Cerberus resets may be triggered by CBS_OCCTRL or CBS_OJCONF

3) Cerberus resets may be triggered by CBS_OCCTRL or CBS_OJCONF

4) Module Resets (via MOD_KRSTx.RST register bits in some modules) have no effect on chip-level reset system. The scope of a module reset is limited to the module itself.

7.2.1.4 Warm and Cold Resets

A cold “power-on” reset is a reset which is triggered for the first time during a system power-up or in response to a temporary power failure. During the power-up the EVR primary undervoltage monitors will trigger a complete reset of the system, placing the

system into a known state. The $\overline{\text{PORST}}$ pin can be used to extend this reset phase and control the timing of its release. The pins and internal states are placed immediately into their reset state when the trigger is asserted.

A warm reset is a reset which is triggered while the system is already operational and the supplies remain stable. It is used to return the system to the same known state. On a warm reset request, any reset of pin states will take place immediately, but the internal circuitry will only be reset after the system has been brought into a state where memory contents and debug trace data will not be corrupted and the current consumption has been ramped down. Note that $\overline{\text{PORST}}$ may be asserted as a warm reset.

7.2.1.5 EVR Resets and PORST

The $\overline{\text{PORST}}$ pin is a bidirectional reset in/output intended for external triggering of power-related resets. If this pin is left open then the default behavior shall be that the device remains in a reset state. If the $\overline{\text{PORST}}$ pin remains asserted after a power event then the reset will be extended until it is deasserted. This does not replace the ESR pins functional reset.

7.2.1.6 Module Reset Behavior

Table 7-8 lists how the various functions of the TC26x B-Step are affected by each reset type. An “X” means that this block has at least some register/bits that are affected by the corresponding reset.

Table 7-8 Effect of Reset on Device Functions

Module / Function	Application Reset	Debug Reset	System Reset	Warm “Power-on” Reset	Cold Power-on Reset
All TriCore CPU Cores ¹⁾	X	X	X	X	X
Peripherals (except SCU)	X	X	X	X	X
SCU	X	Not affected	X	X	X
Flash Memory	Not affected	Not affected, reliable	X	X	X
JTAG Interface	Not affected	Not affected	Not affected	X	X

Table 7-8 Effect of Reset on Device Functions (cont'd)

Module / Function	Application Reset	Debug Reset	System Reset	Warm "Power-on" Reset	Cold Power-on Reset
OCDS	Not affected	X	Not affected	X	X
MCDS	Not affected	X	Not affected	X	X
Backup Clock	Not affected	Not affected	Not affected	Not affected	Not affected
XTAL Oscillator, PLLs	Not affected	Not affected	X	X	X
Port Pins	X	Not affected	X	X	X
Pins ESRx	Not affected	Not affected	X	X	X
EVR	Not affected	Not affected	Not affected	Not affected	X
Standby Controller	Not affected	Not affected	Not affected	X	X

On-chip Static RAMs²⁾

CAN	Initialized	Initialized	Initialized	Initialized	Uninitialized
All DSPRs³⁾	Not affected, reliable	Not affected, reliable	Not affected, reliable	Not affected, reliable	Uninitialized
All PSPRs⁴⁾	Not affected, reliable	Not affected, reliable	Not affected, reliable	Not affected, reliable	Uninitialized
All PCACHES and DCACHES⁵⁾	Cache invalidated	Cache invalidated	Cache invalidated	Cache invalidated	Uninitialized
LMU⁶⁾	Not affected, reliable	Not affected, reliable	Not affected, reliable	Not affected, reliable	Uninitialized
Other	Not affected, reliable	Not affected, reliable	Not affected, reliable	Not affected, reliable	Uninitialized

- 1) Lockstepped checker cores are initialized to the same reset state as the main core to avoid comparator fails during start-up
- 2) Reliable here means that also the redundancy is not affected by the reset.
- 3) DSPR is partially used as a scratchpad by the startup firmware. Previous data stored in the upper 32kB will be overwritten on start-up. Auto-initialization on reset can be optionally configured by PROCOND.RAMIN and PROCOND.RAMINSEL. Auto-initialization is disabled for CPU0 DSPR if the standby RAM is enabled.

- 4) Depending upon PROCOND.RAMIN and RAMINSEL setting, these memories may be automatically erased on cold or warm resets.
- 5) Tag memories are invalidated by hardware at reset. Depending upon PROCOND.RAMIN setting, cache content may also be automatically erased at reset (See MTU chapter for details).
- 6) Depending upon PROCOND.RAMIN and RAMINSEL setting, these memories may be automatically erased on cold or warm resets.

7.2.1.7 General Reset Operation

A reset is generated if an enabled reset request trigger is asserted. Most reset triggers can be configured to initiate different types of reset. No action (disabled) is a valid configuration which is selected by setting the corresponding bit field in the Reset Configuration Register to 00_B.

A Debug Reset can only be requested by dedicated reset request triggers and can not be selected via a Reset Configuration Register. For more information see also register RSTCON.

7.2.1.8 Reset Generation

The figure below shows the RCU controlling the reset generation for the complete device, with the exception of the JTAG reset domain.

Note: The JTAG reset domain is controlled by the \overline{TRST} pin.

The RCU detects the different reset requests, schedules a shutdown of the system, and then generates the appropriate internal reset pulse(s).

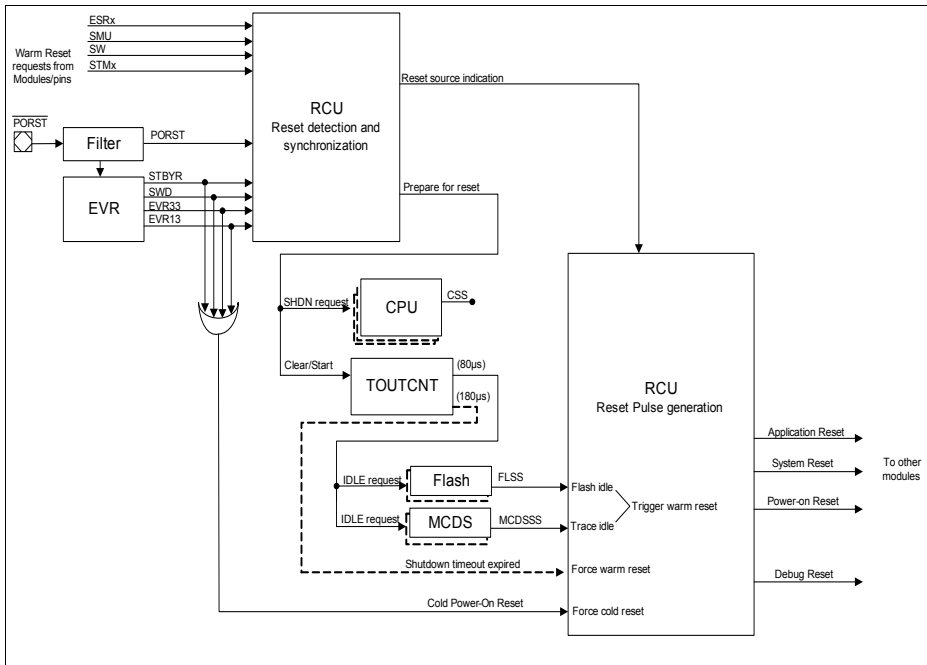


Figure 7-19 Reset Overview

7.2.1.9 Shutdown and Reset Delay Timeout Counter (TOUTCNT)

The system is automatically brought to a stable state before the internal reset(s) are applied so that the RAM content may be reliably reused after a warm reset

On detection of a warm Application Reset, System Reset or Power-On Reset trigger, a Shutdown Trap request is sent by the RCU to the CPU(s). This request causes the CPU(s) which are not already in a halt or idle state to unconditionally execute a ROM shutdown routine which executes a controlled rampdown of the device power consumption (to prevent current jumps which might trigger EVR reset). At the end of this routine each CPU executes a WAIT instruction and stalls, to ensure that all ongoing write transactions have been completed. Interrupts and NMI to the CPUs during the shutdown routine will not reawaken CPUs.

The internal reset starts when all relevant system modules are Idle. If timeout counter TOUTCNT exceeds a timeout period (180 microseconds) before the system become idle then the reset is started regardless. This timeout ensures that a reset would still occur in the case of an internal deadlock.

The RSTCON2.CSS bits indicate whether each CPU successfully flushed its write buffers and reached an idle state before the previous reset. These bits can therefore be used to determine whether RAM content integrity can be trusted after the previous reset cycle.

7.2.1.10 Reset Triggers

There are two types of reset triggers for the reset control logic:

- Triggers that lead to a specific reset
- Triggers that lead to a configurable reset

Specific Reset Triggers

Assertion of these triggers leads to a predefined reset type. These triggers can not be enabled / disabled. All specific reset triggers are listed in [Table 7-9](#).

Table 7-9 Specific Reset Triggers

Reset Trigger	Reset Type Request
Cerberus 0 (CB0)	SystemReset
Cerberus 1 (CB1)	DebugReset
Cerberus 3 (CB3)	ApplicationReset
STBYR	Power-on Reset
SWD	Power-on Reset
EVR13	Power-on Reset
EVR33	Power-on Reset

Configurable Reset Triggers

Assertion of these triggers leads to a configurable reset type. These triggers can be enabled / disabled. The result of the reset triggers is defined by the corresponding bit field in register RSTCON.

Prevention of Double SMU Resets

After boot firmware initialization, the default behaviour of the SMU/RCU is that a Watchdog timeout would result in an NMI followed by an Application Reset. If an SMU-induced Application Reset occurs twice, a severe system malfunction is assumed and the TC26x B-Step is held in permanent Application Reset until a Power-On or System Reset occurs. This prevents the device from being periodically reset if the application fails to service the watchdog correctly and the watchdog repeatedly times out.

An internal flag is set when the first SMU reset is requested. If a second reset is also requested by the SMU when the internal flag is already set, the double SMU reset event has occurred and a permanent reset request is automatically generated. This internal flag is only reset by a System Reset or when bit WDTSCON1.CLRIRF is set and bit WDTSCON0.ENDINIT has also been set. A correct service of the WDT does not automatically clear this flag.

If this behaviour is undesirable, then the CLRIRF bit should be written by the application during initialization (before the watchdog times out), to clear the flag and prevent any subsequent occurrence of a permanent reset.

Note: If for any reason random code is executed bit field RSTCON.SMU can be updated unintentionally. This can result in an SMU alarm not leading to a reset. To avoid this after the SSW is finished this bit field should be checked and the Safety WDT ENDINIT protection enabled.

7.2.1.11 Debug Reset Specific Behavior

For safety reasons it is required by the debugger that if the OCDS system is disabled a DebugReset is also asserted every time an Application Reset is asserted.

7.2.1.12 Module Resets

Many modules within TC26x B-Step can be reset individually. The module reset has no effect outside the module itself. A module reset can only be triggered by writing '1' into both of the module reset registers MOD_KRST1.RST and register MOD_KRST0.RST.

The Module Reset register bits may only be written by those masters which have been explicitly configured to have module access (See module ACCEN register). In addition, it is only possible to write to these reset bits during the short time window after a correct ENDINIT password unlock sequence (See Watchdog chapter for details). This prevents accidental module resets by rogue software.

The table below shows the modules which have Module Reset capability

Table 7-10 Module Resets

Module	Module Reset capability
CAN	Module reset implemented in CAN
ASCLIN	Module reset implemented in ASCLIN
GTM	Module reset implemented in GTM
Flexray	Module reset implemented in ERAY
QSPI	Module reset implemented in QSPI
HSSL	Module reset implemented in HSSL

Table 7-10 Module Resets

Module	Module Reset capability
ETHERMAC	Module reset implemented in ETHERMAC
MSC	Module reset implemented in MSC
SENT	Module reset implemented in SENT
VADC	Module reset implemented in VADC
SDADC	Module reset implemented in SDADC
SCR	Module reset implemented in PMC (PMSWCR1.SCRSTREQ)
CCU6	Module reset implemented in CCU6
GPT12	Module reset implemented in GPT12
CPUs	No individual CPU reset capability
SMU	Module reset implemented in SMU
STM	Module reset implemented in STM
PSI5	Module reset implemented in PSI5
PSI5-S	Module reset implemented in PSI5-S
DMA	Per channel reset implemented in DMA
SCU, RCU, PMC, CCU	No module reset capability
PMU, Flash	No module reset capability
LMU	No module reset capability
PORTS	No module reset capability
IR	No module reset capability
IOM	Module reset implemented in IOM
I2C	Module reset implemented in I2C

7.2.1.13 Reset Controller Registers

Status Registers

After a chip level reset has been executed, the Reset Status registers provide information on the trigger of the last reset(s). Warm reset status bits are updated upon each reset cycle. A reset cycle is finished when all resets are de-asserted. Within a reset cycle the status flags are used as sticky flags. Module level resets have no effect on the Reset Status register. Bits which may indicate a cold reset trigger (STBYR, SWD, EVR33, EVR13, PORST and EVR13) provide useful information to the application and are not cleared automatically. The application may clear these bits by writing to bit RSTCON2.CLRC.

RSTSTAT

Reset Status Register

(050_H)

Reset Value: 1381 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		STB YR	0	SWD	EVR 33	EVR 13	0			CB3	CB1	CB0	0	POR ST	
r		rh	r	rh	rh	rh	r			rh	rh	rh	r	rh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					STM 2	STM 1	STM 0	SW	SMU	0	ESR 1	ESR 0			
r					rh	rh	rh	rh	rh	r	rh	rh			

Field	Bits	Type	Description
ESR0	0	rh	Reset Request Trigger Reset Status for ESR0 0 _B The last reset was not requested by this reset trigger 1 _B The last reset was requested by this reset trigger
ESR1	1	rh	Reset Request Trigger Reset Status for ESR1 0 _B The last reset was not requested by this reset trigger 1 _B The last reset was requested by this reset trigger

Field	Bits	Type	Description
SMU	3	rh	Reset Request Trigger Reset Status for SMU (See SMU section for SMU trigger sources, including Watchdog Timers) 0 _B The last reset was not requested by this reset trigger 1 _B The last reset was requested by this reset trigger
SW	4	rh	Reset Request Trigger Reset Status for SW 0 _B The last reset was not requested by this reset trigger 1 _B The last reset was requested by this reset trigger
STM0	5	rh	Reset Request Trigger Reset Status for STM0 Compare Match 0 _B The last reset was not requested by this reset trigger 1 _B The last reset was requested by this reset trigger
STM1	6	rh	Reset Request Trigger Reset Status for STM1 Compare Match (If Product has STM1) 0 _B The last reset was not requested by this reset trigger 1 _B The last reset was requested by this reset trigger
STM2	7	rh	Reset Request Trigger Reset Status for STM2 Compare Match (If Product has STM2) 0 _B The last reset was not requested by this reset trigger 1 _B The last reset was requested by this reset trigger
PORST	16	rh	Reset Request Trigger Reset Status for PORST 0 _B This reset trigger has not occurred since the last clear (by RSTCON2.CLRC) 1 _B This reset trigger has occurred since the last clear (by RSTCON2.CLRC) This bit is also set if the bits CB0, CB1, and CB3 are set in parallel.
0	17	r	Reserved Read as 0; should be written with 0.

Field	Bits	Type	Description
CB0	18	rh	Reset Request Trigger Reset Status for Cerberus System Reset 0_B The last reset was not requested by this reset trigger 1_B The last reset was requested by this reset trigger
CB1	19	rh	Reset Request Trigger Reset Status for Cerberus Debug Reset 0_B The last reset was not requested by this reset trigger 1_B The last reset was requested by this reset trigger
CB3	20	rh	Reset Request Trigger Reset Status for Cerberus Application Reset 0_B The last reset was not requested by this reset trigger 1_B The last reset was requested by this reset trigger
EVR13	23	rh	Reset Request Trigger Reset Status for EVR13 0_B This reset trigger has not occurred since the last clear (by RSTCON2.CLRC) 1_B This reset trigger has occurred since the last clear (by RSTCON2.CLRC)
EVR33	24	rh	Reset Request Trigger Reset Status for EVR33 0_B This reset trigger has not occurred since the last clear (by RSTCON2.CLRC) 1_B This reset trigger has occurred since the last clear (by RSTCON2.CLRC)
SWD	25	rh	Reset Request Trigger Reset Status for Supply Watchdog (SWD) The Supply Watchdog trigger is described in Power Management Controller "Supply Monitoring" chapter 0_B This reset trigger has not occurred since the last clear (by RSTCON2.CLRC) 1_B This reset trigger has occurred since the last clear (by RSTCON2.CLRC)

Field	Bits	Type	Description
STBYR	28	rh	Reset Request Trigger Reset Status for Standby Regulator Watchdog (STBYR) 0 _B This reset trigger has not occurred since the last clear (by RSTCON2.CLRC) 1 _B This reset trigger has occurred since the last clear (by RSTCON2.CLRC)
0	[15:8], [31:29], [27:26], [21,22], 2	r	Reserved Read as 0; should be written with 0.

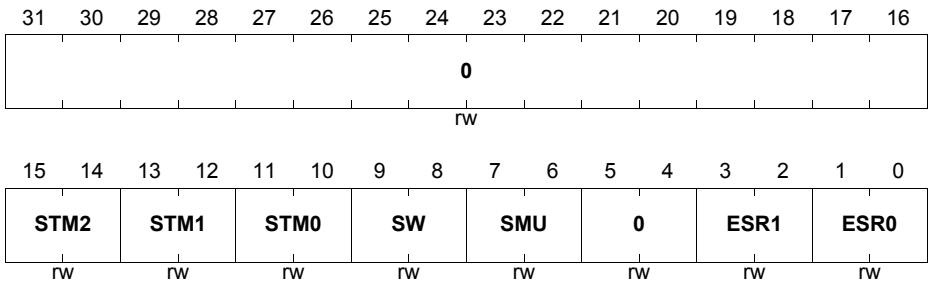
Reset Configuration Registers

RSTCON

Reset Configuration Register

(058_H)

Reset Value: 0000 0282_H



Field	Bits	Type	Description
ESR0	[1:0]	rw	ESR0 Reset Request Trigger Reset Configuration This bit field defines which reset is generated by a reset request trigger from ESR0 reset. 00 _B No reset is generated for a trigger of ESR0 01 _B A System Reset is generated for a trigger of ESR0 reset 10 _B An Application Reset is generated for a trigger of ESR0 reset 11 _B Reserved, do not use this combination
ESR1	[3:2]	rw	ESR1 Reset Request Trigger Reset Configuration This bit field defines which reset is generated by a reset request trigger from ESR1 reset. 00 _B No reset is generated for a trigger of ESR1 01 _B A System Reset is generated for a trigger of ESR1 reset 10 _B An Application Reset is generated for a trigger of ESR1 reset 11 _B Reserved, do not use this combination
0	[5:4]	rw	Reserved Should be written with 0.
SMU	[7:6]	rw	SMU Reset Request Trigger Reset Configuration This bit field defines which reset is generated by a reset request trigger from SMU reset. 00 _B No reset is generated for a trigger of SMU 01 _B A System Reset is generated for a trigger of SMU reset 10 _B An Application Reset is generated for a trigger of SMU reset 11 _B Reserved, do not use this combination

Field	Bits	Type	Description
SW	[9:8]	rw	<p>SW Reset Request Trigger Reset Configuration This bit field defines which reset is generated by a reset request trigger from software reset.</p> <p>00_B No reset is generated for a trigger of software reset</p> <p>01_B A System Reset is generated for a trigger of Software reset</p> <p>10_B An Application Reset is generated for a trigger of Software reset</p> <p>11_B Reserved, do not use this combination</p>
STM0	[11:10]	rw	<p>STM0 Reset Request Trigger Reset Configuration This bit field defines which reset is generated by a reset request trigger from STM0 compare match reset.</p> <p>00_B No reset is generated for an STM0 trigger</p> <p>01_B A System Reset is generated for a trigger of STM0 reset</p> <p>10_B An Application Reset is generated for a trigger of STM0 reset</p> <p>11_B Reserved, do not use this combination</p>
STM1	[13:12]	rw	<p>STM1 Reset Request Trigger Reset Configuration (If Product has STM1) This bit field defines which reset is generated by a reset request trigger from STM1 compare match reset.</p> <p>00_B No reset is generated for a trigger of STM1</p> <p>01_B A System Reset is generated for a trigger of STM1 reset</p> <p>10_B An Application Reset is generated for a trigger of STM1 reset</p> <p>11_B Reserved, do not use this combination</p>

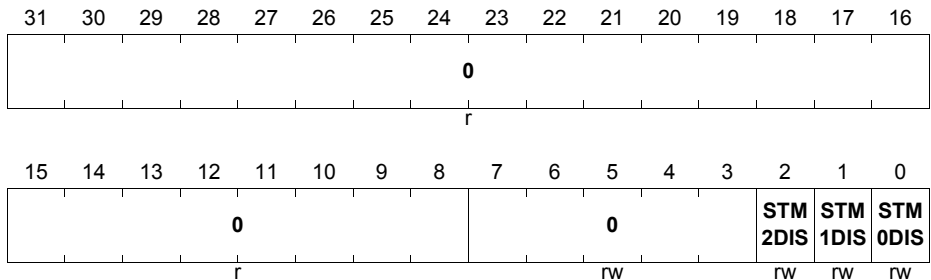
Field	Bits	Type	Description
STM2	[15:14]	rw	STM2 Reset Request Trigger Reset Configuration (If Product has STM2) This bit field defines which reset is generated by a reset request trigger from STM2 compare match reset. 00 _B No reset is generated for a trigger of STM2 01 _B A System Reset is generated for a trigger of STM2 reset 10 _B An Application Reset is generated for a trigger of STM2 reset 11 _B Reserved, do not use this combination
0	[31:16]	rw	Reserved Should be written with 0.

ARSTDIS

Application Reset Disable Register

(05C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
STM0DIS	0	rw	STM0 Disable Reset This bit field defines if an Application Reset leads to an reset for the STM0. 0 _B An Application Reset resets the STM0 1 _B An Application Reset has no effect for the STM0

Field	Bits	Type	Description
STM1DIS	1	rw	STM1 Disable Reset (If Product has STM1) This bit field defines if an Application Reset leads to an reset for the STM1. 0 _B An Application Reset resets the STM1 1 _B An Application Reset has no effect for the STM1
STM2DIS	2	rw	STM2 Disable Reset (If Product has STM2) This bit field defines if an Application Reset leads to an reset for the STM2. 0 _B An Application Reset resets the STM2 1 _B An Application Reset has no effect for the STM2
0	[31:8]	r	Reserved Read as 0; should be written with 0.
0	[7:3]	rw	Reserved Read as 0; should be written with 0.

SW Reset Configuration Register

This register controls the SW Reset operation.

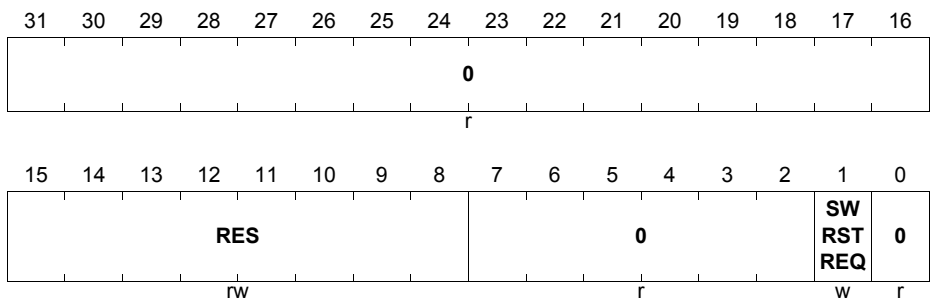
As for other kernel registers, write access to the SWRSTCON register is configurable via SCU_ACCEN0 and additionally is temporally restricted by Safe ENDINIT. These restrictions can be used to provide protection against accidental reset by unauthorised CPUs or system bus masters.

SWRSTCON

Software Reset Configuration Register

(060_H)

Reset Value: 0000 XX00_H



Field	Bits	Type	Description
0	0	rw	Reserved Read as 0; should be written with 0.
SWRSTREQ	1	w	Software Reset Request 0 _B No SW Reset is requested 1 _B A SW Reset request trigger is generated This bit is automatically cleared and read always as zero.
RES	[15:8]	rw	Reserved Should be written with original content.
0	[7:2], [31:16]	r	Reserved Read as 0; should be written with 0.

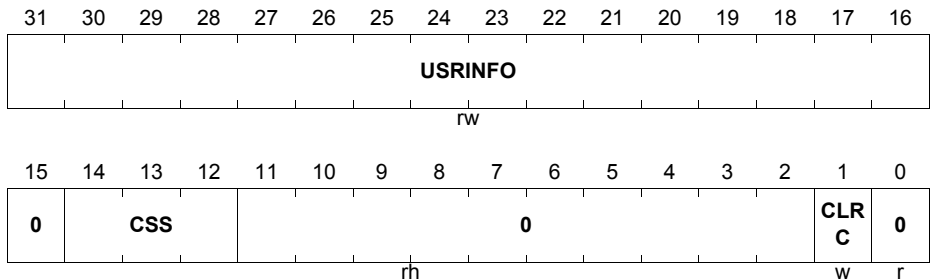
Additional Reset Control Register

RSTCON2

Additional Reset Control Register

(064_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
USRINFO	[31:16]	rw	User Information User data register (Cleared only on Power-on reset).
Reserved	[11:2], 15	r	Reserved Internal use only. Bits may read as 0 or 1. Writes have no effect.
CSS2	14	r	Reserved in this product Bit reads as '1' after warm reset

Field	Bits	Type	Description
CSS1	13	rh	<p>CPU1 Safe State Reached</p> <p>The state of CPU1 before the last warm reset If any bit is zero after an Application Reset (or higher) then it is possible that SRAM content could have been corrupted by the reset</p> <p>0_B CPU1 safe state not achieved prior to last reset 1_B CPU1 in safe state at last reset</p>
CSS0	12	rh	<p>CPU0 Safe State Reached</p> <p>The state of CPU0 before the last warm reset If any bit is zero after an Application Reset (or higher) then it is possible that SRAM content could have been corrupted by the reset</p> <p>0_B CPU0 safe state not achieved prior to last reset 1_B CPU0 in safe state at last reset</p>
CLRC	1	w	<p>Clear Cold Reset Status</p> <p>This bit simultaneously clears the sticky status bits which may indicate any previous cold reset (i.e. RSTSTAT.STBYR, RSTSTAT.SWD, RSTSTAT.EVR33, RSTSTAT.EVR13 and RSTSTAT.PORST).</p> <p>0_B No effect 1_B Clear cold reset RSTSTAT status bits</p>
0	0	r	<p>Reserved</p> <p>Bits read as 0. Should only be written as 0.</p>

7.2.2 External Reset Sources and Indications

ESR interface pins are provided to give an external indication of internal resets, and to provide a mechanism for external sources to trigger internal resets:

ESR pin functions:

- Reset request trigger
- Wakeup trigger
- Reset indication output
- Trap request trigger

7.2.2.1 External Service Requests ($\overline{\text{ESRx}}$)

The ESR pins can be used in various ways:

- ESR inputs can trigger a reset
- ESR outputs can provide a reset indication
- ESR inputs can trigger a trap (NMI)
- ESR pins can be used as general data I/O pins
- ESR inputs can trigger a wake-up from standby mode

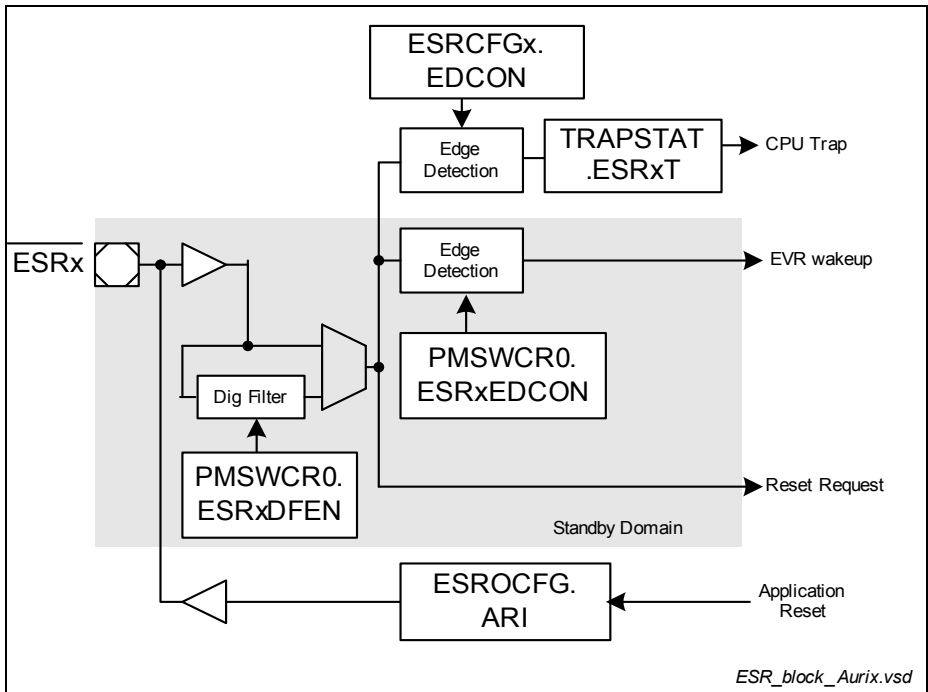


Figure 7-20 ESR Operation

ESRx as Reset Request Trigger

An $\overline{\text{ESR0}}/\overline{\text{ESR1}}$ pin reset request trigger can lead to a Systemor Application Reset. The type of the reset is configured via RSTCON.ESRx.

The input signals $\overline{\text{ESR0}}/\overline{\text{ESR1}}$ can be filtered. The filter can be disabled by register bit PMSWCR0.ESRxDFEN.

If the digital filter is enabled then pulses less than 20ns cannot trigger a reset and pulses longer than 100ns will always result in a trigger..

The behavior of $\overline{\text{ESR0}}/\overline{\text{ESR1}}$ pins can be configured by programming registers ESRCFG0/1. The pad control functionality can be configured independently for each pin. The configuration comprises:

- Selection of driver type (open-drain or push-pull)
- Enable of the output driver (input and/or output capability)
- Enable of internal pull-up or pull-down resistance

ESRx as Reset Output

The external pins $\overline{\text{ESR0}}/\overline{\text{ESR1}}$ can provide a reset output indication (open drain) for Application Resets .

Register ESROCFG.ARI determines the output of the ESR pin(s) when one or more are configured as functional reset outputs. ARI is set automatically when any Application Reset starts and cleared by Boot or Application software (Note that an Application Reset also occurs on a System Reset or Power-On Reset). While the ARI bit is set, the configured ESR pin(s) drive active low. A subsequent write to ESROCFG.ARC clears the ARI bit, which deasserts the ESR output(s). The exact duration of the ESR pulse is determined by the value of the Flash Config Sector setting “ESR0CNT” which is copied by Boot Code into FLASH0_PROCOND.ESR0CNT on a cold power-on reset. The effect of this value is shown in [Table 7-11](#). When the same ESR pin is configured as a PORT output, its state can instead be controlled directly by application software so that the application has the possibility to reset external devices.

Do not configure ESR pin to be used as both an output (reset indication or general purpose port) and a reset trigger input simultaneously.

Table 7-11 ESRx Reset Indication Options

ESR0CNT value	ESR Reset Indication Behaviour
0000 0000h	ESR0 deasserted as soon as possible after start of Boot Code execution
> 0000 0000h and < FFFh	ESR0 deasserted after programmed delay (by Boot Code). Boot Code may be extended and Application start may be delayed if programmed delay exceeds the normal Boot Code execution time. Programmed delay is ESR0CNT * 10 microseconds
FFFh	ESR0 not deasserted by Boot Code. Application code must deassert ESROCFG.ARI bit or PORT output to deassert ESRx pin(s).

An external device may extend an existing reset condition by holding the ESR pin active.

ESR Registers

ESRCFG0

ESR0 Input Configuration Register

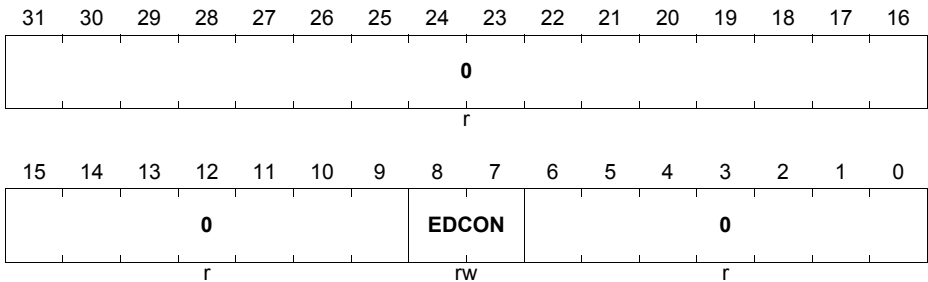
 (070_H)

 Reset Value: 0000 0100_H

ESRCFG1

ESR1 Input Configuration Register

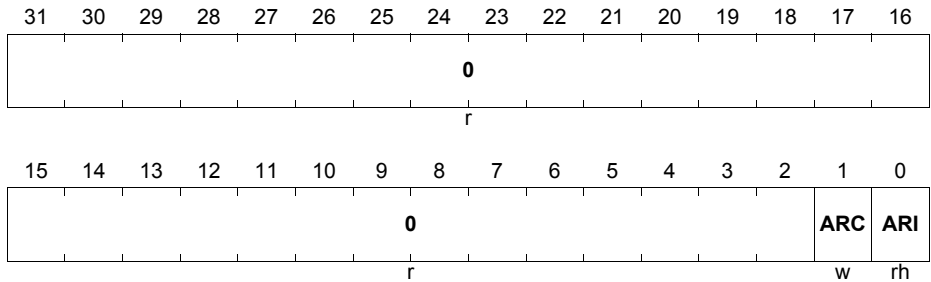
 (074_H)

 Reset Value: 0000 0100_H


Field	Bits	Type	Description
0	[6:0]	r	Reserved Read as 0; should be written with 0.
EDCON	[8:7]	rw	Edge Detection Control This bit field defines the edges that lead to an ESRx trigger of the synchronous path. 00 _B No trigger is generated 01 _B A trigger is generated upon a rising edge 10 _B A trigger is generated upon a falling edge 11 _B A trigger is generated upon a rising OR falling edge
0	[31:9]	r	Reserved Read as 0; should be written with 0.

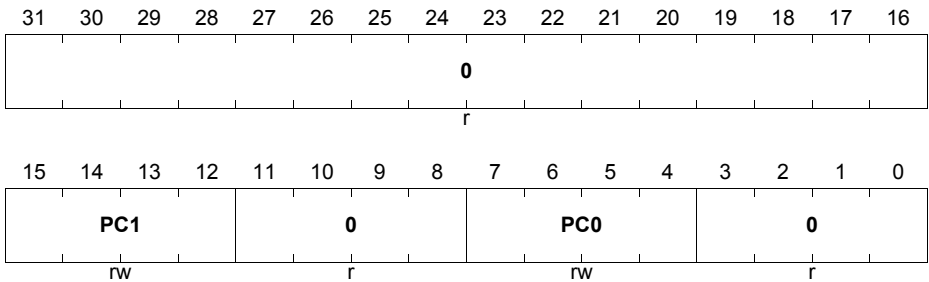
ESROCFG
ESR Output Configuration Register

 (078_H)

 Reset Value: 0000 000X_H


Field	Bits	Type	Description
ARI	0	rh	<p>Application Reset Indicator</p> <p>This bit is set when an Application Reset request trigger occurs and cleared by writing to ARC.</p> <p>0_B No application reset trigger detected (since last clear)</p> <p>1_B Application reset trigger detected (since last clear)</p> <p>When the ARI bit is set and an ESR pin is configured as a reset output, the corresponding ESR input will not re-trigger a reset. This prevents feedback of the reset indication causing a new reset request. Extension of the reset by an external ESR source is handled by SSW.</p> <p><i>Note: Observed reset value after boot will depend upon ARI mode, see Table 7-11</i></p>
ARC	1	w	<p>Application Reset Indicator Clear</p> <p>0_B No effect</p> <p>1_B Clear Application Reset Indicator (ARI)</p> <p>Read as 0</p>
0	[31:2]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

The input/output control registers select the digital output and input driver functionality and characteristics of the pin. Direction (input or output), pull-up or pull-down devices for inputs, and push-pull or open-drain functionality for outputs can be selected by the corresponding bit fields PCx (x = 0-1).

IOCR
Input/Output Control Register (0A0_H) **Reset Value: 0000 20E0_H**


Field	Bits	Type	Description
PC0, PC1	[7:4], [15:12]	rw	Control for ESR Pin x This bit field defines the ESR x functionality according to the coding tables (see Table 7-12 and Table 7-13).
0	[3:0], [11:8], [31:16]	r	Reserved Read as 0; should be written with 0.

Pad Control Coding

Table 7-12 describes the coding of the PC0 bit field that determine the port line functionality.

Table 7-12 PC0 Coding

PC0[3:0]	I/O	Output Characteristics	Selected Pull-up/Pull-down/ Selected Output Function
0X00 _B	Input is active and not inverted; Output is inactive		No input pull device connected
0X01 _B			Input pull-down device connected
0X10 _B			Input pull-up device connected
0X11 _B			No input pull device connected
1000 _B	Input is active and not inverted; Output is active	Push-pull	General-purpose Output
1001 _B			Output drives a 0 for System Resets until ESROCFG.ARI is cleared, a weak pull-up is active otherwise
1010 _B			Reserved, do not use this combination
1011 _B			Reserved, do not use this combination
1100 _B	the input is active and not inverted; Output is active	Open-drain	General-purpose Output
1101 _B			Output drives a 0 for System Resets until ESROCFG.ARI is cleared, a weak pull-up is active otherwise
1110 _B			Output drives a 0 for Application Resets until ESROCFG.ARI is cleared, a weak pull-up is active otherwise
1111 _B			Reserved, do not use this combination

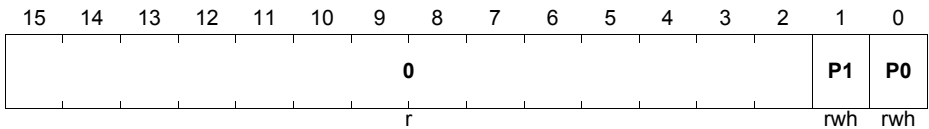
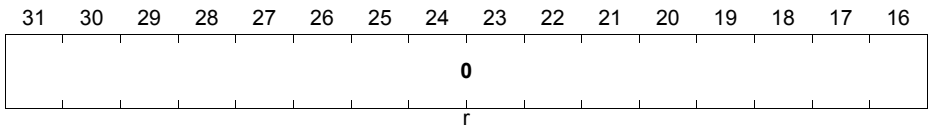
Pad Control Coding

Table 7-13 describes the coding of the PC1 bit field that determine the port line functionality.

Table 7-13 PC1 Coding

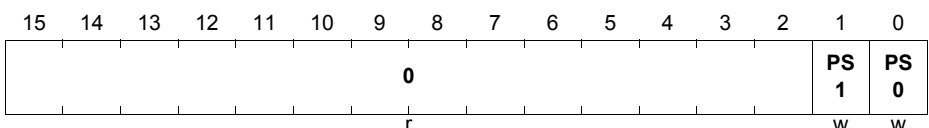
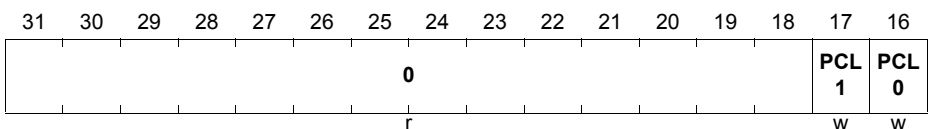
PC1[3:0]	I/O	Output Characteristics	Selected Pull-up/Pull-down/ Selected Output Function
0X00 _B	Input is active and not inverted; Output is inactive		No input pull device connected
0X01 _B			Input pull-down device connected
0X10 _B			Input pull-up device connected
0X11 _B			No input pull device connected
1000 _B	Input is active and not inverted; Output is active	Push-pull	General-purpose Output
1001 _B			Reserved, do not use this combination
1010 _B			Reserved, do not use this combination
1011 _B			Reserved, do not use this combination
1100 _B	the input is active and not inverted; Output is active	Open-drain	General-purpose Output
1101 _B			Reserved, do not use this combination
1110 _B			Output drives a 0 for Application Resets until ESROCFG.ARI is cleared, a 'Z' otherwise
1111 _B			Reserved, do not use this combination

The output register determines the value of a GPIO pin when it is selected by IOCR as output. Writing a 0 to a OUT.Px (x = 0-1) bit position delivers a low level at the corresponding output pin. A high level is output when the corresponding bit is written with a 1. Note that each single bit or group of bits of OUT.Px can be set/cleared by writing appropriate values into the output modification register OMR.

OUT
ESR Output Register
(0A4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
Px (x = 0-1)	x	rwh	Output Bit x This bit determines the level at the output pin $\overline{\text{ESRx}}$ if the output is selected as GPIO output. 0 _B The output level of $\overline{\text{ESRx}}$ is 0 1 _B The output level of $\overline{\text{ESRx}}$ is 1 Px can also be set/cleared by control bits of the OMR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

The output modification register contains control bits that make it possible to individually set, clear, or toggle the logic state of a single pad by manipulating the output register.

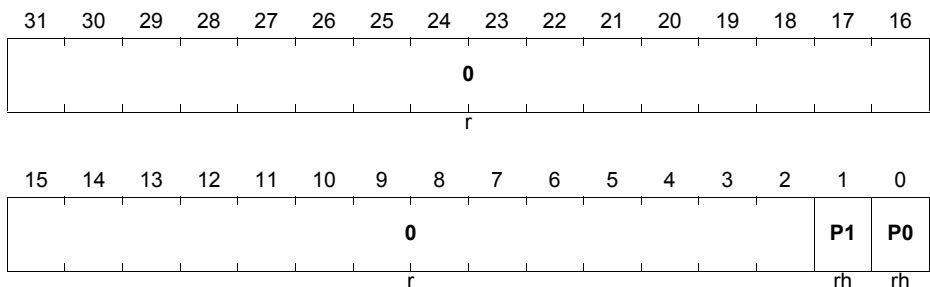
OMR
ESR Output Modification Register
(0A8_H)
Reset Value: 0000 0000_H


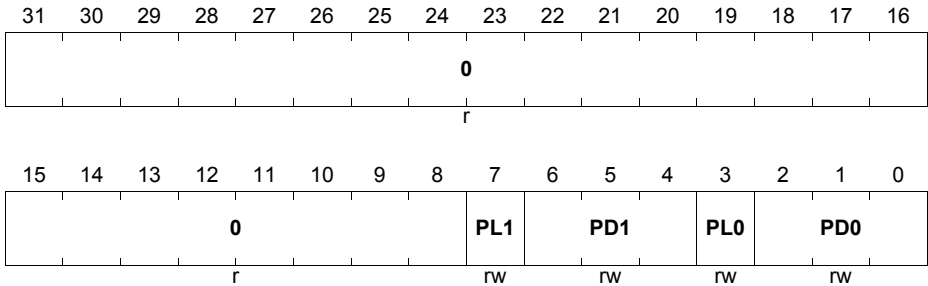
Field	Bits	Type	Description
PSx (x = 0-1)	x	w	ESRx Pin Set Bit x Setting this bit will set or toggle the corresponding bit in the output register OUT. The function of this bit is shown in Table 7-14 . Reading this bit returns 0.
PCLx (x = 0-1)	x + 16	w	ESRx Pin Clear Bit x Setting this bit will clear or toggle the corresponding bit in the port output register OUT. The function of this bit is shown in Table 7-14 . Reading this bit returns 0.
0	[15:2], [31:18]	r	Reserved Read as 0; should be written with 0.

Table 7-14 Function of the Bits PCLx and PSx

PCLx	PSx	Function
0	0	Bit OUT.Px is not changed
0	1	Bit OUT.Px is set
1	0	Bit OUT.Px is cleared
1	1	Bit OUT.Px is toggled

The logic level of a GPIO pin can be read via the read-only port input register IN. Reading the IN register always returns the current logical value at the GPIO pin independently whether the pin is selected as input or output.

IN
ESR Input Register
(0AC_H)
Reset Value: 0000 000X_H


PDR
ESR Pad Driver Mode Register
(9C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
PD0, PD1	[2:0], [6:4]	rw	Pad Driver Mode for ESR Pins 0 and 1 PD0 selects the driver mode for ESR0 output. PD1 selects the driver mode for ESR1 output. X00 _B Speed Grade 1 X01 _B Speed Grade 2 X10 _B Speed Grade 3 X11 _B Speed Grade 4
PL0, PL1	3, 7	rw	Pad Level Selection for ESR Pins 0 and 1 PL0 selects the pad level for ESR0. PL1 selects the pad level for ESR1. 0 _B Automotive Level 1 _B TTL Level
0	[31:8]	rw	Reserved

7.2.3 Boot Software Interface

In order to determine the correct starting point of operation for the software a minimum amount of hardware support is required. As much as possible is done via software. Some decisions have to be made in hardware because they must be known before any software is operational.

For a startup operation there are two general cases that have to be handled:

- Differentiation between Test Mode and Normal Mode for each Power-on Reset event (see [Section 7.2.3.1](#))
- Configuration of the boot option for each Application Reset event (see [Section 7.2.3.2](#))

7.2.3.1 Configuration done with Start-up

At device power-on some basic operating mode selection has to be done. The first decision that has to be made is whether the device should operate in Test Mode or in Normal (Customer) Mode. The Test Mode is only for Infineon internal device testing, it is not intended for any customer and is not related to debug.

If the Normal Mode was selected the next decision is which debug interface type issued for debugging for this session (until the next power-on event).

Table 7-15 Normal Mode / Test Mode Input Selection

Field	Description
TESTMODE	Latched TESTMODE Signal
	0 A Test Mode can be selected
	1 Normal Mode is selected
TRST	Latched TRST Signal
	0 The JTAG interface is active.
	1 The DAP interface is active.

After these two decisions were made the detailed decision has to be made to define the real startup configuration. Most is made via the software and can be supported by some hardware selections depending on the startup configuration that should be selected.

7.2.3.2 Start-up Configuration Options

The states of the HWCFG port pin inputs are latched on the rising edge of an Application Reset and stored in register STSTAT.HWCFG. The update of bit field STSTAT.HWCFG with the latched value is only done if bit STSTAT.LUDIS is cleared. If bit STSTAT.LUDIS is set then the value of STSTAT.HWCFG is not updated.

7.2.3.3 Status Registers

Startup Status Registers

Register STSTAT contains the information used by the boot firmware to identify the start-up settings.

STSTAT

Start-up Status Register

 (0C0_H)

 Reset Value: 000X 80XX_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res				RAM INT		Res				SPD EN	TRS TL	Res	LU DIS	FCB AE	
			r				rh			r		rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOD E	FTM					HWCFG									
rh	rh					rh									

Field	Bits	Type	Description
HWCFG	[7:0]	rh	<p>Hardware Configuration Setting</p> <p>This bit field contains the value that is used by the boot software.</p> <p>This bit field is updated in case of an ApplicationReset with the content of the latches of the HWCFG port pins if bit STSTAT.LUDIS is cleared.</p> <p>This bit field is left unchanged in case of an ApplicationReset and is not updated with the content of the latches of the HWCFG port pins if bit RSTSTAT.SW are cleared and bit STSTAT.LUDIS is set.</p> <p>This bit field is updated with the value written to bit field STCON.HWCFG on a software write action.</p> <p><i>Note: The observed reset value after boot depends upon the state of the HWCFG pins</i></p>

Field	Bits	Type	Description
FTM	[14:8]	rh	Firmware Test Setting In Normal Mode this bit field is updated with 0000000 _B and should be ignored by the boot software.
MODE	15	rh	MODE This bit indicates if the Test Mode is entered or not. 0 _B A Test Mode can be selected 1 _B Normal Mode is selected
FCBAE	16	rh	Flash Config. Sector Access Enable 0 _B Flash config sector is not accessible. Instead the flash memory area is accessed. 1 _B Flash config sector is accessible. The flash memory area can not be accessed. This bit can be cleared by setting bit STCON.CFCBAE. This bit can be set by setting bit STCON.SFCBAE. <i>Note: Reset value of this bit is 0</i>
LUDIS	17	rh	Latch Update Disable 0 _B Bit field STSTAT.HWCFCG is automatically updated with the latched value of the HWCFCG input pins 1 _B Bit field STSTAT.HWCFCG is not updated with the latched value of the HWCFCG input pins This bit can be set by setting bit SYSCON.SETLUDIS. <i>Note: Reset value of this bit is 0</i>
Res	18	rh	Reserved Read as 0; should be written with 0.
TRSTL	19	rh	TRSTL Status This bit simply displays the value of TRSTL.
SPDEN	20	rh	Single Pin DAP Mode Enable 0 _B Single Pin DAP Mode is disabled 1 _B Single Pin DAP Mode is enabled
Res	[23:21]	r	Reserved Read as 0; should be written with 0.

Field	Bits	Type	Description
RAMINT	24	rh	<p>RAM Content Security Integrity</p> <p>In normal operation this bit can be set or cleared by the application (via SYSCON). If a test boot mode is entered, the bit is automatically cleared (and cannot be set again in test mode) because the content may have been altered</p> <p>0_B RAM Security Integrity cannot be guaranteed</p> <p>1_B RAM Security Integrity maintained</p> <p>Note: This bit is reset only by a cold power-on reset.</p>
Res	[31:25]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

7.2.4 NMI Trap Generation

The NMI trap structure is shown in [Figure 7-21](#). The trap request trigger or the corresponding trap set bit (in register TRAPSET) can trigger the NMI trap generation. An NMI trap is always issued simultaneously to all CPUs in the system. The trap flag can be cleared by software by writing to the corresponding bit in register TRAPCLR. A NMI request is only generated if the trap source was not disabled. Otherwise only the trap status flag is set but no NMI request is generated.

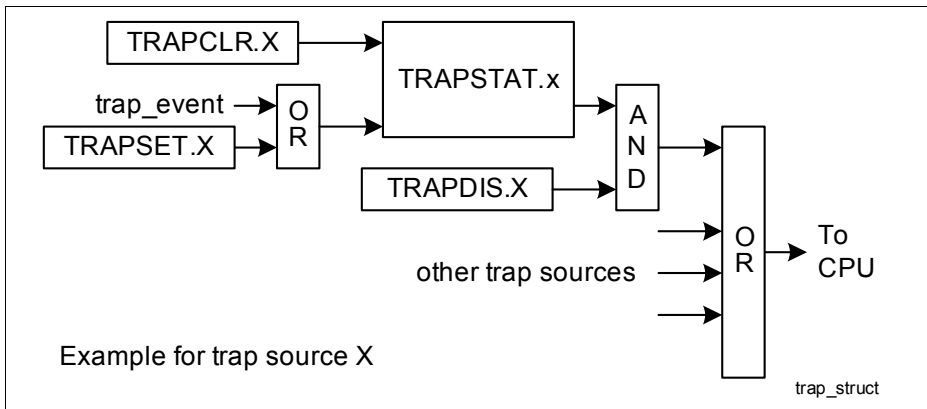


Figure 7-21 NMI Trap Generation

Handling NMI Traps

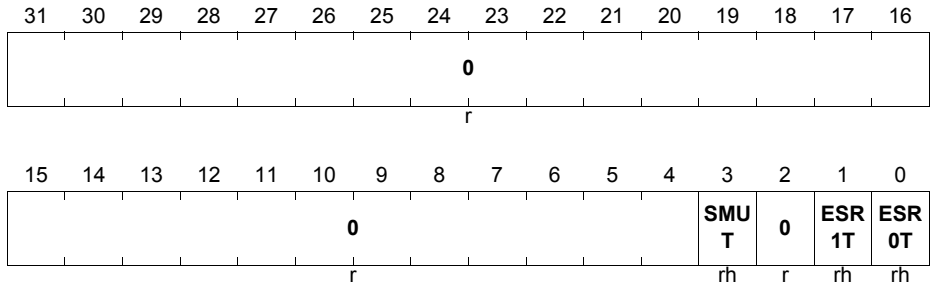
As an NMI trap is generated while the trap source is enabled AND the trap status flag is set, it is recommended to clear the trap status flag before the trap source is enabled. The trap status flag can be set before the trap source is enabled and simply enabling the trap source can result in unintended NMI traps. At the end of a NMI trap handling routine the trap status flag should be cleared.

7.2.4.1 Trap Control Registers

TRAPSTAT

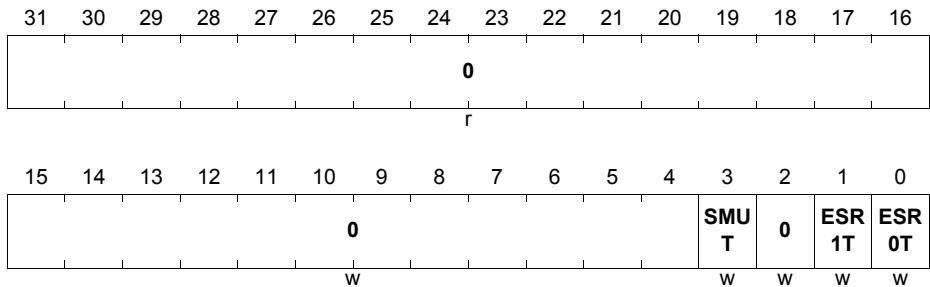
Trap Status Register

 (124_H)

 Reset Value: 0000 000X_H


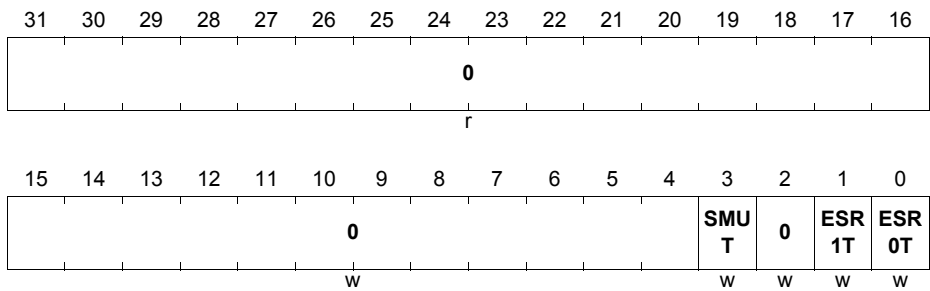
Field	Bits	Type	Description
ESR0T	0	rh	<p>ESR0 Trap Request Flag</p> <p>This bit is set if an ESR0 event is triggered.</p> <p>0_B No trap was requested since this bit was cleared the last time</p> <p>1_B A trap was requested since this bit was cleared the last time</p> <p>This bit can be cleared by setting bit TRAPCLR.ESR0T.</p> <p>This bit can be set by setting bit TRAPSET.ESR0T.</p> <p><i>Note: Observed reset value after boot will depend upon ARI mode because of ESR pin transition, see Table 7-11</i></p>
ESR1T	1	rh	<p>ESR1 Trap Request Flag</p> <p>This bit is set if an ESR1 event is triggered.</p> <p>0_B No trap was requested since this bit was cleared the last time</p> <p>1_B A trap was requested since this bit was cleared the last time</p> <p>This bit can be cleared by setting bit TRAPCLR.ESR1T.</p> <p>This bit can be set by setting bit TRAPSET.ESR1T.</p> <p><i>Note: Reset value of this bit is 0</i></p>

Field	Bits	Type	Description
SMUT	3	rh	SMU Alarm Trap Request Flag This bit is set if an SMU Alarm is indicated . 0_B No trap was requested since this bit was cleared the last time 1_B A trap was requested since this bit was cleared the last time This bit can be cleared by setting bit TRAPCLR.SMUT. This bit can be set by setting bit TRAPSET.SMUT. <i>Note: Reset value of this bit is 0</i>
0	2, [31:4]	r	Reserved Read as 0.

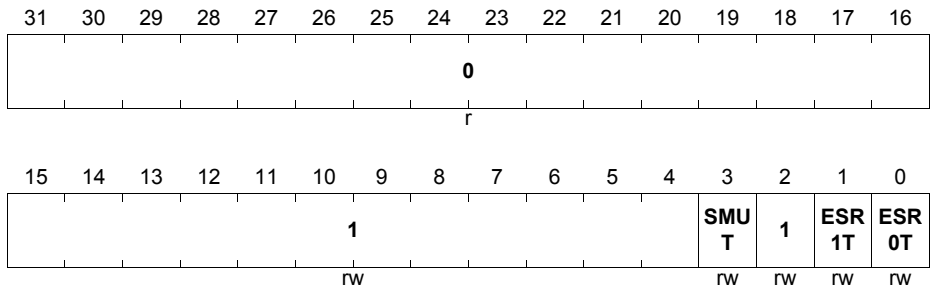
TRAPSET
Trap Set Register
(128_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
ESR0T	0	w	Set Trap Request Flag ESR0T Setting this bit sets bit TRAPSTAT.ESR0T. Clearing this bit has no effect. Reading this bit returns always zero.
ESR1T	1	w	Set Trap Request Flag ESR1T Setting this bit sets bit TRAPSTAT.ESR1T. Clearing this bit has no effect. Reading this bit returns always zero.

Field	Bits	Type	Description
SMUT	3	w	Set Trap Request Flag SMUT Setting this bit sets bit TRAPSTAT.SMUT. Clearing this bit has no effect. Reading this bit returns always zero.
0	2, [31:4]	r	Reserved Read as 0; should be written with 0.

TRAPCLR
Trap Clear Register
(12C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
ESR0T	0	w	Clear Trap Request Flag ESR0T Setting this bit clears bit TRAPSTAT.ESR0T. Clearing this bit has no effect. Reading this bit returns always zero.
ESR1T	1	w	Clear Trap Request Flag ESR1T Setting this bit clears bit TRAPSTAT.ESR1T. Clearing this bit has no effect. Reading this bit returns always zero.
SMUT	3	w	Clear Trap Request Flag SMUT Setting this bit clears bit TRAPSTAT.SMUT. Clearing this bit has no effect. Reading this bit returns always zero.
0	[31:4], 2	r	Reserved Read as 0; should be written with 0.

TRAPDIS
Trap Disable Register
(130_H)
Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ESR0T	0	rw	Disable Trap Request ESR0T 0 _B A trap request can be generated for this source 1 _B No trap request can be generated for this source
ESR1T	1	rw	Disable Trap Request ESR1T 0 _B A trap request can be generated for this source 1 _B No trap request can be generated for this source
SMUT	3	rw	Disable Trap Request SMUT 0 _B A trap request can be generated for this source 1 _B No trap request can be generated for this source
0	[31:16]	r	Reserved Read as 0; should be written with 0.
1	[15:4], 2	r	Reserved Read as 1; should be written with 1.

7.2.5 RCU Register Address

Table 7-16 Registers Address Spaces - RCU Kernel Registers

Module	Base Address	End Address	Note
SCU	F003 6000 _H	F003 63FF _H	-

7.2.6 RCU Kernel Registers

This section describes the kernel registers of the RCU module. Most of the RCU kernel register names described in this section will be referenced in other parts of the TC26x B-Step User's Manual by the module name prefix "SCU_".

RCU Kernel Register Overview

Table 7-17 Register Overview of RCU (Offset from SCU Register Base)

Short Name	Long Name	Offset Addr. 1)	Access Mode		Reset	Description See
			Read	Write		
RSTSTAT	Reset Status Register	050 _H	U, SV	BE	Power-on Reset	Page 7-103
-	Reserved	054 _H	BE	BE	-	-
RSTCON	Reset CON Register	058 _H	U, SV	SV, SE, P	Power-on Reset	Page 7-106
ARSTDIS	Application Reset Disable Register	05C _H	U, SV	SV, SE, P	Power-on Reset	Page 7-109
SWRSTCON	Software Reset Configuration Register	060 _H	U, SV	SV, SE, P	Power-on Reset	Page 7-110
RSTCON2	Reset Configuration Register 2	064 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-111
Reserved	Reserved	068 _H	U, SV	BE	-	-
ESRCFG0	ESR0 Configuration Register	070 _H	U, SV	SV, SE, P	System Reset	Page 7-116

Table 7-17 Register Overview of RCU (Offset from SCU Register Base)

Short Name	Long Name	Offset Addr. 1)	Access Mode		Reset	Description See
			Read	Write		
ESRCFG1	ESR1 Configuration Register	074 _H	U, SV	SV, SE, P	System Reset	Page 7-116
ESROCFG	ESR Reset Output Configuration Register	078 _H	U, SV	SV, SE, P	System Reset	Page 7-117
PDR	Pad Driver Mode Register	09C _H	U, SV	SV, E, P	System Reset	Page 7-124
IOCR	Input/Output Control Register	0A0 _H	U, SV	U, SV, P	System Reset	Page 7-118
OUT	Output Register	0A4 _H	U, SV	U, SV, P	System Reset	Page 7-121
OMR	Output Modification Register	0A8 _H	U, SV	U, SV, P	System Reset	Page 7-121
IN	Input Register	0AC _H	U, SV	BE	System Reset	Page 7-122
STSTAT	Start-up Status Register	0C0 _H	U, SV	BE	Power-on Reset (RAMINT on cold Power-on only)	Page 7-126
Reserved	Reserved	0C4 _H	U, SV	BE	Power-on Reset	
TRAPSTAT	Trap Status Register	124 _H	U, SV	BE	System Reset	Page 7-130
TRAPSET	Trap Set Register	128 _H	U, SV	SV, E, P	System Reset	Page 7-131
TRAPCLR	Trap Clear Register	12C _H	U, SV	U, SV, P	System Reset	Page 7-132
TRAPDIS	Trap Disable Register	130 _H	U, SV	SV, E, P	Application Reset	Page 7-133

1) The absolute register address is calculated as follows:
Module Base Address + Offset Address (shown in this column)

7.3 Power Supply & Power Management Controller (PMC)

This chapter describes the Power Supply of the TC26x B-Step and the Power Management Controller (PMC).

This is described in the following sections:

- Power Supply and Control (see [Section 7.3.1](#))
- Power Management (see [Section 7.3.2](#))
- Power Supply and Power Management Registers (see [Table 7-23](#))

7.3.1 Power Supply and Control

7.3.1.1 Introduction

On-chip linear and switch mode voltage regulators are implemented in TC26x B-Step thereby enabling a single source power supply concept. The external nominal system supply may be either 5 V or 3.3 V. There are two separate parallel Embedded Voltage Regulators (EVR33 and EVR13) generating 3.3 V and 1.3 V supply voltages from the external supply. The nomenclature of these EVRs in the document are EVR33 and EVR13 respectively. Linear Drop Out (LDO) or Switch Mode Power Supply (SMPS) mode may be selected for the EVR13 regulator in case of 5 V or 3.3 V external supply. EVR33 regulator is implemented as a LDO regulator and is required only in case of 5 V external supply.

Depending on the chosen EVR mode, the actual power consumption, EMI requirements and thermal constraints of the system; additional external components like MOSFETs, inductors and capacitors may be required. It is also possible to supply 3.3 V and 1.3 V supply voltages externally thus ensuring compliance to the legacy supply concept.

All supply and generated voltages are monitored internally against overshoot and brownout conditions based on programmable thresholds. In case these thresholds are violated, either a cold power on reset is triggered or an alarm is provided to the SMU.

In case of single 5 V external supply, the ports would predominantly have 5 V logic level. However it is possible to run a part of the port domain, namely the Flexport (P11.x), on 3.3 V logic level even if the device is powered by 5 V. The Flexport could be supplied either directly from the 5 V external supply or from the internally generated 3.3 V EVR supply. The routing of the supply ($V_{\text{FLEX}} = 5 \text{ V}$ or 3.3 V) for the Flexport domain in both cases is done externally. In case of single 3.3 V external supply ($V_{\text{EXT}} = V_{\text{DDP3}} = V_{\text{FLEX}} = V_{\text{DDM}} = 3.3 \text{ V}$), the complete port and analog domain would have 3.3 V logic level.

All internal supplies except analog domain (V_{AREFX} & V_{DDM}) may be supplied by the EVRs. The analog supply domain is separated from the main EVR supply domain and can be supplied by separate external regulators or trackers. It is possible to have a mixed supply scheme with a 5 V ADC domain ($V_{\text{DDM}} / V_{\text{AREFX}} = 5 \text{ V}$) and the remaining system running on 3.3 V supply ($V_{\text{EXT}} = V_{\text{DDP3}} = 3.3 \text{ V}$).

7.3.1.2 Supply Mode and Topology Selection

The choice of the supply scheme at startup is based on the latched status of HWCFG[0:2] pins before PORST release and is indicated by **PMSWSTAT.HWCFG**EVR status flags. Following supply modes are supported and are further enumerated in **Table 7-18**.

- Single source 5 V supply level is supported in following topologies.

- EVR13 in SMPS mode with external switches and EVR33 in LDO mode with internal pass devices.
- EVR13 and EVR33 both in LDO mode with internal pass devices.
- EVR13 in LDO mode with external pass device and EVR33 in LDO mode with internal pass devices.
- Single source 3.3 V supply level is supported in following topologies.
 - EVR13 in SMPS mode with external switches and EVR33 is inactive.
 - EVR13 in LDO mode with internal pass devices and EVR33 is inactive.
 - EVR13 in LDO mode with external pass device and EVR33 is inactive.
- Supplies are provided externally and the respective EVRs are in disabled state.
 - 5 V and 1.3 V supplied externally. EVR33 in LDO mode with internal pass devices.
 - 5 V and 3.3 V supplied externally. 1.3 V is generated using the EVR13.
 - 5 V, 3.3 V and 1.3 V are supplied externally.

EVR13 is enabled or disabled at startup via the HWCFG[2] configuration pin. The selection between LDO or SMPS topology for EVR13 is decided via HWCFG[0] configuration pin. In case EVR13 SMPS mode is selected, V_{GATE1P} and V_{GATE1N} pins shall be connected to the gate of an external P-channel MOSFET and N-channel MOSFET respectively as shown in [Figure 7-27](#). In case EVR13 LDO mode with external pass device is selected, V_{GATE1P} pin shall be connected to the gate of an external P-channel MOSFET pass device as shown in [Figure 7-23](#). The pass device detector senses the presence of external pass device based on the current sinking capability of the V_{GATE1P} pin as indicated in [EVRSTAT.EXTPASS13](#) flag. In case EVR13 LDO with internal pass devices is selected, the gate control pad V_{GATE1P} shall be connected to ground as shown in [Figure 7-23](#).

EVR33 is enabled or disabled at startup via the HWCFG[1] configuration pin. In case of single source 3.3 V supply, EVR33 is disabled and V_{DDX3} & V_{EXT} pins are supplied externally by 3.3 V. In case EVR33 LDO mode is selected, internal pass devices may be used.

In case of smaller packages, HWCFG[0:2] configuration pins may be absent and both EVRs are active at startup in LDO mode. EVR33 may be disabled via [EVR33CON.EVR33OFF](#) bit and EVR13 via [EVR13CON.EVR13OFF](#) bit if required. The current state of EVRs are reflected in [EVRSTAT.EVRx3](#) flags. HWCFG[6] pin is latched during start-up to decide the default pin behaviour. Ports behave as inputs with pull-up if HWCFG[6] = 1 or are in tristate if HWCFG[6] = 0 as reflected in [PMSWSTAT.TRIST](#) bit. All supplies must be available and be stable with the release of the \overline{PORST} signal.








Table 7-18 Supply Mode and Topology selection

No.	HWCFG [0:2] ¹⁾	VGATE1P ²⁾ VGATE1N ³⁾	Supply Pin Voltage Level / Source	Selected Supply Scheme
a.)	011 _B	VGATE1P/ VGATE1N connected to gate of P- /N-ch. MOSFET.	$V_{EXT} = 5.0\text{ V}$ $V_{DDM}/V_{AREFX} = 5\text{ V} / 3.3\text{ V}$ $V_{FLEX} = 5\text{ V} / 3.3\text{ V}$ $V_{DDP3}/V_{DDFL3} = \text{EVR33}$ $V_{DD} = \text{EVR13}$ $V_{SS}/V_{SSM}/V_{AGND} = 0\text{ V}$	5 V single source supply, EVR13 in SMPS mode, EVR33 in LDO mode. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode supported.
b.)	111 _B	VGATE1P shall be connected to ground.	$V_{EXT} = 5.0\text{ V}$ $V_{DDM}/V_{AREFX} = 5\text{ V} / 3.3\text{ V}$ $V_{FLEX} = 5\text{ V} / 3.3\text{ V}$ $V_{DDP3}/V_{DDFL3} = \text{EVR33}$ $V_{DD} = \text{EVR13}$ $V_{SS}/V_{SSM}/V_{AGND} = 0\text{ V}$	5 V single source supply, EVR13 in LDO mode with internal pass devices, EVR33 in LDO mode. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode supported
c.)	111 _B	VGATE1P shall be connected to P-ch. MOSFET.	$V_{EXT} = 5.0\text{ V}$ $V_{DDM}/V_{AREFX} = 5\text{ V} / 3.3\text{ V}$ $V_{FLEX} = 5\text{ V} / 3.3\text{ V}$ $V_{DDP3}/V_{DDFL3} = \text{EVR33}$ $V_{DD} = \text{EVR13}$ $V_{SS}/V_{SSM}/V_{AGND} = 0\text{ V}$	5 V single source supply, EVR13 in LDO mode with external pass device, EVR33 in LDO mode. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode supported.
d.)	X10 _B	VGATE1P has int. pull- up active when left open.	$V_{EXT} = 5.0\text{ V}$ $V_{DDM}/V_{AREFX} = 5\text{ V} / 3.3\text{ V}$ $V_{FLEX} = 5\text{ V} / 3.3\text{ V}$ $V_{DDP3}/V_{DDFL3} = \text{EVR33}$ $V_{DD} = 1.3\text{ V external}$ $V_{SS}/V_{SSM}/V_{AGND} = 0\text{ V}$	5 V & 1.3 V external supply, EVR13 inactive, EVR33 in LDO mode. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode is supported and 1.3V supply shall be switched off by external regulator after Standby state is entered.
e.)	001 _B ⁴⁾	VGATE1P/ VGATE1N connected to gate of P- /N-ch. MOSFET.	$V_{EXT}/V_{DDP3}/V_{DDFL3} = 3.3\text{ V}$ $V_{DDM}/V_{AREFX} = 5\text{ V} / 3.3\text{ V}$ $V_{FLEX} = 3.3\text{ V}$ $V_{DD} = \text{EVR13}$ $V_{SS}/V_{SSM}/V_{AGND} = 0\text{ V}$	3.3 V single source supply, EVR13 in SMPS mode, EVR33 inactive. 5 V or 3.3 V ADC domain. 3.3 V Flexport domain. Standby Mode supported.

Table 7-18 Supply Mode and Topology selection

No.	HWCFG [0:2] ¹⁾	VGATE1P ²⁾ VGATE1N ³⁾	Supply Pin Voltage Level / Source	Selected Supply Scheme
f.)	101 _B ⁴⁾	VGATE1P shall be connected to ground.	$V_{EXT}/V_{DDP3}/V_{DDFL3} = 3.3V$ $V_{DDM}/V_{AREFX} = 5V / 3.3V$ $V_{FLEX} = 3.3V$ $V_{DD} = EVR13$ $V_{SS}/V_{SSM}/V_{AGND} = 0V$	3.3 V single source supply, EVR13 in LDO mode with internal pass devices, EVR33 inactive. 5 V or 3.3 V ADC domain. 3.3 V Flexport domain. Standby Mode supported.
g.)	101 _B ⁴⁾	VGATE1P connected to P-ch. MOSFET.	$V_{EXT}/V_{DDP3}/V_{DDFL3} = 3.3V$ $V_{DDM}/V_{AREFX} = 5V / 3.3V$ $V_{FLEX} = 3.3V$ $V_{DD} = EVR13$ $V_{SS}/V_{SSM}/V_{AGND} = 0V$	3.3 V single source supply, EVR13 in LDO mode with external pass device, EVR33 inactive. 5 V or 3.3 V ADC domain. 3.3 V Flexport domain. Standby Mode supported
h.)	X00 _B	VGATE1P has int. pull-up active when left open.	$V_{EXT} = 5V \text{ or } 3.3V$ $V_{DDM}/V_{AREFX} = 5V / 3.3V$ $V_{FLEX} = 5V \text{ or } 3.3V$ $V_{DDP3}/V_{DDFL3} = 3.3V$ $V_{DD} = 1.3V$ $V_{SS}/V_{SSM}/V_{AGND} = 0V$	5 V, 3.3 V and 1.3 V are supplied externally, EVR13 and EVR33 inactive. 5 V or 3.3 V ADC domain. 5 V or 3.3 V Flexport domain. Standby Mode is supported and 3.3V and 1.3V supplies shall be switched off by external regulator after Standby state is entered.

- 1) if HWCFG[0,1,2,6] pins are left unconnected, it is ensured that EVR33 and EVR13 (LDO) are active by default.
- 2) VGATE1P pin is connected to a P- ch. MOSFET in case of EVR13 SMPS or LDO with external pass device. In case EVR13 LDO uses internal pass devices, VGATE1P pin is to be connected to ground. In case EVR13 is inactive, internal pull-up is active at VGATE1P pin as long as the pin is not externally connected to ground.
- 3) VGATE1N pin need to be connected to the gate of the N- channel MOSFET in case SMPS regulator mode is selected. In case EVR13 LDO mode is selected or EVR13 is inactive, VGATE1N pin behaves like a normal port pin (P32.0) and is default configured as input with weak internal pull-down after start-up/cold \overline{PORST} .
- 4) In this HWCFG combination, also VEXT may be supplied with 5V instead of 3.3V nominal voltage but with all other supply pin voltages remaining the same. This is to support the external supply case that 5 V and 3.3 V is supplied externally and 1.3 V is generated using the EVR13 regulator. It has to be ensured that the external regulator supplies the 5V at VEXT pins and 3.3V at VDDP3/VDDFL3 pins.

HWCFCG [0] P14.6	HWCFCG [1] P14.5	HWCFCG [2] P14.2	HWCFCG [3] P14.3	HWCFCG [4] P10.5	HWCFCG [5] P10.6	HWCFCG [6] P14.4
						
0 - SMPS 1 - LDO (default)	0 - EVR33OFF 1 - EVR33ON (default)	0 - EVR13OFF 1 - EVR13ON (default)	0 - Boot from pins HWCFCG [5:4] 1 - Flash BML boot (default)	HWCFCG [4:5] [0 0]- Generic Bootstrap (P14.0/1) [0 1]- ABM, Generic Bootstrap on fail (P14.0/1) [1 0]- ABM, ASC Bootstrap on fail (P15.2/3) [1 1]- Internal start from Flash (default)		Default Pad state 0 - Pins in tristate 1 - Pins with pull-up (default)

- 1.) HWCFCG [6] has weak internal pull-up active at start-up if the pin is left unconnected.
- 2.) If HWCFCG [6] is left unconnected or is externally pulled high, HWCFCG [0:5] pins have weak internal pull-ups active at start-up.
- 3.) If HWCFCG [6] is connected to ground, HWCFCG [0:5] pins are in tristate. External pull devices required for all HWCFCG pins.
- 4.) In packages smaller than QFP 144, HWCFCG [0:2] pins are absent & internally pulled high ensuring EVR13 (LDO) & EVR33 is active
- 5.) HWCFCG [0:2] and HWCFCG [6] pins are latched during supply ramp-up (VEXT < 2.97V) and stored in PMSWSTAT.HWCFCGEVR & TRIST register bits. The remaining HWCFCG pins are latched on internal reset release (between 100us – 180us after reset assertion) and stored in STSTAT register.

Figure 7-22 Hardware Configuration (HWCFCG) pins

7.3.1.3 Linear Regulator Mode

The LDO part of the EVR composes of the following sub-blocks.

- EVR33 regulator
- EVR13 regulator

The EVR13 regulator supplies mainly the core domain including RAMs. The EVR33 regulator supplies the flash modules, LVDS pads, oscillators and the JTAG interface. The EVR33 is additionally capable of driving a limited amount of non-inductive 3.3 V external load and may supply the Flexport in case configured as a 3.3 V Port. Each EVR constitutes a digital regulator, a pass device control unit and a voltage feedback loop. The pass device outputs are appropriately buffered by capacitors to handle load transients so as not to violate the operating voltage limits. Each EVR can be individually disabled via HWCFG pins as described in [Chapter 7.3.1.2](#). Each EVR can also be disabled via protected registers to enter power saving modes. The EVR13 can be disabled via [EVR13CON.EVR13OFF](#) and EVR33 via [EVR33CON.EVR33OFF](#) bits respectively.

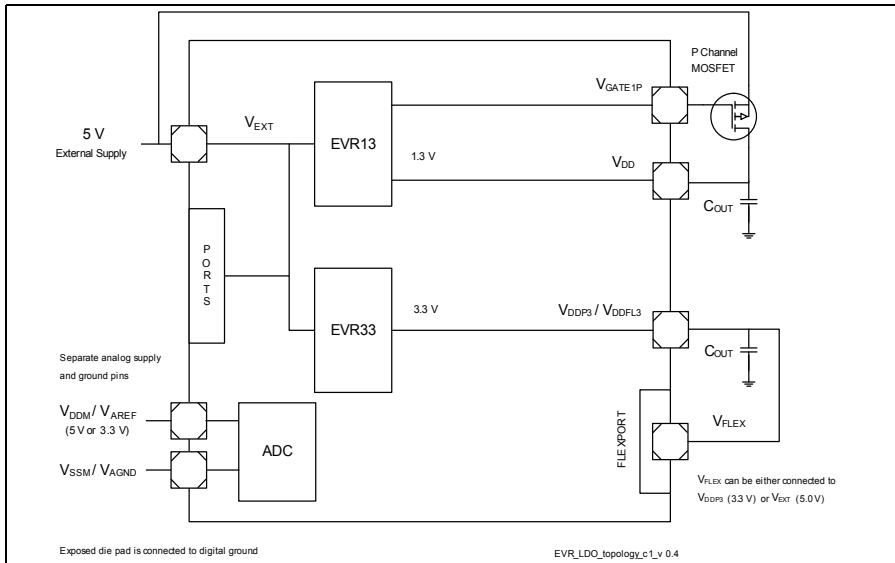


Figure 7-23 EVR LDO topology (c) - 5 V single supply with ext. pass device

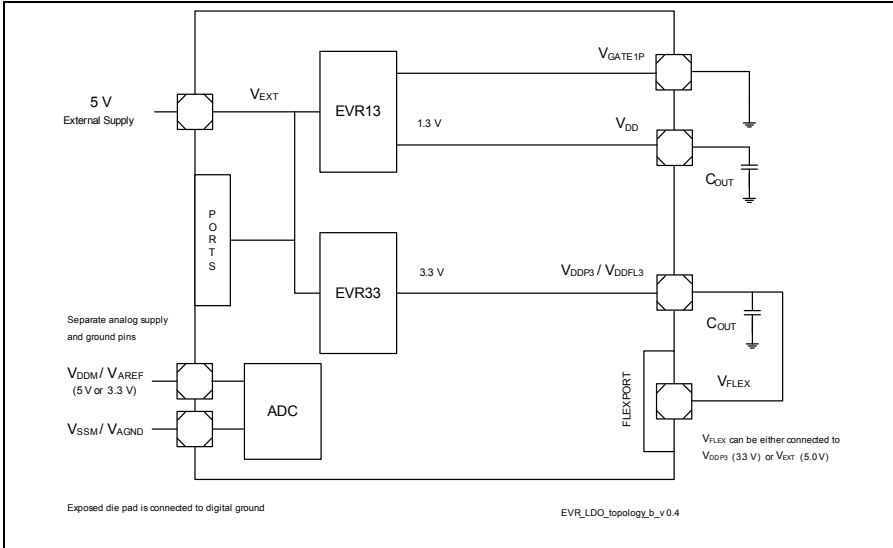


Figure 7-24 EVR LDO topology (b) - 5 V single supply with int. pass device

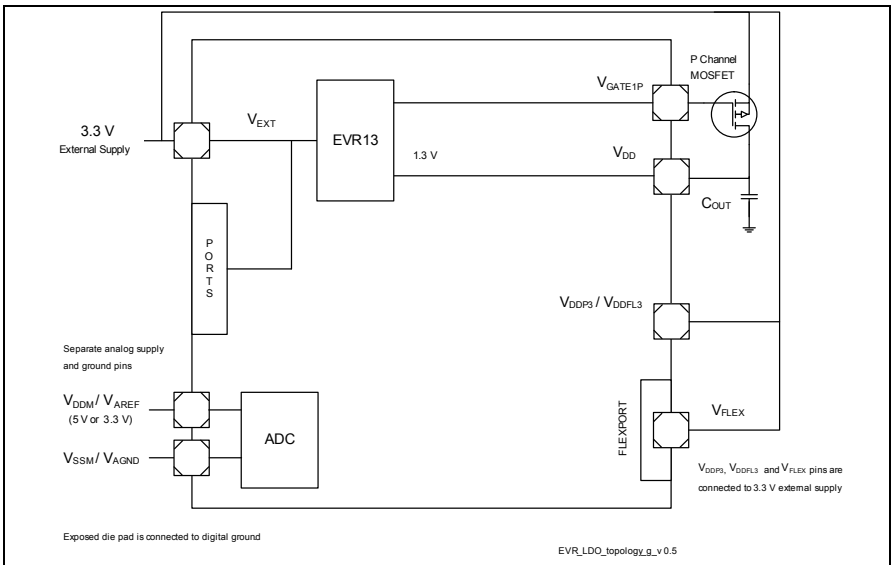


Figure 7-25 EVR LDO topology (g) - 3.3 V single supply with ext. pass device

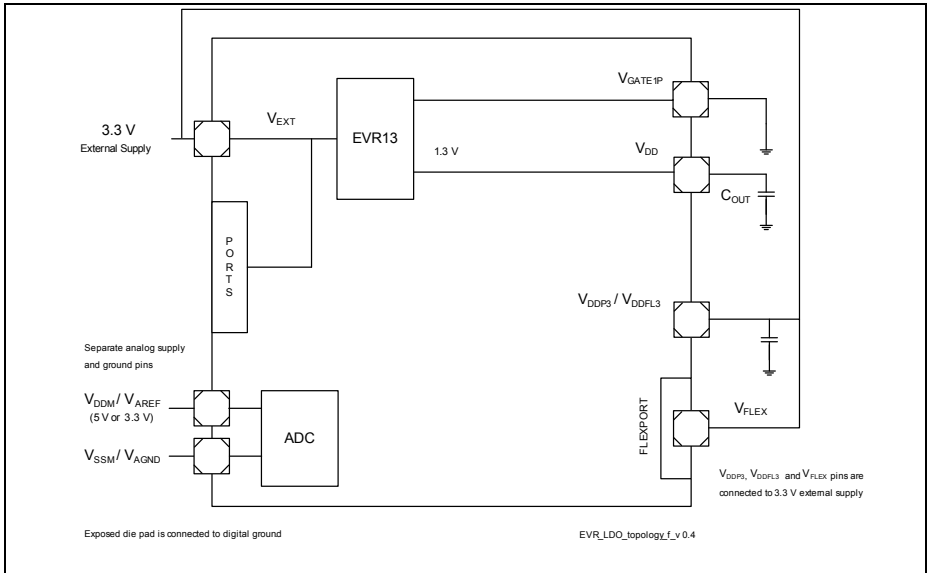


Figure 7-26 EVR LDO topology (f) - 3.3 V single supply with int. pass device

7.3.1.4 Step-down Regulator Mode

The Step-down regulator provides a higher efficiency of power conversion compared to the linear voltage regulator. However it requires additional external components and injects more switching noise into the system. The integrated DC/DC regulator modulates an external charge device to buffer the energy in a LC filter in order to generate a regulated core supply. The 5 V or 3.3 V external V_{EXT} supply shall be provided to the complementary MOSFET switches as shown in [Figure 7-27](#) and [Figure 7-28](#).

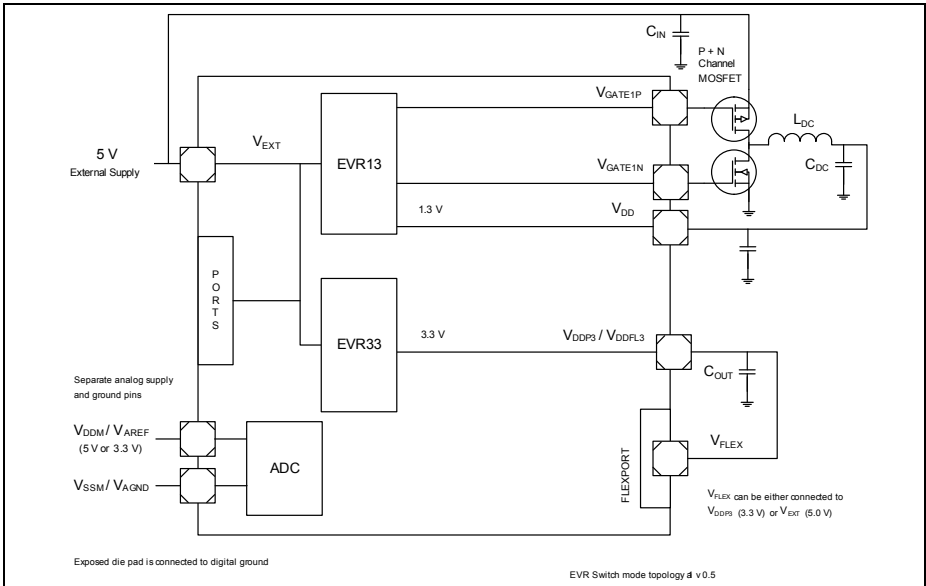


Figure 7-27 EVR Switch mode topology (a) - 5 V single supply

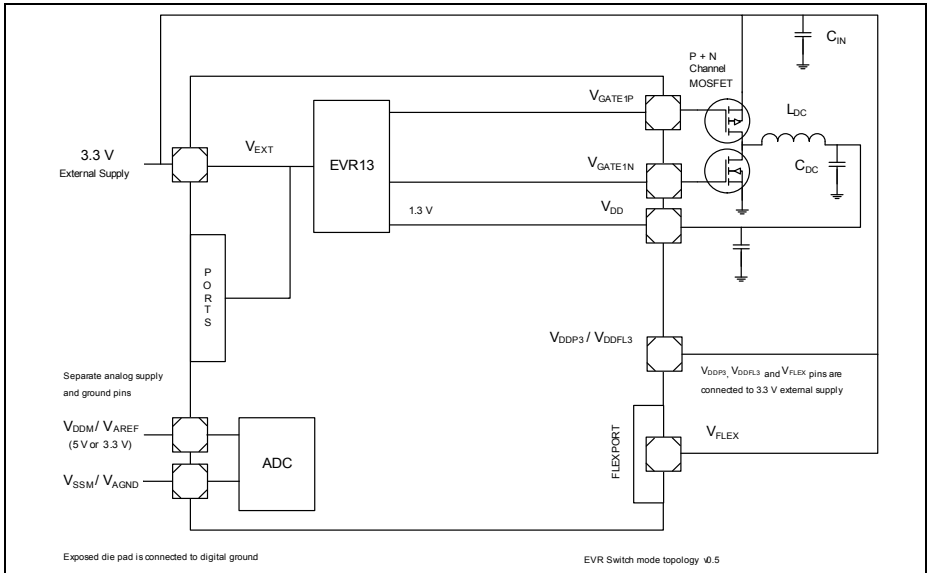


Figure 7-28 EVR Switch mode topology (e) - 3.3 V single supply

The control strategy involves synchronous switching of the charge device at a defined switching frequency using pulse width modulation. The switching frequency is selected between 0.4 MHz to 2.0 MHz based on application needs. The duty cycle has a resolution of the base clock frequency (100MHz internal clock source) and is controlled using a PID controller which reacts to the deviation of the output voltage against a reference voltage. The output voltage value is approximately equal to the input voltage multiplied by the duty cycle factor.

During the start-up phase, a different control strategy is used in order to avoid current overload in the coil and overshoot of the output voltage. In this phase, only the P-channel MOSFET is switched ON/OFF while the N-Channel MOSFET remains always OFF and behaves like a diode. This prevents the coil current from becoming negative in the start-up phase with a pre charged capacitor (especially immediately after a reset). The open loop operation ends when the ADC indicates that the output voltage has reached the target value. At this point the PID controller is configured and the normal closed loop operation begins. After a voltage transient (typically an overshoot), EVR13 is ready. The default switching frequency is 1.5 MHz during EVR start-up and is derived from the internal 100 MHz clock source. The step down regulator is consequently trimmed by the Firmware after reset release to achieve a more accurate EVR13 output voltage.

It is possible to synchronize an external step-down regulator which may run at much lower switching frequencies than the internal regulator. A scaled synchronisation clock output is provided and routed to an external pin (P33.13 pin - DCDCSYNC output).

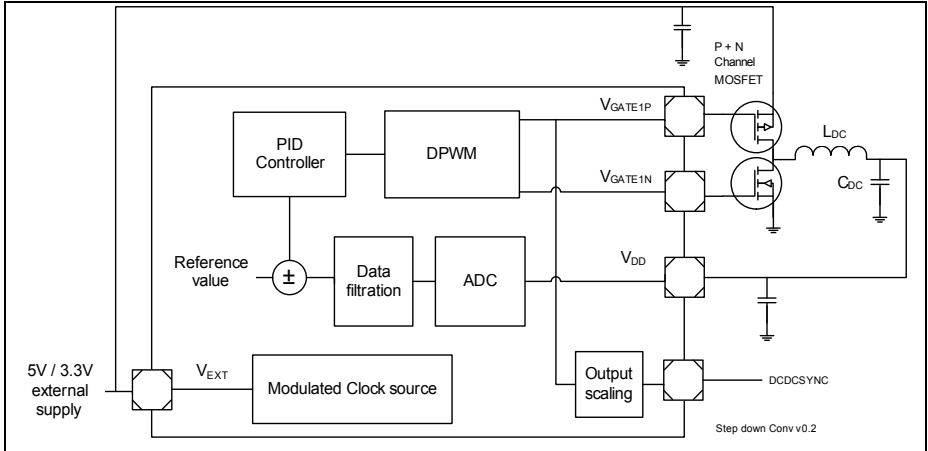


Figure 7-29 Step down regulator

Frequency spreading may be activated in case of SMPS mode to comply with EMI / EMC requirements. The spreading increases the switching period and 2% spreading as configured in [EVRSDCTRL1.SDFREQSPRD](#) field means that the average switching periods are 2% below the programmed switching frequency which is configured in [EVRSDCTRL1.SDFREQ](#) field.

7.3.1.5 External Supply Modes

The external supply modes involve deactivating any or both of the EVR13 and EVR33 regulators. In this mode, EVR33 is disabled via the HWCFG[1] configuration pin and the EVR13 is disabled via the HWCFG[2] configuration pin respectively.

Following external supply modes are supported.

- 5 V and 1.3 V supplied externally. 3.3 V is generated using the EVR33 regulator as shown in **Figure 7-30**.
- 5 V and 3.3 V supplied externally. 1.3 V is generated using the EVR13 regulator.
- 5 V, 3.3 V and 1.3 V are all supplied externally as shown in **Figure 7-31**.

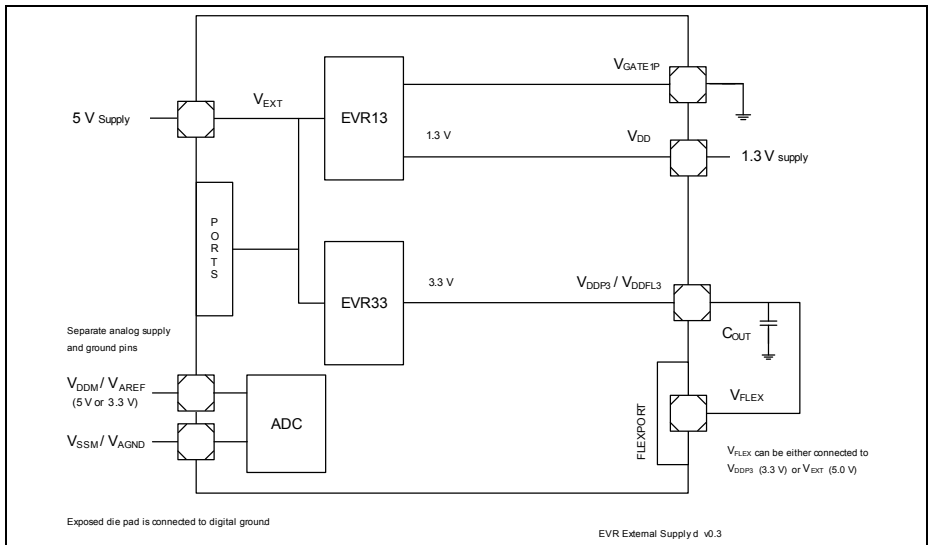


Figure 7-30 External Supply mode (d) - 5 V and 1.3 V externally supplied

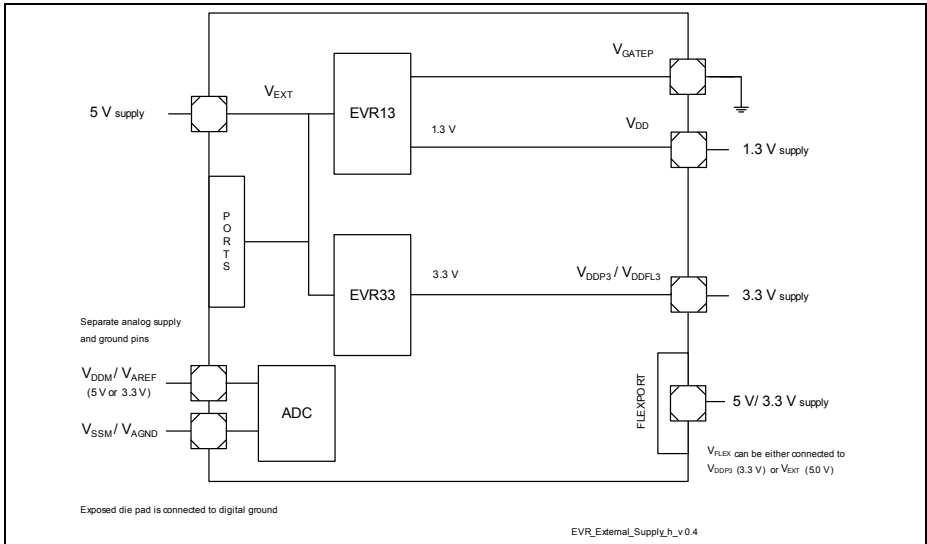


Figure 7-31 External Supply mode (h) - 5 V, 3.3 V & 1.3 V externally supplied

7.3.1.6 Components and Layout

The efficiency of the step-down regulator is influenced by the characteristics of the selected components and also the placement and routing of the components on the PCB. The additional external components constitute a coil, a capacitor and a complementary P-channel and N-channel MOSFET.

The coil need to have a low ESR and a relatively constant characteristic against temperature and current variations. Likewise the capacitor need to have a low ESR, a constant frequency characteristic and minimum DC voltage dependence. The capacitors may be a parallel combination of smaller and larger capacitors or capacitors of equal size for better efficiency. The P-channel MOSFET is preferred with lower R_{DS_ON} and gate capacitance values. The P-channel MOSFET gate need to be connected to VGATE1P pin which is a common gate pin used also to connect the pass device in EVR13 LDO mode. The N-channel MOSFET gate needs to be connected to VGATE1N pin for synchronous switching of the step-down regulator.

In case of usage of Emulation devices, it should be taken care that the component choice also considers the current additionally drawn by ED RAM and EEC part.

Component parameters and requirements would be documented in datasheet.

It should be taken care that each supply pin in QFP packages or a pair of supply pins in BGA packages has a decoupling capacitor close to the pins. Supply pins belonging to a common supply rail shall be connected together after the respective decoupling

capacitors and shall be buffered by an additional larger capacitor based on the constraints of the regulator which supplies the rail. It should be taken care to have a low trace resistance to the decoupling capacitors and buffer capacitors for better performance and EMI/EMC behaviour. The dimensioning of the buffer capacitors is based predominantly on the load jumps triggered during reset events and stability criteria of the regulator. During software triggered non reset events, it is recommended to limit the load jumps (di_{EXT}/dt , di_{DD}/dt) to a maximum of 100 mA/100 us. For example, during clock ramp-up phase it is recommended to limit the clock switching steps so as not to violate 100 mA/100 us limit. However in case of non deterministic events like loss of PLL lock or an NMI event to all CPUs, a higher current jump is expected.

7.3.1.7 Voltage Monitoring

The PMC module implements a staggered voltage monitoring build upon a primary and a secondary monitor. The primary monitor ensures that the microcontroller is put into a reset state when the lowest operational threshold is violated. The secondary monitor serves as an additional safety monitor providing over- and under-voltage alarms.

Primary under-voltage monitoring of the external V_{EXT} supply, V_{DD} / EVR13 supply and V_{DDP3} / EVR33 supply are inherently carried out to ensure proper functioning of the system. The EVR control loop as well as the under-voltage monitors are based on a primary bandgap reference. The thresholds for the basic monitoring are non-configurable and represents the lowest possible thresholds for the correct functioning of the system. In case of violation of these thresholds, cold \overline{PORST} is activated and \overline{PORST} pin is pulled low (strong current sink). Following the reset release and firmware boot, it can be inferred from STBYR, EVR13, EVR33 or SWD bits in RSTSTAT register as to whether the violation of these thresholds led to the previous reset. The primary under-voltage monitoring is kept active even if the respective EVRs have been disabled and the supply is provided externally as shown in [Table 7-19](#).

In case of EVR33, primary under-voltage monitoring may be disabled for a definite duration via [EVR33CON.RST33OFF](#) bit. This allows handling a higher voltage drop of external supply up to 3.0 V without issuing a reset.

It needs to be ensured that none of the 3.3 V modules like Flash, system PLL and oscillator are active during the phase when VDDP3 voltage sinks below the primary reset threshold level. Once the VDDP3 supply regains the primary reset threshold level indicated by a secondary alarm event, the modules may be actively used again.

The primary Supply WatchDog (SWD monitor) monitors the ramp-up of external supply voltage and keeps the microcontroller in cold Power On Reset state as long as the supply has not reached the operational range. Likewise, it also detects ramp-down or brown out condition of external supply and sets the device into a cold Power On Reset state when the voltage has dropped below the lowest operational threshold. Nevertheless, It is recommended to monitor externally all supplies generated external to the microcontroller and to assert \overline{PORST} reset pin in case of violation of the lowest operational limits. In case of 5 V V_{EXT} nominal external supply and 3.3 V in turn being generated by the internal EVR33 LDO regulator, the external 5 V supply need to be kept above 4 V considering the pass device drop-out margin. A drop of V_{EXT} below this value may lead to a cold power-on reset as the generated 3.3 V supply may drop below the primary under-voltage threshold limit. The limits are documented in the datasheet.

The external V_{EXT} supply, V_{DD} / EVR13 and V_{DDP3} / EVR33 supplies are measured by 8 bit analog converters and the measured value is indicated in [EVRADCSTAT](#) register. This may be used to do a plausibility check of external regulator references.

Additional secondary over-voltage and under-voltage monitoring against programmable thresholds is provided for all supplied and generated voltages using a secondary bandgap reference. This includes the external supply voltage and the internally

generated EVR13 and EVR33 output voltages as shown in [Figure 7-32](#). The secondary voltage monitors are kept active even if the respective EVRs have been disabled and the supply is provided externally as shown in [Table 7-19](#). In case of a threshold violation, an SMU alarm event is generated.

The secondary monitor violation is notified depending on the direction of voltage transition as programmed in [EVRMONCTRL](#) register. The appropriate thresholds for voltage monitoring can be programmed in the [EVROVMON](#) and [EVRUVMON](#) registers. In case of an active monitoring violation, respective status flags are set in the [EVRSTAT](#) register. It can be inferred from OV13, OV33 or OVSWD bits in [EVRSTAT](#) register as to whether over-voltage thresholds for the respective voltage domains were violated. Likewise, it can be inferred from UV13, UV33 or UVSWD bits in [EVRSTAT](#) register as to whether under-voltage thresholds were violated. It is also possible to deactivate the generation of SMU alarms from secondary monitors in [EVRMONCTRL](#) register.

The primary bandgap is compared with a secondary bandgap to detect bandgap drifts as reflected in [EVRSTAT.BGPROK](#) register bit. The bandgap BIST is only carried out during a supply ramp-up.

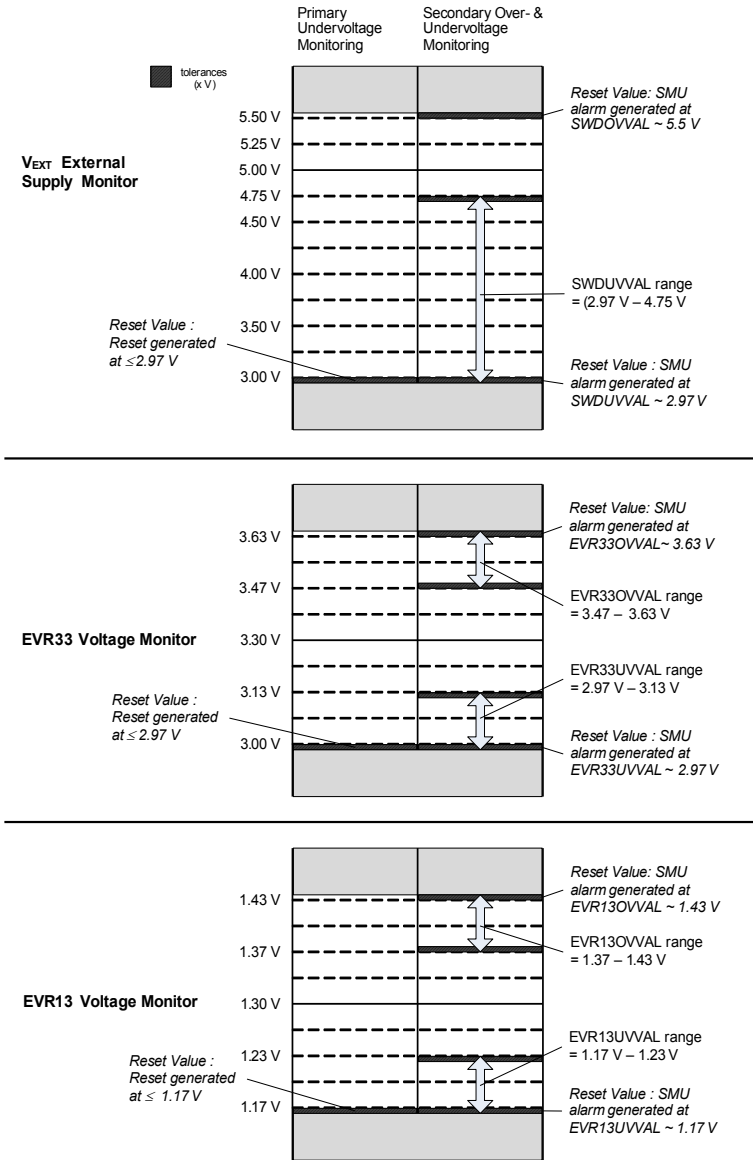
In case of primary monitor violation, respective status bits are set to indicate the event as shown in [Table 7-19](#). These bits maybe evaluated during consequent start-up after cold power-fail reset to recognize which among the supply rails had the power-fail. Likewise for secondary monitor violation, the respective status bits may be evaluated to discriminate between an overvoltage or an undervoltage event and to recognize which supply rail had triggered the alarm event to SMU.

Table 7-19 Voltage Monitoring

Supply Pin / Rail	Primary Under-voltage monitor State (ON/OFF)	Secondary Over & Under-voltage Monitor State (ON/OFF)	Supply Range V	Is the Pin supplied
RUN or SLEEP system mode during supply modes a-h.				
V _{EXT}	ON. RSTSTAT.SWD RSTSTAT.STBYR	ON EVRSTAT.OVSWD EVRSTAT.UVSWD	2.97– 5.50 V	External 5V or 3.3V Supply to be provided
V _{DDM}	-	-	2.97– 5.50 V	External Supply to be provided
V _{DDP3}	ON RSTSTAT.EVR33	ON EVRSTAT.OV33 EVRSTAT.UV33	2.97– 3.63 V	EVR33 active or external 3.3V supply to be provided
V _{DD}	ON RSTSTAT.EVR13	ON EVRSTAT.OV13 EVRSTAT.UV13	1.17– 1.43 V	EVR13 active or external 1.3V supply to be provided

Table 7-19 Voltage Monitoring

Supply Pin / Rail	Primary Under-voltage monitor State (ON/OFF)	Secondary Over & Under-voltage Monitor State (ON/OFF)	Supply Range V	Is the Pin supplied
STANDBY system mode during supply modes a-h.				
V_{EXT}	OFF RSTSTAT.STBYR	OFF	2.97– 5.50 V	ON
V_{DDM}	-	-	0– 5.50 V	ON or OFF
V_{DDP3}	OFF	OFF	0 V	OFF
V_{DD}	OFF	OFF	0 V	OFF



Monitoring tolerances and levels are specified in datasheet section EVR : Supply monitoring

Voltage_Monitoring_v0.3

Figure 7-32 Voltage Monitoring

Scaling EVR set point values and primary / secondary monitor thresholds

The adaptation of the EVR set point values, primary reset thresholds and secondary over and undervoltage thresholds shall be carried out in a sequential manner.

For example, in case of a change of VDD core supply range from the default 1.3V ± 10% (1.170 V to 1.430 V) to 1.3V - 5% + 10% (1.235 V to 1.430 V), following procedure may be carried out. Depending on external regulator constraints and application requirements, the setpoint may alternatively be raised by + 2.5% to 1.33 V nominal value to have an operational range of 1.33 V ± 7.5% symmetrical across the nominal value.

- Disable the Core supply secondary undervoltage alarm via **EVRRMONCTRL** register. Overvoltage monitoring is left active at the default 1.43 V (10%) limit as configured in **EVROVMON.EVR13OVVAL** register bits.
 - **EVRRMONCTRL**=0021 2101_H or **EVRRMONCTRL.EVR13UVMOD**=00_B.
 - **EVROVMON.EVR13OVVAL** = $FD_H - 253_D \sim 1.43V - 1.45V \sim 11\%$ (reset value)
- The EVR13 setpoint value maybe changed in maximum steps of 2.5 % to limit the current dynamics. The default EVR13 setpoint value is changed in **EVRTTRIM.EVR13TRIM** register in case LDO regulator is used. In case of SMPS regulator, **EVRTTRIM.SDVOUTSEL** value has to be changed. In case VDD core supply is provided externally, then the external regulator need to be configured so that it provides 1.33 V instead of 1.3 V.
- **EVRTTRIM.EVR13TRIM** = $[(1.33 / 5.7692 \text{ mV}) + 1] = E7_H = 231_D$.
- **EVRTTRIM.SDVOUTSEL** = $[(VIN - 725 \text{ mV}) / 5 \text{ mV}] = 79_H = 121_D$.
- VDD / EVR13 voltage value is measured and is indicated in **EVRRADCSTAT.ADC13V** register bits. This may allow a plausibility check whether the voltage change happened by comparing the value before and after the change. The voltage change would take less than 20 us in case of internal regulators to be visible at the VDD rail.
- The primary reset threshold limit is trimmed via **EVRRSTCON.RST13TRIM** register bits. The reset value is raised by 5% from 1.170 V at VDD pin to 1.235 V at VDD pin.
- **EVRRSTCON.RST13TRIM** = $[(1,235 \text{ V} / 5.7692 \text{ mV}) - 5] = D0_H = 208_D$
- In case not enough margin is available for the application, secondary undervoltage monitoring of core voltage maybe completely disabled. Otherwise the default limit of **EVRRVUMON.EVR13UVVAL** bits may be adapted by raising it by 5%.
 - **EVRRVUMON.EVR13UVVAL** = $[(1,235 \text{ V} / 5.7692 \text{ mV}) - 1] = D4_H = 212_D$.

7.3.1.8 100 MHz EVR Clock Source

The 100 MHz clock source is a precise back-up on-chip clock supplied by EVR Pre-regulator. The clock is used by the main EVRx3 regulators and monitors and serves as the system clock during the Startup phase. It is further used as an independent clock reference for clock monitoring and can be used as a back-up clock in case of loss of lock or crystal failures.

7.3.1.9 Sequence during Power-up and Power-down

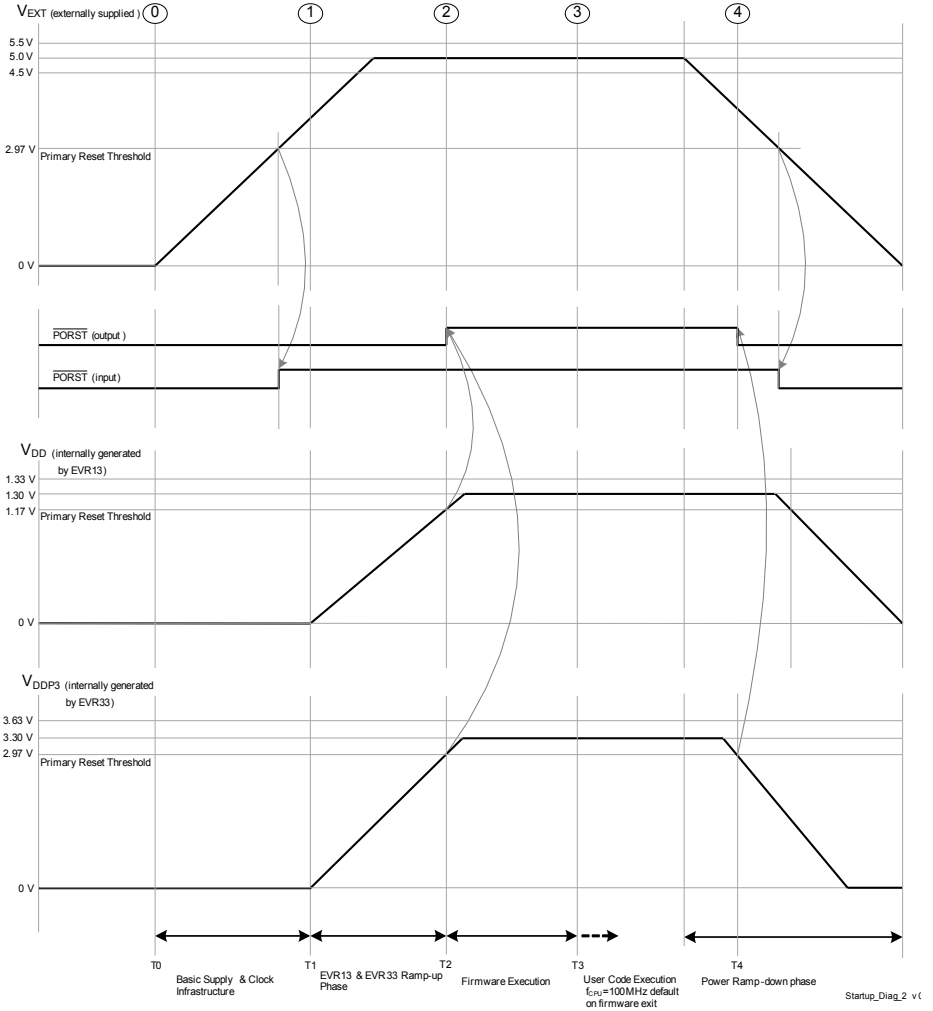


Figure 7-33 Single Supply mode (a, b & c) - 5 V single supply

Single Supply mode (a, b & c)

5 V single supply mode. 1.3 V and 3.3 V are generated internally by the EVR13 and EVR33 internal regulators.

- The rate at which current is drawn from the external regulator (dI_{EXT}/dt) is limited during the basic infrastructure and EVRx3 regulator start-up phase (T_0 upto T_2) to a maximum of 100 mA/100 μ s. EVRx3 is also robust against a voltage ramp-up starting from a residual voltage between 0-1 V. Start-up slew rates for supply rails should comply to datasheet values.
- Furthermore it is also ensured that the current drawn from the external regulator (dI_{EXT}/dt) is limited during the Firmware start-up phase (T_2 upto T_3) to a maximum of 100 mA/100 μ s.
- PORST is active/asserted when either PORST (input) or PORST (output) is active/asserted.
- $\overline{\text{PORST}}$ (input) active means that the reset is held active by external agents by pulling the PORST pin low. It is required to keep the $\overline{\text{PORST}}$ (input) asserted for a minimum duration after the external supply is above the respective primary reset threshold as documented in datasheet.
- $\overline{\text{PORST}}$ (output) active means that μ C asserts the reset internally and drives the $\overline{\text{PORST}}$ pin low thus propagating the reset to external devices. The $\overline{\text{PORST}}$ (output) is asserted by the μ C when atleast one among the three supply domains (1.3 V, 3.3 V or 5 V) violate their primary under-voltage reset thresholds. The $\overline{\text{PORST}}$ (output) is deasserted by the μ C when all supplies are above their primary reset thresholds and the basic supply and clock infrastructure is available.
- The power sequence as shown in **Figure 7-33** is enumerated below
 - T_1 refers to the point in time when basic supply and clock infrastructure is available as the external supply ramps up. The supply mode is evaluated based on the HWCFG[0:2,6] pins and consequently a soft start of EVR13 and EVR33 regulators are initiated.
 - T_2 refers to the point in time when all supplies are above their primary reset thresholds. EVR13 and EVR33 regulators have ramped up. $\overline{\text{PORST}}$ (output) is deasserted and HWCFG[3:5] pins are latched on $\overline{\text{PORST}}$ rising edge. Firmware execution is initiated.
 - T_3 refers to the point in time when Firmware execution is completed. User code execution starts with a default frequency of 100 MHz.
 - T_4 refers to the point in time during the Ramp-down phase when atleast one of the externally provided or generated supplies (1.3 V, 3.3 V or 5 V) drop below their respective primary under-voltage reset thresholds.

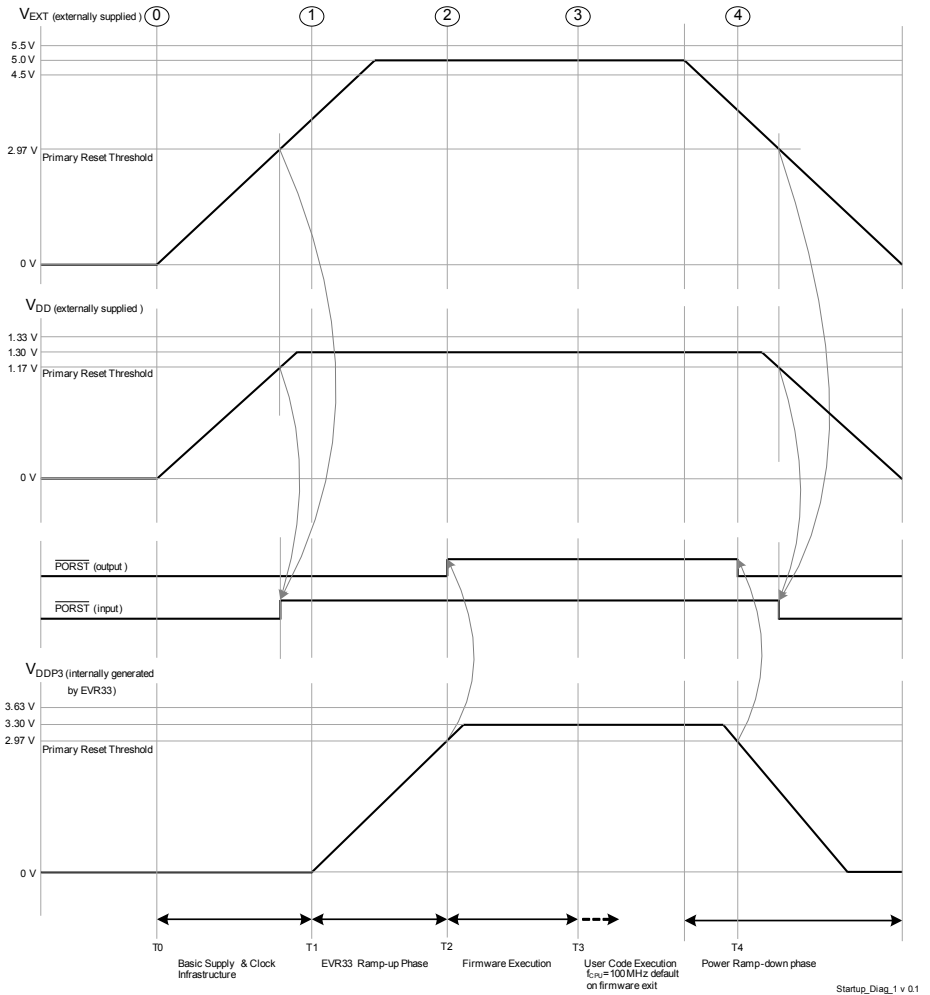


Figure 7-34 External Supply mode (d) - 5 V and 1.3 V externally supplied

External Supply mode (d)

5 V and 1.3 V supplies are externally supplied. 3.3V is generated internally by the EVR33 regulator.

- External supplies V_{EXT} and V_{DD} may ramp-up or ramp-down independent of each other with regards to start, rise and fall time(s). EVRx3 is also robust against a voltage ramp-up starting from a residual voltage between 0-1 V. Start-up slew rates for supply rails should comply to datasheet values.
- The rate at which current is drawn from the external regulator (di_{EXT}/dt or di_{DD}/dt) is limited in the Start-up phase to a maximum of 100 mA/100 us.
- \overline{PORST} is active/asserted when either \overline{PORST} (input) or \overline{PORST} (output) is active/asserted.
- \overline{PORST} (input) active means that the reset is held active by external agents by pulling the \overline{PORST} pin low. It is required to keep the \overline{PORST} (input) asserted for a minimum duration after the external supply is above the respective primary reset threshold as documented in datasheet.
- \overline{PORST} (output) active means that μC asserts the reset internally and drives the \overline{PORST} pin low thus propagating the reset to external devices. The \overline{PORST} (output) is asserted by the μC when atleast one among the three supply domains (1.3 V, 3.3 V or 5 V) violate their primary under-voltage reset thresholds. The \overline{PORST} (output) is deasserted by the μC when all supplies are above their primary reset thresholds and the basic supply and clock infrastructure is available.
- The power sequence as shown in [Figure 7-34](#) is enumerated below
 - T1 refers to the point in time when basic supply and clock infrastructure is available as the external supplies ramp up. The supply mode is evaluated based on the HWCFG[0:2,6] pins and consequently a soft start of EVR33 regulator is initiated.
 - T2 refers to the point in time when all supplies are above their primary reset thresholds. EVR33 regulator has ramped up. \overline{PORST} (output) is deasserted and HWCFG[3:5] pins are latched on \overline{PORST} rising edge. Firmware execution is initiated.
 - T3 refers to the point in time when Firmware execution is completed. User code execution starts with a default frequency of 100 MHz.
 - T4 refers to the point in time during the Ramp-down phase when atleast one of the externally provided or generated supplies (1.3 V, 3.3 V or 5 V) drop below their respective primary under-voltage reset thresholds.

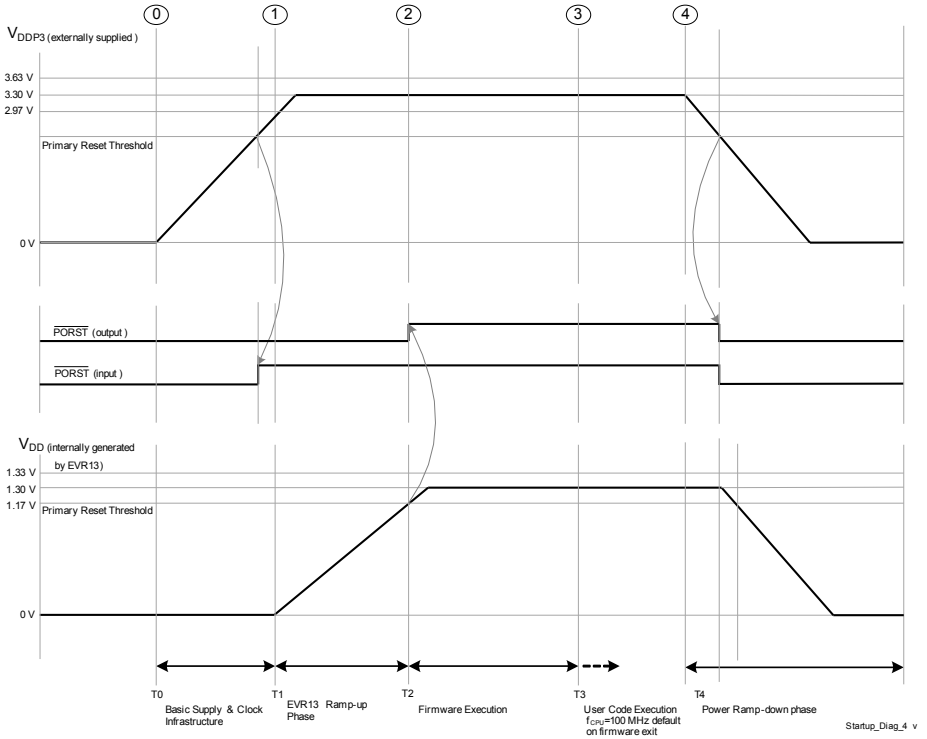


Figure 7-35 Single Supply mode (e & f) - 3.3 V single supply

Single Supply mode (e & f)

3.3 V single supply mode. 1.3 V is generated internally by the EVR13 regulator.

- The rate at which current is drawn from the external regulator (di_{EXT}/dt) is limited in the Start-up phase to a maximum of 100 mA/100 μ s. EVR13 is also robust against a voltage ramp-up starting from a residual voltage between 0-1 V. Start-up slew rates for supply rails should comply to datasheet values.
- \overline{PORST} is active/asserted when either \overline{PORST} (input) or \overline{PORST} (output) is active/asserted.
- \overline{PORST} (input) active means that the reset is held active by external agents by pulling the \overline{PORST} pin low. It is required to keep the \overline{PORST} (input) asserted for a minimum duration after the external supply is above the respective primary reset threshold as documented in datasheet.
- \overline{PORST} (output) active means that μ C asserts the reset internally and drives the \overline{PORST} pin low thus propagating the reset to external devices. The \overline{PORST} (output) is asserted by the μ C when at least one among the three supply domains (1.3 V or 3.3 V) violate their primary under-voltage reset thresholds. The \overline{PORST} (output) is deasserted by the μ C when all supplies are above their primary reset thresholds and the basic supply and clock infrastructure is available.
- The power sequence as shown in [Figure 7-35](#) is enumerated below
 - T1 refers to the point in time when basic supply and clock infrastructure is available as the external supply ramps up. The supply mode is evaluated based on the HWCFG[0:2,6] pins and consequently a soft start of EVR13 regulator is initiated.
 - T2 refers to the point in time when all supplies are above their primary reset thresholds. EVR13 regulator has ramped up. \overline{PORST} (output) is deasserted and HWCFG[3:5] pins are latched on \overline{PORST} rising edge. Firmware execution is initiated.
 - T3 refers to the point in time when Firmware execution is completed. User code execution starts with a default frequency of 100 MHz.
 - T4 refers to the point in time during the Ramp-down phase when at least one of the externally provided or generated supplies (1.3 V or 3.3 V) drop below their respective primary under-voltage reset thresholds.

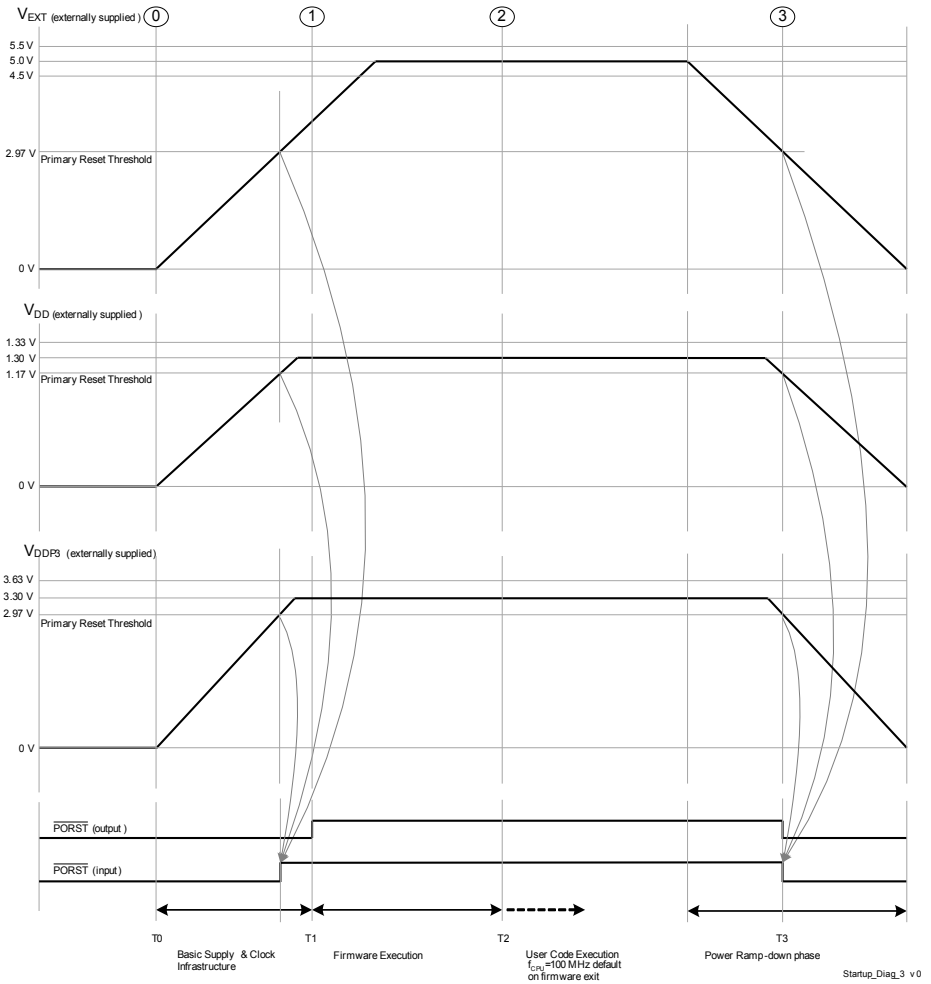


Figure 7-36 External Supply mode (h) - 5 V, 3.3 V & 1.3 V externally supplied

External Supply mode (h)

All supplies, namely 5 V, 3.3 V & 1.3 V, are externally supplied.

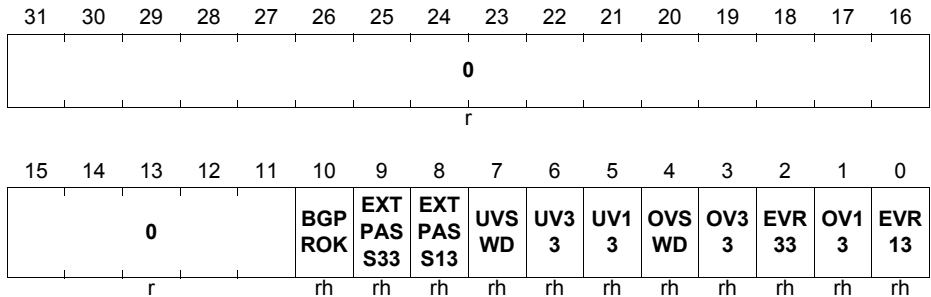
- External supplies V_{EXT} , V_{DDP3} & V_{DD} may ramp-up or ramp-down independent of each other with regards to start, rise and fall time(s). The supply system is also robust against a voltage ramp-up starting from a residual voltage between 0 - 1 V. Start-up slew rates for supply rails should comply to datasheet values.
- The rate at which current is drawn from the external regulator (di_{EXT}/dt , di_{DD}/dt or di_{DDP3}/dt) is limited in the Start-up phase to a maximum of 100 mA/100 μ s.
- \overline{PORST} is active/asserted when either \overline{PORST} (input) or \overline{PORST} (output) is active/asserted.
- \overline{PORST} (input) active means that the reset is held active by external agents by pulling the \overline{PORST} pin low. It is required to keep the \overline{PORST} (input) asserted for a minimum duration after the external supply is above the respective primary reset threshold as documented in datasheet.
- \overline{PORST} (output) active means that μ C asserts the reset internally and drives the \overline{PORST} pin low thus propagating the reset to external devices. The \overline{PORST} (output) is asserted by the μ C when atleast one among the three supply domains (1.3 V, 3.3 V or 5 V) violate their primary under-voltage reset thresholds. The \overline{PORST} (output) is deasserted by the μ C when all supplies are above their primary reset thresholds and the basic supply and clock infrastructure is available.
- The power sequence as shown in [Figure 7-36](#) is enumerated below
 - T1 refers to the point in time when all supplies are above their primary reset thresholds and basic clock infrastructure is available. The supply mode is evaluated based on the HWCFG[0:2,6] pins. \overline{PORST} (output) is deasserted and HWCFG[3:5] pins are latched on \overline{PORST} rising edge. Firmware execution is initiated.
 - T2 refers to the point in time when Firmware execution is completed. User code execution starts with a default frequency of 100 MHz.
 - T3 refers to the point in time during the Ramp-down phase when atleast one of the externally provided supplies (1.3 V, 3.3 V or 5 V) drop below their respective primary under-voltage reset thresholds.

7.3.1.10 EVR Control Registers

All EVR registers with LCK bits have corresponding shadow registers in the EVR Standby domain. When writing consecutive EVR registers with LCK bits, it need to be ensured that the consecutive register is written only when the respective LCK bit is in unlocked state. This is because all EVR shadow registers are addressed using a shared bus across multiple clock domains and power isolation. An ongoing write will block the complete bus inhibiting parallel or consecutive writes on other registers until LCK bit is released. In a multi- CPU system, it might be better to read back the value from the register to ensure that it is written before proceeding writing to the next register. After a cold $\overline{\text{PORST}}$, warm and system resets these registers may return a value 0 or the updated value by the Firmware. The reset value reflects the default isolation value in the shadow register. Otherwise, a read will provide the value of the most recent write operation. The status registers **EVRSTAT** and **EVRADCSTAT** are updated during Start-up and after every EVR measurement cycle and the reset value is dependent on the EVR topology. Therefore the value read from register may differ from the documented reset value.

EVRSTAT

EVR Status Register (0B0_H) **Reset Value: 0000 0400_H**



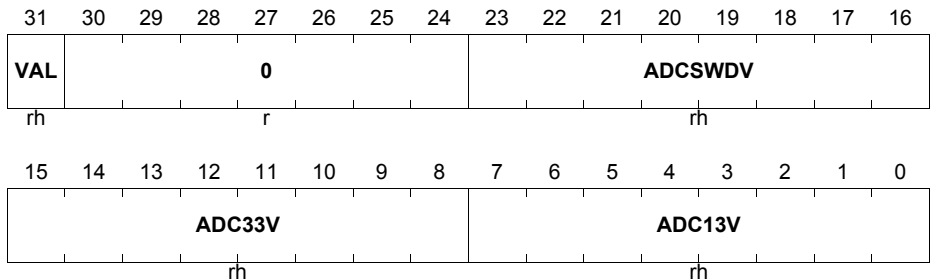
Field	Bits	Type	Description
EVR13	0	rh	EVR13 status This bit is set if the internal EVR13 LDO or SMPS regulator is active. EVR13 is activated if HWCFG[2] pin level is latched high during start-up phase. 0 _B EVR13 is inactive. 1 _B EVR13 is active.

Field	Bits	Type	Description
OV13	1	rh	<p>EVR13 Regulator Over-voltage event flag This bit is set if EVR13 secondary voltage monitor recognises a 1.3 V over-voltage event. An alarm is raised to the SMU.</p> <p>0_B No over-voltage condition happened. 1_B EVR13 Over-voltage event indication as configured in EVROVMON register.</p>
EVR33	2	rh	<p>EVR33 status This bit is set if the internal EVR33 LDO regulator is active. EVR33 is activated if HWCFG[1] pin level is latched high during start-up phase.</p> <p>0_B EVR33 is inactive. 1_B EVR33 is active.</p>
OV33	3	rh	<p>EVR33 Regulator Over-voltage event flag This bit is set if EVR33 secondary voltage monitor recognises a 3.3 V over-voltage event. An alarm is raised to the SMU.</p> <p>0_B No over-voltage condition happened. 1_B EVR33 Over-voltage event indication as configured in EVROVMON register.</p>
OVSWD	4	rh	<p>Supply Watchdog (SWD) Over-voltage event flag This bit is set if VEXT secondary voltage monitor recognises an over-voltage event. An alarm is raised to the SMU.</p> <p>0_B No over-voltage condition happened. 1_B SWD Over-voltage event indication as configured in EVROVMON register.</p>
UV13	5	rh	<p>EVR13 Regulator Under-voltage event flag This bit is set if EVR13 secondary voltage monitor recognises a 1.3 V under-voltage event. An alarm is raised to the SMU.</p> <p>0_B No under-voltage condition happened. 1_B EVR13 Under-voltage event indication as configured in EVRUVMON register.</p>

Field	Bits	Type	Description
UV33	6	rh	<p>EVR33 Regulator Under-voltage event flag</p> <p>This bit is set if EVR33 secondary voltage monitor recognises a 3.3 V under-voltage event. An alarm is raised to the SMU.</p> <p>0_B No under-voltage condition happened. 1_B EVR33 Under-voltage event indication as configured in EVRUVMON register.</p>
UVSWD	7	rh	<p>Supply Watchdog (SWD) Under-voltage event flag</p> <p>This bit is set if VEXT secondary voltage monitor recognises an under-voltage event. An alarm is raised to the SMU.</p> <p>0_B No under-voltage condition happened. 1_B SWD Under-voltage event indication as configured in EVRUVMON register.</p>
EXTPASS13	8	rh	<p>External Pass Device for EVR13</p> <p>This bit is set if an external device is detected at VGATE1P pin for EVR13 regulator in LDO or SMPS mode. If VGATE1P pin is externally connected to ground, then the bit is cleared. If the pin is left open or sinking of current is not possible via the pin, the bit is set.</p> <p>0_B No external pass device used 1_B External pass device is used for EVR13 LDO or external charge device is used for EVR13 DC-DC regulator</p>
EXTPASS33	9	rh	<p>External Pass Device for EVR33</p> <p>This bit is set if an external pass device is detected at VGATE3P pin for EVR33 regulator in LDO mode. In case VGATE3P pin is absent in the package, then this signal is internally connected to ground and the bit is cleared.</p> <p>0_B No external pass device used 1_B External pass device used for EVR33 LDO</p>

Field	Bits	Type	Description
BGPROK	10	rh	Primary Bandgap status This bit is set after any reset and indicates that the primary bandgap voltage is in operational range. In case primary bandgap is not ok, we remain in cold reset state. On a Cold PORST release, this bit would be set to 1. 0_B Primary bandgap not ok. 1_B Primary bandgap is ok.
0	[31:11]	r	Reserved Read as 0.

The over-voltage and under-voltage status signals are reported to SMU.EMM unit. An alarm for the upper and lower bound is supported in the SMU.EMM unit.

EVRADCSTAT
EVR ADC Status Register
(19C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
ADC13V	[7:0]	rh	ADC 1.3 V Conversion Result This bit field contains the last conversion result of the ADC measurement of the $V_{DD} / EVR13$ supply. $VIN = [LSB * (ADC13V-1)]$; $LSB = 5.7692 \text{ mV}$ Eg. $1.30 \text{ V} - E2_H - 226_D$ (refer datasheet)
ADC33V	[15:8]	rh	ADC 3.3 V Conversion Result This bit field contains the last conversion result of the ADC measurement of the $V_{DDP3} / EVR33$ supply. $VIN = [LSB * (ADC33V-1)]$; $LSB = 23.077 \text{ mV}$ Eg. $3.30 \text{ V} - 90_H - 144_D$ (refer datasheet)

Field	Bits	Type	Description
RST13TRIM	[7:0]	rw	<p>1.3 V Regulator Reset Trim Value</p> <p>This bit field selects the hard reset generation level of the EVR13 regulator. Nominal programmed value is 5 LSB lower to take care of local noise and on-chip drops to sense point. Tolerance is documented in datasheet as VDDPRIUV parameter.</p> <p>1.170 V – C5_H – 197_D ~ -10% at VDD pin 1.203 V – CB_H – 203_D ~ -7.5% at VDD pin 1.235 V – D0_H – 208 ~ -5% at VDD pin Threshold = [(VIN / LSB) - 5] LSB = 5.7692 mV</p>
RST13OFF	24	rw	<p>EVR13 Reset Enable</p> <p>0_B A reset trigger signal is generated and forwarded to the SCU by the EVR13 block depending on the selected reset trim value.</p> <p>1_B No reset trigger signal is generated and forwarded to the SCU by the EVR13 block depending on the selected reset trim value.</p> <p>This bit can only be changed if bit BPRST13OFF is set in parallel.</p>
BPRST13OFF	25	w	<p>Bit Protection RST13OFF</p> <p>Setting this bit enables that bit RST13OFF can be changed in this write operation.</p> <p>This bit is read as zero.</p>
RST33OFF	26	rw	<p>EVR33 Reset Enable</p> <p>0_B A reset trigger is generated and forwarded to the SCU by the EVR33 primary voltage monitor.</p> <p>1_B No reset trigger is generated by the EVR33 primary voltage monitor.</p> <p>This bit can only be changed if bit BPRST33OFF is set in parallel. The EVR33 reset is disabled by application to support voltage drop up to nominal 3.0 V during cranking.</p>
BPRST33OFF	27	w	<p>Bit Protection RST33OFF</p> <p>Setting this bit enables that bit RST33OFF can be changed in this write operation.</p> <p>This bit read also as zero.</p>

Field	Bits	Type	Description
RSTSWDOFF	28	rw	<p>EVR SWD Reset Enable</p> <p>0_B A reset trigger signal is generated and forwarded to the SCU by the external supply monitoring block depending on the selected reset trim value.</p> <p>1_B No reset trigger signal is generated and forwarded to the SCU by the external supply monitoring block depending on the selected reset trim value.</p> <p>This bit can only be changed if bit BPRSTSWDOFF is set in parallel.</p>
BPRSTSWDOFF	29	w	<p>Bit Protection RSTSWDOFF</p> <p>Setting this bit enables that bit RSTSWDOFF can be changed in this write operation.</p> <p>This bit read also as zero.</p>
RES	[23:8]	r	<p>Reserved</p> <p>These bits should not be written.</p> <p>Writes to other bitfields must ensure that these bits are not changed.</p>
LCK	31	rh	<p>Lock Status</p> <p>This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p>0_B The register is unlocked and can be updated</p> <p>1_B The register is locked and cannot be updated</p>
0	30	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

EVROVMON
EVR Over-voltage Configuration Register(1A4_H)
Reset Value: 00F1 9EFD_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	0			0							SWDOVVAL				
rh	r			r							rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EVR33OVVAL								EVR13OVVAL							
rw								rw							

Field	Bits	Type	Description
EVR13OVVAL	[7:0]	rw	1.3 V Regulator Over-voltage threshold This field defines the over-voltage monitoring threshold level of the EVR13 regulator. Tolerance is documented in datasheet as VDDMON parameter. $1.30\text{ V} - E_{2H} - 226_D$ (refer datasheet) Threshold = $[(VIN / LSB) + 1]$ LSB = 5.7692 mV
EVR33OVVAL	[15:8]	rw	3.3 V Regulator Over-voltage threshold This field defines the over-voltage monitoring threshold level of the EVR33 regulator. Tolerance is documented in datasheet as VDDP3MON parameter. $3.29\text{ V} - 90_H - 144_D$ (refer datasheet) Threshold = $[(VIN / LSB) + 1]$ LSB = 23.077 mV
SWDOVVAL	[23:16]	rw	Supply monitor (SWD) Over-voltage threshold value This field defines the over-voltage threshold level of the external VEXT supply monitor. Tolerance is documented in datasheet as VEXTMON parameter. 5 V external supply $5.01\text{ V} - DB_H - 219_D$ (refer datasheet) 3.3 V external supply $3.29\text{ V} - 90_H - 144_D$ (refer datasheet) Threshold = $[(VIN / LSB) + 1]$ LSB = 23.077 mV

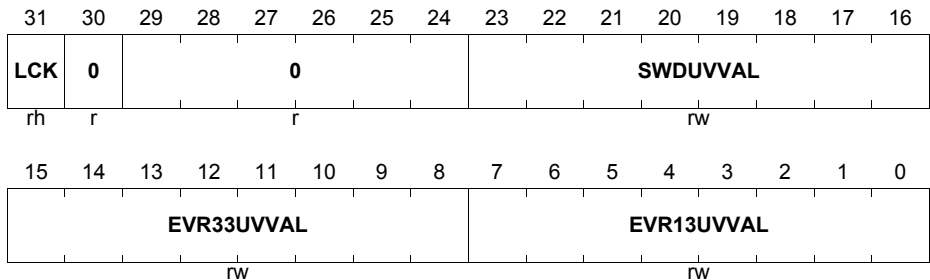
Field	Bits	Type	Description
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0_B The register is unlocked and can be updated 1_B The register is locked and cannot be updated
0	[29:24], 30	r	Reserved Read as 0; should be written with 0.

A configurable threshold with upper and lower voltage bounds can be defined in EVROVMON and EVRUVMON registers for monitoring EVR13 and EVR33 regulators. It need to be considered that after a system reset, Firmware overwrites the register values.

The EVROVMON and EVRUVMON registers can be configured to trigger SMU alarms via software. It is also possible to activate alarms via software in the SMU unit.

EVROVMON

EVR Under-voltage Configuration Register(1A0_H) **Reset Value: 00D0 84CD_H**



Field	Bits	Type	Description
EVR13UVVAL	[7:0]	rw	1.3 V Regulator Under-voltage threshold This field defines the under-voltage monitoring threshold level of the EVR13 regulator. $1.30\text{ V} - E_{2H} - 226_D$ (refer datasheet) Threshold = $[(VIN / LSB) + 1]$ LSB = 5.7692 mV

Field	Bits	Type	Description
EVR33UVVAL	[15:8]	rw	3.3 V Regulator Under-voltage threshold This field defines the under-voltage monitoring threshold level of the EVR33 regulator. $2.99\text{ V} - 83_{\text{H}} - 131_{\text{D}} \sim -10\%$ (refer datasheet) $\text{Threshold} = \lceil (\text{VIN} / \text{LSB}) + 1 \rceil$ $\text{LSB} = 23.077\text{ mV}$
SWDUVVAL	[23:16]	rw	Supply monitor (SWD) Under-voltage threshold value This field defines the under-voltage threshold level of the external VEXT supply monitor. 5 V external supply $5.01\text{ V} - \text{DB}_{\text{H}} - 219_{\text{D}}$ (refer datasheet) 3.3 V external supply $3.29\text{ V} - 90_{\text{H}} - 144_{\text{D}}$ (refer datasheet) $\text{Threshold} = \lceil (\text{VIN} / \text{LSB}) + 1 \rceil$ $\text{LSB} = 23.077\text{ mV}$
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0_{B} The register is unlocked and can be updated 1_{B} The register is locked and cannot be updated
0	[29:24], 30	r	Reserved Read as 0; should be written with 0.

EVRMONCTRL
EVR Monitor Control Register
(1A8_H)
Reset Value: 0021 2121_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SLC K				0			0		SWDUVM OD		0		SWDOVM OD	
r	rwh				r			r		rw		r		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		EVR33UV MOD		0		EVR33OV MOD		0		EVR13UV MOD		0		EVR13OV MOD	
r		rw		r		rw		r		rw		r		rw	

Field	Bits	Type	Description
EVR13OVMOD	[1:0]	rw	<p>1.3 V Regulator Over-voltage monitoring mode</p> <p>00_B Over-voltage monitoring does not trigger SMU alarm events but violation indicated in EVRSTAT register.</p> <p>01_B An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition</p> <p>10_B An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition</p> <p>11_B An over-voltage event is triggered when the threshold is crossed in either direction</p>
EVR13UVMOD	[5:4]	rw	<p>1.3 V Regulator Under-voltage monitoring mode</p> <p>00_B Under-voltage monitoring does not trigger SMU alarm events but violation indicated in EVRSTAT register.</p> <p>01_B An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition</p> <p>10_B An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition</p> <p>11_B An under-voltage event is triggered when the threshold is crossed in either direction</p>
EVR33OVMOD	[9:8]	rw	<p>3.3 V Regulator Over-voltage monitoring mode</p> <p>00_B Over-voltage monitoring does not trigger SMU alarm events but violation indicated in EVRSTAT register.</p> <p>01_B An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition</p> <p>10_B An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition</p> <p>11_B An over-voltage event is triggered when the threshold is crossed in either direction</p>

Field	Bits	Type	Description
EVR33UVMOD	[13:12]	rw	3.3 V Regulator Under-voltage monitoring mode 00 _B Under-voltage monitoring does not trigger SMU alarm events but violation indicated in EVRSTAT register. 01 _B An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition 10 _B An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition 11 _B An under-voltage event is triggered when the threshold is crossed in either direction
SWDOVMOD	[17:16]	rw	Supply monitor (SWD) Over-voltage monitoring mode 00 _B Over-voltage monitoring does not trigger SMU alarm events but violation indicated in EVRSTAT register. 01 _B An over-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition 10 _B An over-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition 11 _B An over-voltage event is triggered when the threshold is crossed in either direction
SWDUVMOD	[21:20]	rw	Supply monitor (SWD) Under-voltage monitoring mode 00 _B Under-voltage monitoring does not trigger SMU alarm events but violation indicated in EVRSTAT register. 01 _B An under-voltage event is triggered when the threshold is crossed in a lower to higher voltage transition 10 _B An under-voltage event is triggered when the threshold is crossed in a higher to lower voltage transition 11 _B An under-voltage event is triggered when the threshold is crossed in either direction

Field	Bits	Type	Description
0	2,3,6,7, 10,11, 14,15, 18,19, 23,22, 30,31	r	Reserved Read as 0; should be written with 0.
0	[29:24]	r	Reserved Read as 0; should be written with 0.

The default setting after reset is that over-voltage indication is notified via an SMU alarm when the overvoltage threshold is crossed in a lower to higher voltage transition.

The default setting after reset is that under-voltage indication is notified via an SMU alarm when the under-voltage threshold is crossed in a higher to lower voltage transition.

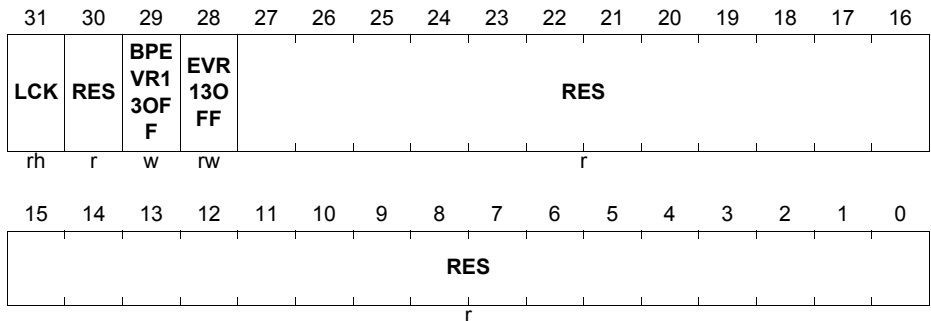
It can be configured in EVRMONCTRL register to generate an interrupt when the over-under-voltage thresholds are crossed in either direction. This may be used to notify when the violation condition disappears with respect to secondary voltage monitoring.

EVR13CON

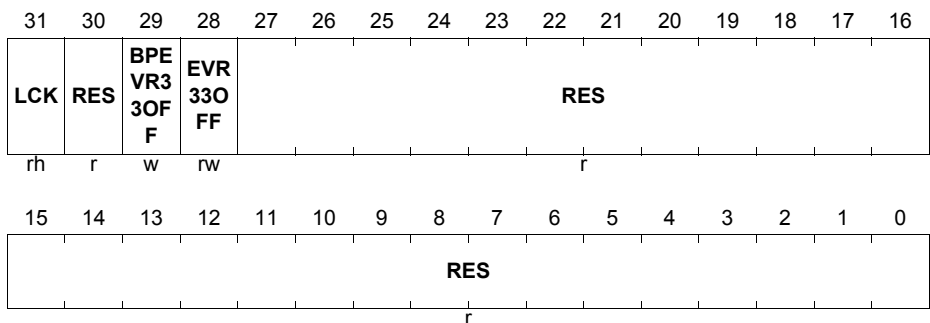
EVR13 Control Register

(0B8_H)

Reset Value: 0380 0380_H



Field	Bits	Type	Description
EVR13OFF	28	rw	EVR13 Regulator Enable 0_B The EVR13 regulator module is enabled 1_B The EVR13 regulator module is disabled/switched off. The device may be operating in standby or with external 1.3V supply. This bit can only be changed if bit BPEVR13OFF is set in parallel.
BPEVR13OFF	29	w	Bit Protection EVR13OFF Setting this bit enables that bit EVR13OFF can be changed in this write operation. This bit is read as zero.
RES	30, [27:0]	r	Reserved These bits should not be written. Writes to other bitfields shall ensure that these bits are not changed.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0_B The register is unlocked and can be updated 1_B The register is locked and cannot be updated

EVR33CON
EVR33 Control Register
(0BC_H)
Reset Value: 0780 0780_H


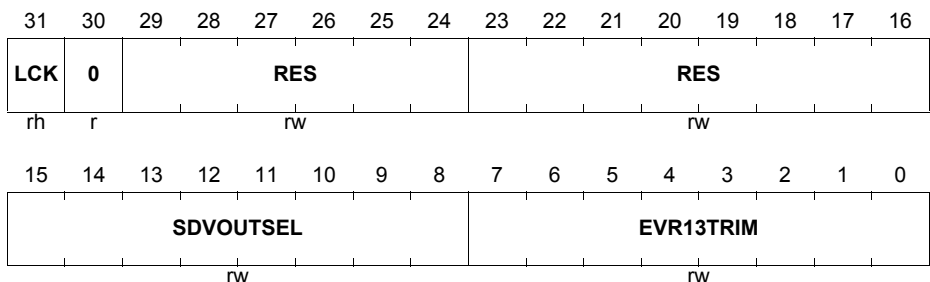
Field	Bits	Type	Description
EVR33OFF	28	rw	<p>EVR33 Regulator Enable</p> <p>0_B The EVR33 regulator module is enabled</p> <p>1_B The EVR33 regulator module is disabled/switched off. The device may be operating in standby or with external 3.3V supply.</p> <p>This bit can only be changed if bit BPEVR33OFF is set in parallel.</p>
BPEVR33OFF	29	w	<p>Bit Protection EVR33OFF</p> <p>Setting this bit enables that bit EVR33OFF can be changed in this write operation.</p> <p>This bit is read as zero.</p>
RES	30, [27:0]	r	<p>Reserved</p> <p>These bits should not be written.</p> <p>Writes to other bitfields shall ensure that these bits are not changed.</p>
LCK	31	rh	<p>Lock Status</p> <p>This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect.</p> <p>0_B The register is unlocked and can be updated</p> <p>1_B The register is locked and cannot be updated</p>

EVRTRIM

EVR Trim Register

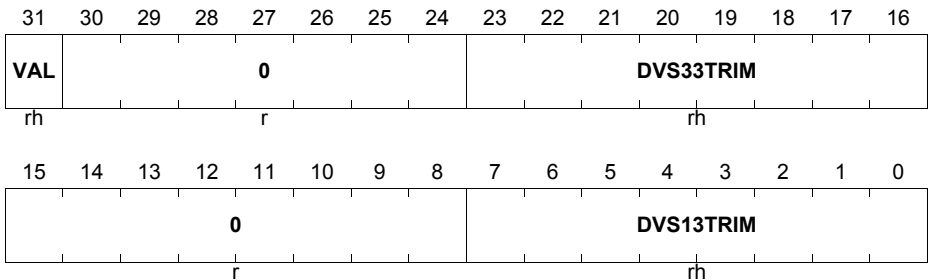
(198_H)

Reset Value: 008F 73E4_H



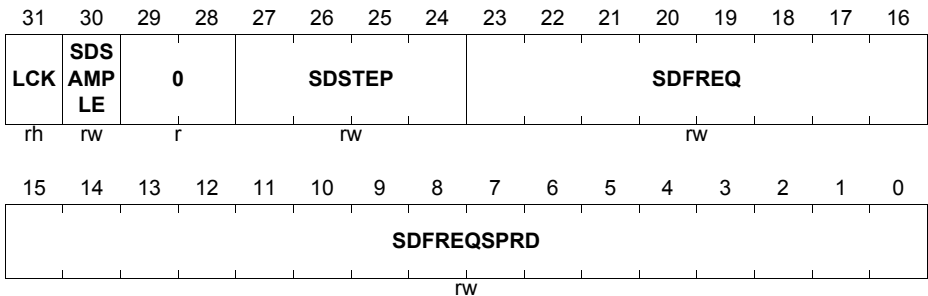
Field	Bits	Type	Description
EVR13TRIM	[7:0]	rw	1.3 V Regulator Voltage Trim Value This bit field defines the output voltage of the EVR13. $1.3\text{ V} - E2_{\text{H}} - 226_{\text{D}}$ $1.33\text{ V} - E7_{\text{H}} - 231_{\text{D}} (+2.5\%)$ Set Point = $[(V_{\text{IN}} / \text{LSB}) + 1]$ LSB = 5.7692 mV
SDVOUTSEL	[15:8]	rw	SD Regulator Voltage selection The V_{DD} output level selection for the Step down regulator. $1.3\text{ V} - 73_{\text{H}} - 115_{\text{D}}$ $1.33\text{ V} - 79_{\text{H}} - 121_{\text{D}} (+2.5\%)$ Set Point = $[(V_{\text{IN}} - 725\text{ mV}) / \text{LSB}]$ LSB = 5 mV
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0_{B} The register is unlocked and can be updated 1_{B} The register is locked and cannot be updated
RES	30,[29:16]	r	Reserved These bits should not be written. Writes to other bitfields must ensure that these bits are not changed.

EVRTRIM and EVRRSTCON register may be used to generate voltage stress conditions to subject the modules to voltages beyond normal operating ranges.

EVDRVSTAT
EVR Status Register for Voltage Scaling(0B4_H)
Reset Value: 0000 0000_H


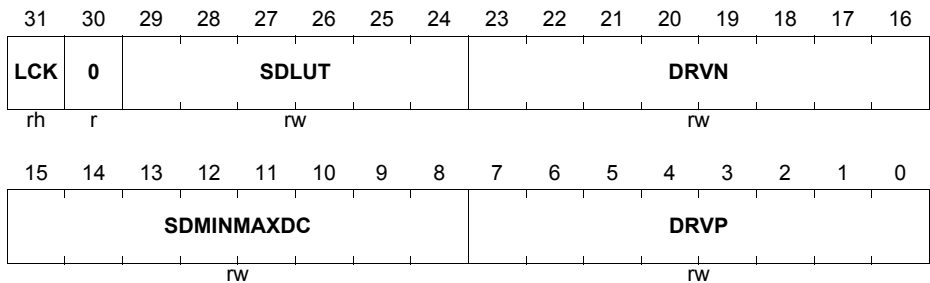
Field	Bits	Type	Description
DVS13TRIM	[7:0]	rh	1.3 V Regulator Voltage Trim Status EVR13 Setpoint trim value currently used by EVR as was configured in EVRTRIM.EVR13TRIM register bits.
DVS33TRIM	[23:16]	rh	3.3 V Regulator Voltage Trim Status EVR33 Setpoint Trim value currently used by EVR.
VAL	31	rh	Valid Status This bit indicates if the register is being updated with a new value and indicates the start and end of a transfer. Values are updated in a single cycle and therefore consistent within the register irrespective of the VAL bit value. 0 _B The register is not being updated 1 _B The register is being updated
0	[15:8], [30:24]	r	Reserved Read as 0; should be written with 0.

EVRSCTRL1, EVRSCTRL2, EVRSCTRL3, EVRSCTRL4, EVRSDCOEFF1, EVRSDCOEFF2, EVRSDCOEFF3, EVRSDCOEFF4, EVRSDCOEFF5 and EVRSDCOEFF6 SMPS registers have different reset values between EVR and SCU. The firmware updates the registers with values from configuration sector at start-up.

EVRSDCTRL1
EVR13 SD Control Register 1
(1B0_H)
Reset Value: 0940 051F_H


Field	Bits	Type	Description
SDFREQSPRD	[15:0]	rw	<p>Frequency Spread Threshold</p> <p>This bit field defines the frequency spread of the nominal SMPS regulator switching frequency.</p> <p>Frequency spread % = (SDFREQSPRD / FFFF_H)% 0000_H 0% frequency spread 028F_H 1% frequency spread 051F_H 2% frequency spread</p>
SDFREQ	[23:16]	rw	<p>Regulator Switching Frequency</p> <p>The SMPS regulator switching frequency is configurable between 0.4 MHz upto 2.0 MHz. The switching frequency is equal to (100 MHz / SDFREQ) value.</p> <p>7D_H 0.8 MHz (100 MHz/125) switching frequency 64_H 1.0 MHz (100 MHz/100) switching frequency 40_H 1.56 MHz (100 MHz/64) switching frequency 37_H 1.82 MHz (100 MHz/55) switching frequency</p> <p>After start-up, the SMPS regulator runs with 1.5625 MHz by default. Later it may be changed by application to suit EMI/EMC and component needs. The switching frequency is derived from the internal 100 MHz clock source.</p>
SDSTEP	[27:24]	rw	<p>Droop Voltage Step</p> <p>This bit field defines the voltage offset for droop compensation on a load jump. The voltage offset applied is equal to the SDSTEP x 5 mV.</p>

Field	Bits	Type	Description
SDSAMPLE	30	rw	ADC Sampling Scheme Selection of the output voltage sampling scheme as once or twice during a single switching period. 0 _B Single sampling scheme. (default) 1 _B Double sampling scheme.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and cannot be updated
0	[29:28]	r	Reserved Read as 0; should be written with 0.

EVRSDCTRL2
EVR13 SD Control Register 2
(1B_{4H})
Reset Value: 15CC 06CC_H


Field	Bits	Type	Description
DRVP	[7:0]	rw	P-Driver Setting Configuration of the external high side driver. (P-channel MOSFET).
SDMINMAXDC	[15:8]	rw	Minimum Duty Cycle The SMPS regulator duty cycle is limited to a minimum Ton / Toff in 100 MHz cycles to ensure proper commutation of the switches.

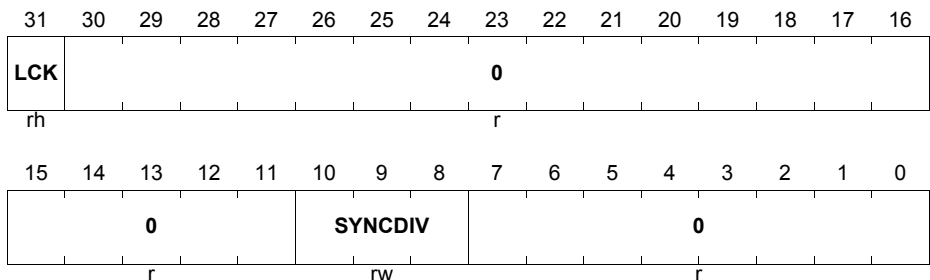
Field	Bits	Type	Description
DRVN	[23:16]	rw	N-Driver Setting Configuration of the external low side driver. (N-channel MOSFET).
SDLUT	[29:24]	rw	Non-linear Starting Point Non linear slope setting.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and cannot be updated
0	30	r	Reserved Read as 0; should be written with 0.

EVRSDCTRL3
EVR13 SD Control Register 3
(1B8_H)
Reset Value: 0009 0A3C_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK		0				SDVOKLVL									
rh		r				rw									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDPID								SDPWMPRE							
rw								rw							

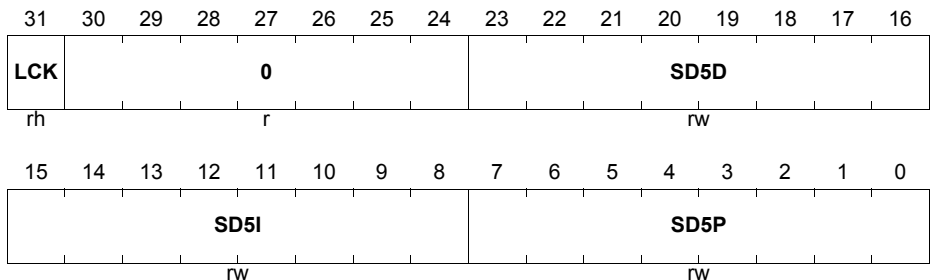
Field	Bits	Type	Description
SDPWMPRE	[7:0]	rw	PWM Preset Value Initial dutycycle value at startup calculated as (SDFREQ -SDPRWMPRE) / SDFREQ. This influence the initial coil current peak. 5A _H 10% [(100-90)/100] dutycycle for 1 MHz 36 _H 15% [(64-54)/64] dutycycle for 1,5 MHz 30 _H 13% [(55-48)/55] dutycycle for 1,8 MHz

Field	Bits	Type	Description
SDPID	[15:8]	rw	PID Control to be defined.
SDVOKLVL	[23:16]	rw	Configuration of Voltage OK Signal Filter and threshold configuration of the Voltage OK signal at startup.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and cannot be updated
0	[30:24]	r	Reserved Read as 0; should be written with 0.

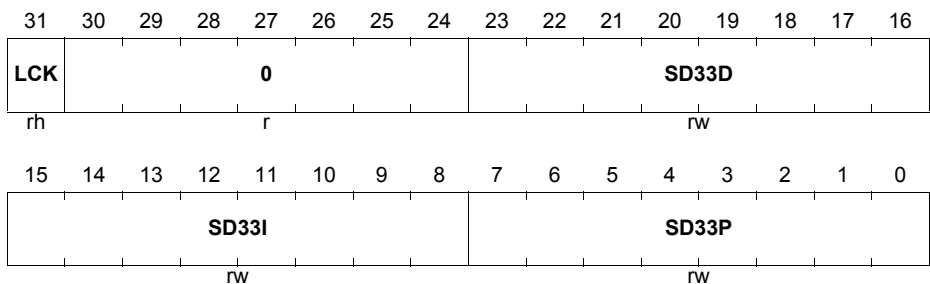
EVRSDCTRL4
EVR13 SD Control Register 4
(1BC_H)
Reset Value: 0000 1001_H


Field	Bits	Type	Description
0	[7:0]	r	Reserved

Field	Bits	Type	Description
SYNCDIV	[10:8]	rw	Clock Divider Ratio for external DCDC SYNC signal This bitfield defines the divider factor for the clock signal to synchronise external SMPS regulator to the internal SMPS EVR13 regulator. $000_B f_{DCDCSYNC} = f_{DCDC}$ $001_B f_{DCDCSYNC} = f_{DCDC}/2$ $010_B f_{DCDCSYNC} = f_{DCDC}/4$ $011_B f_{DCDCSYNC} = f_{DCDC}/8$ $100_B f_{DCDCSYNC} = f_{DCDC}/16$ $101_B f_{DCDCSYNC} = f_{DCDC}/32$ All other combinations are reserved.
0	[30:11]	r	Reserved Read as 0; should be written with 0.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0_B The register is unlocked and can be updated 1_B The register is locked and cannot be updated

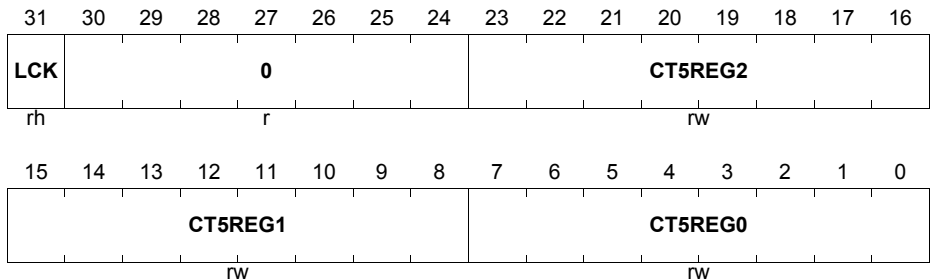
EVRSDCOEFF1
EVR13 SD Coefficient Register 1
(1C0_H)
Reset Value:0069 6D6C_H


Field	Bits	Type	Description
SD5P	[7:0]	rw	P Coefficient P control parameter for the PID regulator (5V). Bits [7:4] - Mantissa, Bits [3:0] - Exponent.
SD5I	[15:8]	rw	I Coefficient I control parameter for the PID regulator (5V). Bits [7:4] - Mantissa, Bits [3:0] - Exponent.
SD5D	[23:16]	rw	D Coefficient D control parameter for the PID regulator (5V). Bits [7:4] - Mantissa, Bits [3:0] - Exponent.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and cannot be updated
0	[30:24]	r	Reserved Read as 0; should be written with 0.

EVRSDCOEFF2
EVR13 SD Coefficient Register 2
(1C4_H)
Reset Value:0069 6D6C_H


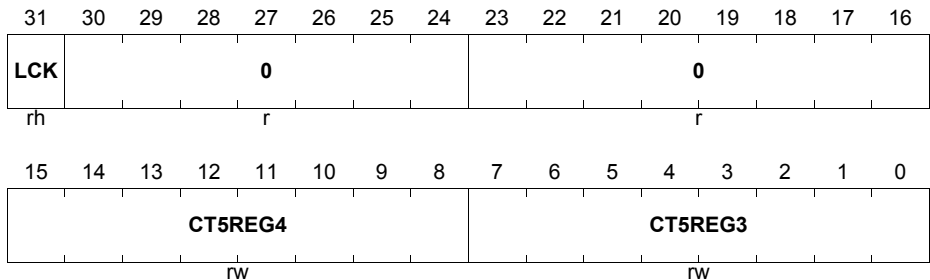
Field	Bits	Type	Description
SD33P	[7:0]	rw	P Coefficient P control parameter for the PID regulator (3.3V). Bits [7:4] - Mantissa, Bits [3:0] - Exponent.

Field	Bits	Type	Description
SD33I	[15:8]	rw	I Coefficient I control parameter for the PID regulator (3.3V). Bits [7:4] - Mantissa, Bits [3:0] - Exponent.
SD33D	[23:16]	rw	D Coefficient D control parameter for the PID regulator (3.3V). Bits [7:4] - Mantissa, Bits [3:0] - Exponent.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and cannot be updated
0	[30:24]	r	Reserved Read as 0; should be written with 0.

EVRSDCOEFF3
EVR13 SD Coefficient Register 3
(1C8_H)
Reset Value:00B6 5212_H


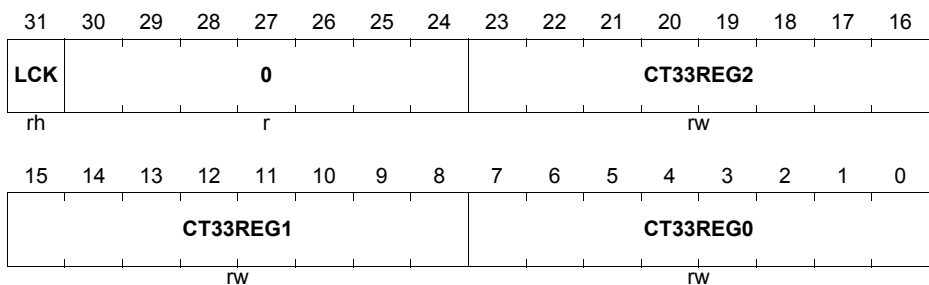
Field	Bits	Type	Description
CT5REG0	[7:0]	rw	Commutation trimming Trimming of the commutation parameters of the external driver (5V). sd_ctreg0_trim_i(4) activates VGATE1N pull-down 0 = Clamp to VSS 1 = No Clamp

Field	Bits	Type	Description
CT5REG1	[15:8]	rw	Commutation trimming Trimming of the commutation parameters of the external driver (5V).
CT5REG2	[23:16]	rw	Commutation trimming Trimming of the commutation parameters of the external driver (5V).
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and cannot be updated
0	[30:24]	r	Reserved Read as 0; should be written with 0.

EVRSDCOEFF4
EVR13 SD Coefficient Register 4
(1CC_H)
Reset Value:0000 5AB6_H


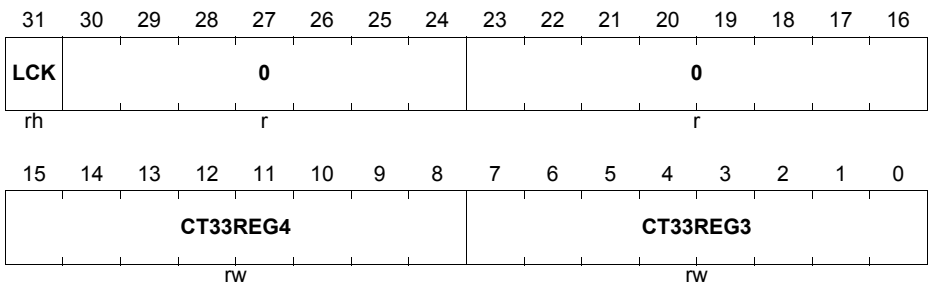
Field	Bits	Type	Description
CT5REG3	[7:0]	rw	Commutation trimming Trimming of the commutation parameters of the external driver (5V).
CT5REG4	[15:8]	rw	Commutation trimming Trimming of the commutation parameters of the external driver (5V).

Field	Bits	Type	Description
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and cannot be updated
0	[30:16]	r	Reserved Read as 0; should be written with 0.

EVRSDCOEFF5
EVR13 SD Coefficient Register 5
(1D0_H)
Reset Value:00B6 5212_H


Field	Bits	Type	Description
CT33REG0	[7:0]	rw	Commutation trimming Trimming of the commutation parameters of the external driver (3.3V).
CT33REG1	[15:8]	rw	Commutation trimming Trimming of the commutation parameters of the external driver (3.3V).
CT33REG2	[23:16]	rw	Commutation trimming Trimming of the commutation parameters of the external driver (3.3V).

Field	Bits	Type	Description
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0_B The register is unlocked and can be updated 1_B The register is locked and cannot be updated
0	[30:24]	r	Reserved Read as 0; should be written with 0.

EVRSDCOEFF6
EVR13 SD Coefficient Register 6 (1D4_H) **Reset Value:0000 5AB6_H**


Field	Bits	Type	Description
CT33REG3	[7:0]	rw	Commutation trimming Trimming of the commutation parameters of the external driver (3.3V).
CT33REG4	[15:8]	rw	Commutation trimming Trimming of the commutation parameters of the external driver (3.3V).
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0_B The register is unlocked and can be updated 1_B The register is locked and cannot be updated

Field	Bits	Type	Description
0	[30:16]	r	Reserved Read as 0; should be written with 0.

7.3.2 Power Management

7.3.2.1 Power Management Overview

The Power Management scheme allows activation of power down modes so that the system operates with the minimum required power for the corresponding application state. A progressive reduction in power consumption is achieved by invoking Idle, Sleep or Standby modes respectively. The Idle mode is specific to each individual CPU where as Sleep and Standby modes influence the complete system.

As shown in [Table 7-20](#), there are two power modes available for each CPU:

- CPU Run Mode
- CPU Idle Mode

Table 7-20 CPU Power Management

Mode	Description
Run Mode	The CPU clock is active and code is being executed.
Idle Mode	<p>CPU may enter Idle Mode on following events:</p> <ul style="list-style-type: none"> • On a SW Idle request issued by setting register bits PMCSRx.REQSLP = 01_B when CPU has no active tasks to perform. • On a SW Idle request (PMCSRy.REQSLP = 01_B) issued by another CPU. • On a SMU Idle request in case CPU faults are detected triggering safety alarms which are configured in SMU to set CPU into Idle state. <p>The CPU code execution is halted and CPU clock is disabled in Idle state. The peripherals continue to remain active. CPU RAM memories (PSPR / DSPR) are accessible to other bus masters and peripherals.</p> <p>CPU may exit Idle mode on following events:</p> <ul style="list-style-type: none"> • When an interrupt occurs on a CPU returning the CPU to Run Mode. • When a trap occurs like an NMI trap event. • When the CPU watchdog or Safety watchdog timer overflow events trigger an SMU alarm in turn leading to a CPU interrupt. • When a MSB bit wrap of the CPU Watchdog counter takes place. • When an Application reset, System reset or any higher reset occurs.

As shown in [Table 7-21](#), there are three main power modes available for the system:

- System Run Mode
- System Sleep Mode
- System Standby Mode

Table 7-21 System Power Management

Mode	Description
Run Mode	At least one master CPU has not requested Sleep Mode or Standby mode and is in Run mode. All peripheral modules are active.
Sleep Mode	<p>System may enter Sleep Mode on following events:</p> <ul style="list-style-type: none"> • On a SW Sleep request issued by setting $PMCSRx.REQSLP = 10_B$ by the master CPU. <p>CPU code execution is halted and CPU Idle state is entered. Peripherals are set into sleep state if so configured in the respective $CLCx.EDIS$ bit. Ports retain their earlier programmed state.</p> <p>System may exit Sleep mode on following events:</p> <ul style="list-style-type: none"> • When an interrupt or trap occurs on the master CPU. • When an NMI trap event takes place. • When the CPU watchdog or Safety watchdog timer overflow events trigger an SMU alarm leading in turn to a master CPU interrupt. • When a MSB bit wrap of master CPU Watchdog counter takes place. • When an Application reset, System reset or any higher reset occurs.
Standby Mode	<p>System may enter Standby Mode on following events if so configured:</p> <ul style="list-style-type: none"> • On a SW Standby request issued by setting $PMCSRx.REQSLP = 11_B$ by the master CPU. • On a secondary undervoltage event during V_{EXT} supply ramp-down. • On an $NMI / \overline{ESR1}$ assertion event. <p>The Standby domain constituting the Standby RAM, the 8 bit Standby Controller, shared ports and the wake-up unit remain actively supplied. The power to the rest of the chip is completely switched off.</p> <p>System may exit Standby mode on following events in case V_{EXT} remains supplied during Standby state:</p> <ul style="list-style-type: none"> • when a wake-up edge is detected on selected pins / $\overline{ESR1}$. • when a wake-up request is issued by the 8 bit standby controller. • when \overline{PORST} assertion is detected.

Furthermore, flexible reduction of power consumption is possible through following measures:

- Reduction of specific CPU power consumption by means of CPU clock scaling.
- Disabling the module clock by setting bit DISR in module CLC register.
- Reducing the system frequency without changing individual peripheral clocks.
- Reducing peripheral clock frequency without changing system clock frequency.

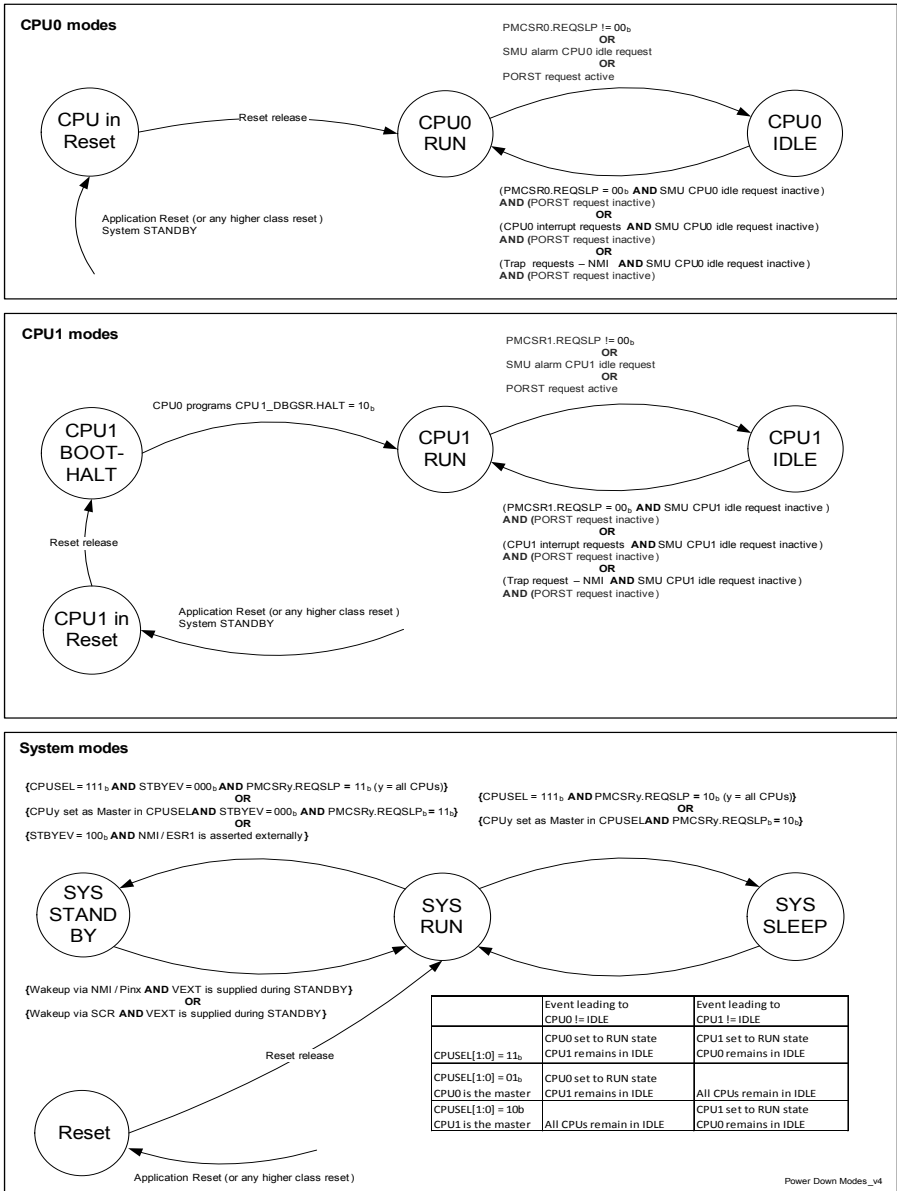


Figure 7-37 Power down modes and transitions

7.3.2.2 Idle Mode

In case there are no active tasks to perform, a CPU may be requested to enter Idle mode during runtime by writing to the respective PMCSR_x register and setting the bitfield REQSLP = 01_B.

Entering Idle Mode :

Following events can invoke a CPU_x Idle request

- CPU_x setting itself in Idle by writing its own PMCSR_x register: The respective PMCSR_x register shall be accessed by setting CPU_x ENDINIT = 0_B and consequently writing the bitfield REQSLP = 01_B. The Idle transition takes place only when CPU_x ENDINIT = 1_B is set back again. This ensures that a CPU_x does not enter Idle mode when it's WDT_x is in Time-Out mode and ENDINIT_x = 0_B to avoid wake-up on a consequent WDT time-out. Safety ENDINIT mechanism shall not be used by a CPU to set itself into Idle to avoid wake-up on a Safety WDT time-out. It is to be noted that CPU is set into Idle state during entry into SLEEP (REQSLP = 10_B) mode as well. In STANDBY mode (REQSLP = 11_B), CPU is set first into Idle state before power is switched off. Idle mode may also be simultaneously triggered for additional CPUs based on PMSWCR1.CPUIDLSEL configuration.
- CPU_y setting CPU_x into Idle: The PMCSR_x register shall be accessed by CPU_y by setting Safety ENDINIT = 0_B. The Idle request is issued immediately on setting PMCSR_x.REQSLP = 01_B. The device no longer waits for Safety ENDINIT = 1_B to be set back to trigger Idle transition. In case CPU_x is already in Idle owing to an SMU Idle request, REQSLP_x write accesses by CPU_y shall be ignored to keep CPU_x in permanent idle until the next application reset. SMU Software alarms maybe used alternatively to set a CPU_x in permanent Idle via CPU_y.
- CPU_x being set into Idle by SMU alarm: Idle mode may be requested by an SMU alarm in case of detected CPU faults thus setting the faulty CPU in Idle without initiating a full application reset.

The CPU watchdog may be disabled or slowed down by reprogramming the timers before triggering Idle request via Software. On an Idle request, the CPU finishes its current operations and sends an acknowledge back to the Power Management unit. It then enters an inactive state in which the CPU and the respective DMI and PMI memory units are shut off.

State during Idle mode

During Idle Mode, memory accesses to the DMI and PMI from other bus masters cause these units to wake-up automatically to handle these transactions. When memory transactions are complete, the DMI and PMI return to Idle state again. Once Idle Mode is entered, the state is reflected in PMCSR_x.PMST status bits. CPU Idle request from the SMU is reflected in PMCSR_x.SMUSLP bit.

Exiting Idle mode

In Idle mode, if there is no outstanding SMU Idle request to the CPU, the CPU will return to Run mode in response to the following events:

- An interrupt / trap received from an interrupt / trap source mapped to the CPU.
- An NMI trap request is received to wake-up all the CPUs.
- A MSB bit wrap of the corresponding CPU Watchdog counter occurs.
- Setting the register bits `PMCSRx.REQSLP = 00B` to set the CPUx into Run mode.

If there is an outstanding SMU Idle request to the CPU, then the CPU may only return to Run mode after an application reset or any higher reset. The system enters reset state on an Application, System reset or any higher reset. If it is woken by a watchdog timer overflow event routed via the SMU to the CPU or by an NMI or by an interrupt, the CPU will immediately vector to the appropriate interrupt / trap handler.

7.3.2.3 Sleep Mode

System may be requested to enter Sleep mode via software by master CPU by writing to the CPU's PMCSR_x register and setting the bitfield REQSLP = 10_B.

Sleep Mode is a system mode and may be entered as a unanimous decision of all the CPUs when ALL of the PMCSR_x registers in the system request it AND **PMSWCR1**.CPUSEL = 111_B. Sleep Mode may also be entered based on a singular decision of a master CPU based on the configuration of the CPUSEL register. The PMCSR_x register shall be accessed by setting CPU_x ENDINIT = 0_B. The Sleep request is issued only after CPU_x ENDINIT bit is set back again. Safety ENDINIT mechanism shall not be used by a CPU_x to issue Sleep request.

Entering Sleep Mode

It should be ensured to select individual clocks from Clock Control Unit for peripherals which need to remain active during Sleep mode. This allows the reduction of system clock frequencies, namely SRI and SPB clocks, to the minimum possible values via the low power divider. Low power modes for Analog and Flash modules may also be requested. The CLC_x.EDIS register bit is cleared for all peripherals intended to be inactive in Sleep mode. The watchdogs may be disabled or slowed down before issuing a Sleep request.

State during Sleep Mode

Sleep Mode is disabled for a unit if CLC_x.EDIS bit is set. The sleep request is ignored in this case and the corresponding unit continues normal operation as intended. If CLC_x.EDIS is cleared, the clock of the module is gated. The exact sequence used for entering Sleep Mode is determined for each module by the setting of register field OCS_x.SUS. CPU Idle state is entered for all the CPUs as described in the previous section. All ports retain their earlier programmed state.

Exiting Sleep Mode

The system will exit Sleep mode on any wakeup event that causes any master CPU to exit Idle Mode depending on CPUSEL configuration. Only the master CPU associated with the interrupt wake-up event would be set into Run mode (REQSLP = RUN, PMST = RUN). Other CPUs will remain in Idle (REQSLP = SLEEP, PMST = IDLE). An NMI trap event will wake-up ALL the CPUs. A MSB bit wrap of the corresponding master CPU Watchdog counter would also wake-up the master CPU. The response of the CPU to being woken up from Sleep Mode is also the same as for Idle Mode. Peripheral units that have entered Sleep Mode will switch back to their selected Run Mode operation.

7.3.2.4 Standby Mode

The Standby domain constitutes the Standby RAM, the 8 bit Standby Controller, the power management unit, the wake-up unit, the V_{EXT} monitor and basic infrastructure components. The Standby domain is supplied by the EVR pre-regulator and is per default clocked by the 100 KHz internal low power clock source in Standby Mode.

Following events may trigger Standby mode entry based on PMSWCR1.STBYEV bits.

- Standby entry on SW request (PMCSRx.REQSLP = 11_B). The Standby request is issued only after CPUx ENDINIT bit is set back again. Safety ENDINIT mechanism shall not be used by a CPUx to issue Standby request.
- Standby entry on $\overline{ESR1}$ / \overline{NMI} edge event.

Following events may trigger wake-up from Standby mode.

- Wake-up via \overline{NMI} / Pinx: Wake-up on rising, falling or any edge of \overline{NMI} / $\overline{ESR1}$, Pin A or Pin B pins. Standby domain is supplied via V_{EXT} supply pins which in turn is supplied by an external regulator with Standby support.
- Wake-up via SCR: Wake-up is triggered by the Standby controller. V_{EXT} is supplied by an external regulator with Standby support. Wake-up on V_{EXT} supply ramp-up is unavailable in this configuration.

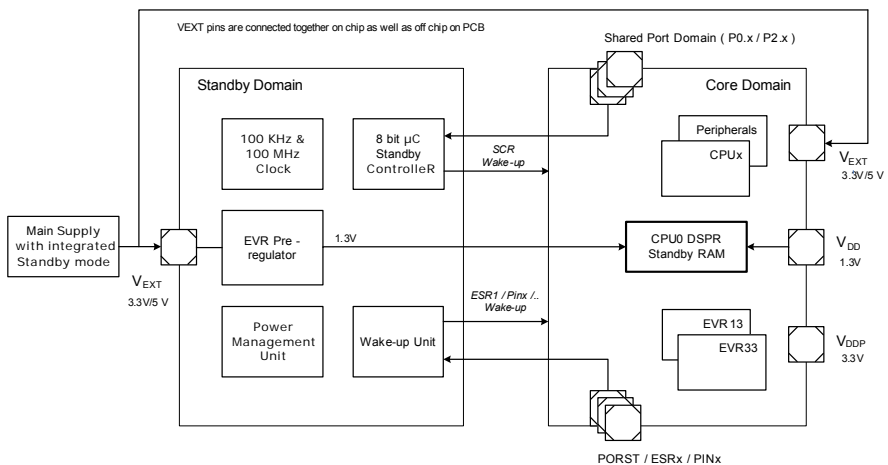


Figure 7-38 Standby domain

The Standby RAM constitutes a single block of ECC protected DSPR RAM of CPU0. The RAM remains supplied during Standby mode if configured in PMSWCR0.STBYRAMSEL bits. On wake-up, the status of the Standby RAM is reflected in PMSWSTAT.STBYRAM bits. During Standby state, the supply to the Standby RAM cell array is supplied by EVR Pre-regulator.

External events may be mapped to ESRx / PINx pins in turn acting as wake-up signals for the system. In Run Mode, NMI / ESR1 pin may be used as fault or functional interface for external devices. In Standby Mode, an edge event on the NMI / ESR1 pin may be configured to trigger wake-up of the main core domain via **PMSWCR0**.ESR1WKEN bit and is reflected in **PMSWSTAT**.ESR1WKEN status flag. It can be configured to trigger a wake-up on rising, falling or both edges via **PMSWCR0**.ESR1EDCON bit. Glitches on ESR1 input are filtered out by activating the filter via **PMSWCR0**.ESR1DFEN bit. Additional pins (PINA - P14.1 and PINB - P15.1) may likewise be configured to trigger wake-up via **PMSWCR0**.xEDCON, xDFEN & xWKEN bits. On wake-up, **PMSWSTAT**.ESR1WKP or PINxWKP event flags provide information as to the wake-up source. It should be taken care after wake-up to clear the event flags via **PMSWSTATCLR** register. In case new wake-up events are captured while **PMSWSTAT**.xWKP flags are still set, then **PMSWSTAT**.xOVRRUN flags are set to indicate an overrun state owing to consecutive unserved wake-up events.

The 8 bit Standby controller (SCR) subsystem constitutes an XC800 core, 8KB XRAM memory, various timer and communication peripherals and 16 shared pins executing autonomous activity during Idle, Sleep and Standby modes. Various Standby functions and periodic monitoring tasks may be encapsulated in the SCR with minimal power consumption overhead. The SCR interacts with the main core domain via a dedicated SPI (QSPI2) interface and **PMSWCR2** register bits. The SCR is enabled via **PMSWCR0**.SCREN bit. The SCR enable bit needs to be set after every SCR reest triggered from the SCU side. After start-up, CPU0 programs the SCR via SPI interface and copies the code into the internal XRAM. A reset may be issued to the SCR via **PMSWCR1**.SCRSTREQ register bit. Consequently **PMSWSTAT**.SCRST register bit is flagged to indicate SCR reset and shall be cleared via **PMSWSTATCLR**.SCRSTCLR bit. After reset release, the firmware initializes the SCR subsystem based on the hardware configuration programmed in **PMSWCR1**.SCRCFG bits. The 100 KHz clock source is the default clock active in Standby Mode. The clock source may be switched to the internal 100 MHz clock source via **PMSWCR0**.SCRCLKSEL register bit thus enabling higher performance on the SCR subsystem. The SCR clock system in turn divides the 100 MHz clock to a value below 20 MHz maximum SCR clock frequency. SCR port module shares a part of the port domain with the main port system which may be kept active during Standby mode. The control to these pins need to be allocated explicitly to the SCR via port configuration registers. The SCR can make a direct interrupt request to the CPU in Run Mode and likewise CPU can also trigger a direct interrupt request to the SCR. It is possible to access the SCR XRAM via QSPI2 (SLSO15) interface. Simultaneous access to XRAM via QSPI2 and the XC800 is arbitrated by the SCR memory arbiter. Additionally, It is also possible to exchange data between SCR and main domain via **PMSWCR2** register. Unused wake-up pins may be configured as tristate in Standby Mode. SCR may wakeup the main core domain from Standby state if configured in **PMSWCR0**.SCRWKEN register bit. The enabling of SCR wake-up via **PMSWCR0**.SCRWKEN should be programmed when SCR is running at 20 MHz. A

wake-up request is issued by the SCR SW via SCRWKP bit in STDBYWKP register as documented in the SCR SCU. On wake-up of the main core domain, **PMSWSTAT.SCRWKP** event flag is set which shall be cleared via **PMSWSTATCLR.SCRWKPCLR** register bit. Some of the system status and control registers are also available in the SCR subsystem. Critical SCR errors like XRAM ECC errors, SCR watchdog overflow event and reset are also communicated back to the main core domain via **PMSWCR2** register bits. The SCR is reset in case **PORST** is externally asserted. The SCR is not affected by an Application or System reset.

Entering Standby Mode (ESR1 / Pin or SCR Wake-up configuration)

The Standby Mode entry may be requested by writing to PMCSR_x register to set bitfield REQSLP = 1_B or via NMI / $\overline{\text{ESR1}}$ assertion as configured in **PMSWCR1**.STBYEV bits.

Standby mode via SW may be entered based on a singular decision from a master CPU based on the configuration in the CPUSEL register. It may also be entered as a unanimous decision of all the CPUs when ALL of the PMCSR_x registers in the system request it and **PMSWCR1**.CPUSEL = 111_B. Each PMCSR_x register is written by the corresponding CPU_x.

Before entering standby mode, various modules should be sequentially shut off in a sequence mainly to avoid large current jump on standby entry.

- All peripherals and interrupts associated with CPUs except master CPU are switched off. This is to avoid wake-up of the CPUs once they are put into IDLE state.
- All CPUs except the master CPU are sequentially put into IDLE state. CPU watchdogs may be disabled or reconfigured for slower modes.
- Peripheral module clocks are switched off in respective CLC.DISR registers.
- Master CPU frequency reduction in steps compliant to load jump constraints.
- Master CPU code execution is switched from Flash to PSPR RAM. Flash modules are deactivated via FCON register.
- System Clock is switched to the internal 100 MHz clock source. System PLL & E-Ray PLL are switched off. Clock dividers are programmed to lower values.
- Set **PMSWCR1**.IRADIS bit to disable Idle Request Acknowledge sequence activation for fast Standby Mode entry. This ensures that standby request is not blocked by a pending reset request / sequence.
- Select the Standby RAM block via **PMSWCR0**.STBYRAMSEL bits. Copy Standby RAM redundancy data (16 words in size) to DSPR0 starting at 0xD000 2000h. The procedure is also described in Boot ROM chapter in section "Preparation before to enter Standby mode". Dcache write back to be executed before Standby entry.
- Select the clock source which need to be active on entry into Standby Mode via **PMSWCR0**.SCRCLKSEL bits. Wake-up trigger edge configuration and filter activation is configured via **PMSWCR0**.xxxEDCON and **PMSWCR0**.xxxDFEN bits.
- Configure pad state via **PMSWCR0**.TRISTREQ bit. All pads may either be in tristate or have pull-up devices active. Configure **PMSWCR0**.ESR0TRIST bit to configure ESR0 behavior as reset output or tristate during Standby and on wake-up. In case of HWCFG [2:5] pins it is recommended to tie them to external pull devices.
- Enable ESR1 or PIN_x pins for wake-up via **PMSWCR0**.xxxWKEN bits. The wake-up from 8 bit should be activated via **PMSWCR0**.SCRWKEN bit. The enabling of SCR wake-up via **PMSWCR0**.SCRWKEN should be programmed when SCR is running at 20 MHz. The shared ports are configured either to be in tristate during standby or left to the control of SCR. SCR should be running before entering the Standby state in case used.
- Configure Standby Entry event in **PMSWCR1**.STBYEV register bits.

- Standby request issued. An orderly shut down of various sub-systems is triggered to enter Standby mode.

State during Standby Mode

The Standby RAM (DSPR RAM of CPU0), the 8 bit Standby controller, the shared ports and the wake-up logic are kept alive in this mode. PORST pin serves as a wakeup source when asserted and shall be held high externally for NMI and SCR Wake-up configuration to remain in Standby state.

All other ports are set in their default reset state. The default port behavior during standby and after wake-up may be configured as pull-up or tri-state accordingly by **PMSWCR0.TRISTREQ** register bit. The ESR0 pin may be configured as reset output or tristate during Standby mode by configuring **PMSWCR0.ESR0TRIST** bit.

The SCR continues to operate as a stand-alone 8 bit controller executing the intended operations. It should be ensured that the shared ports are configured in the corresponding port registers and the ownership of the pins are assigned either to the SCR or the main domain. In case the SCR needs to drive outputs during Standby mode, the default clock may need to be switched from 100 KHz to 20 MHz clock source. The SCR may request the EVR to activate or deactivate the 20 MHz clock. SCR may be programmed to issue wake-up based on inputs from internal modules or shared pins. When the wake-up of the main core domain is required, the SCR issues a wake-up request via SCRWKP bit in STDBYWKP register as documented in the SCR SCU

Exiting Standby Mode - Wake-up event

The wake-up event in case of NMI/Pin or SCR wake-up configuration may happen on

- ESR1 edge transition (NMI trap)
- Pin A or Pin B edge transition (P14.1 or P15.1)
- Wake-up from SCR via register STDBYWKP.SCRWKP in turn caused by following events
 - Edge transitions at the shared ports
 - RTC interrupt
 - SCR watchdog overflow
 - Selected interrupts from communication modules

The main EVRx3 regulators are ramped up on wake-up based on the latched configuration in **PMSWSTAT.HWCFGEVR** register bits. Firmware is executed thereafter including installation of RAM redundancy data. On wake-up, all pads are either in tristate or are connected to pull-ups as indicated in **PMSWSTAT.TRIST** register bit. ESR0 behaviour is indicated in **PMSWSTAT.ESR0TRIST** register bit. If Standby RAM was supplied during Standby state, it is indicated in **PMSWSTAT.STBYRAM** register bits. Additional RAM integrity checks may be carried out after wake-up. RSTSTAT.STBYR bit indicates that the supply was reliable during Standby. The wake-up and over-run status flags are set in **PMSWSTAT** register and shall be cleared by **PMSWSTATCLR**

register. The wake-up time is quite the same as the normal boot time as EVR need to be started and firmware need to be consequently executed. In case **PMSWSTAT**.STBYRAM bits are set after wake-up, it is ensured that the Standby RAM (CPU0 DSPR) is not initialised after wake-up by Firmware irrespective whether PROCOND.RAMIN / RAMINSEL bits are configured in Flash to initialise the Standby RAM region.

Exiting Standby Mode - Power Fail or $\overline{\text{PORST}}$ assertion

A power fail event of the Standby supply during Standby mode will inevitably result in the loss of Standby RAM contents. Consequently, a cold $\overline{\text{PORST}}$ event is issued and the Standby domain is set into reset. EVR Pre-regulator under-voltage violation is indicated in RSTSTAT.STBYR flag.

In case of NMI or SCR Wake-up configuration, the $\overline{\text{PORST}}$ pin triggers a wakeup from Standby if asserted low during Standby state. The SCR is also set into reset. The device boots up ramping up the regulators followed by firmware execution similar to a normal device start-up.

The Standby RAM contents are kept intact after $\overline{\text{PORST}}$ assertion. In case **PMSWSTAT**.STBYRAM bits are set after wake-up, it is ensured that the Standby RAM (CPU0 DSPR) is not initialised after a warm PORST wake-up by Firmware irrespective whether PROCOND.RAMIN / RAMINSEL bits are configured in Flash to initialise the Standby RAM region. Reset is propagated to external devices via the ESR0 pin on exit from Standby mode depending on **PMSWCRO**.ESR0TRIST configuration. Firmware may elongate ESR0 reset output depending on Flash configuration.

7.3.2.5 Power Management Registers

Power Management Control and Status Register

The set of registers used for Power Management control the issue of power modes, manage wake-up configuration and provide status information on mode transitions. The request for Idle, Sleep or Standby mode is issued via PMCSRx registers.

PMCSR0

Power Management Control and Status Register

(0D4_H)

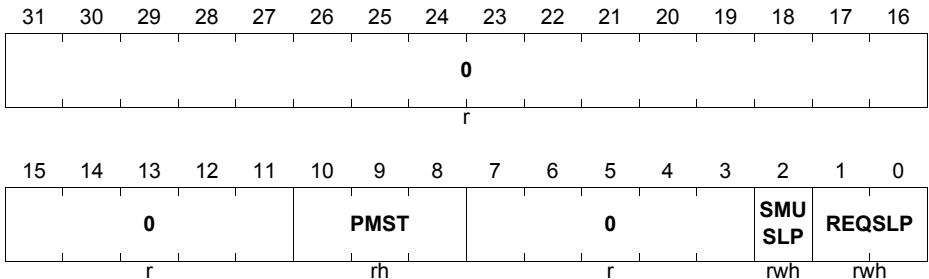
Reset Value: 0000 0100_H

PMCSR1

Power Management Control and Status Register

(0D8_H)

Reset Value: 0000 0100_H



Field	Bits	Type	Function
REQSLP	[1:0]	rwh	<p>Idle Mode and Sleep Mode Request</p> <p>00_B Request CPU Run Mode 01_B Request CPU Idle Mode 10_B Request System Sleep Mode 11_B Request System Standby Mode</p> <p>In Idle Mode or Sleep Mode, these bits are cleared in response to an interrupt for the CPU, or when bit 15 of the corresponding CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) changes from 0 to 1. In Standby Mode, these bits are cleared on wake-up. REQSLP maybe written only when either CPU or Safety ENDINIT bits are set to 0. CPU ENDINIT bit has to be set back after REQSLP is written for the mode transition to take place. In case of Safety ENDINIT, the mode transition will be issued immediately and does not wait till Safety ENDINIT is set back to 1 again.</p>
SMUSLP	2	rwh	<p>SMU CPU Idle Request</p> <p>0_B No SMU Alarm request for CPU Idle Mode 1_B SMU Alarm requested CPU Idle Mode</p> <p>This bit is set to '1' when an SMU Alarm requests that the CPU is put into Idle Mode.</p> <p>This bit is NOT cleared in response to an interrupt to the CPU, or when bit 15 of the CPU Watchdog Timer register (bit WDTCPUxSR.TIM[15]) change from 0 to 1. It is cleared only by reset or by a write of '0' to this register. An attempt to write '1' has no effect.</p>
0	3	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>
PMST	[10:8]	rh	<p>Power management Status</p> <p>This bit field reflects the current status of the CPU.</p> <p>000_B Reserved, do not use this combination 001_B Normal Run Mode¹⁾ 010_B CPU Idle Mode requested 011_B CPU Idle Mode acknowledged 100_B Sleep Mode requested 101_B Reserved, do not use this combination 110_B Standby Mode requested 111_B Reserved, do not use this combination</p>

Field	Bits	Type	Function
0	[7:4], [31:11]	r	Reserved Read as 0; should be written with 0.

1) After a reset, all CPUs are in "Normal Run Mode", but this does not mean that all CPUs are executing code. This mode also includes the CPU "halt" mode which is the start-up default for all except CPU0.

Standby and Wake-up Control Registers

PMSWCRO

Standby and Wake-up Control Register 0 (0C8_H)

Reset Value: 0001 02D0_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	0	ESR0TRIST	0	DCDCSYNC	0	PORSTDF	TRISREQ	TRISTEN	SCRWKEN	SCRCLKSEL	STBYRAMSEL	SCR	EN		
rw	r	rw	r	rw	r	rw	rw	w	rw	rw	rw	rw	rw		rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PINBEDCON	PINBDFEN	PINAEDCON	PINADFE	ESR1EDCON	ESR1DFEN	ESR0EDCON	ESR0DFEN	PINBWKEN	PINAWKEN	ESR1WKEN	0				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r				

Field	Bits	Type	Function
0	0	r	Reserved Read as 0; should be written with 0.
ESR1WKEN	1	rw	ESR1 Wake-up enable from Standby 0 _B System wake-up via ESR1 pin is disabled. 1 _B System wake-up is enabled via ESR1 pin.
PINAWKEN	2	rw	Pin A Wake-up enable from Standby 0 _B System wake-up via Pin A is disabled. 1 _B System wake-up is enabled via Pin A.
PINBWKEN	3	rw	Pin B Wake-up enable from Standby 0 _B System wake-up via Pin B is disabled. 1 _B System wake-up is enabled via Pin B.
ESR0DFEN	4	rw	Digital Filter Enable This bit activates ESR0 digital spike filter. 0 _B The filter is bypassed 1 _B The filter is used

Field	Bits	Type	Function
ESR0EDCON	[6:5]	rw	Edge Detection Control This bit field defines the edge of a ESR0 wake-up trigger 00 _B No trigger is generated 01 _B A trigger is generated upon a rising edge 10 _B A trigger is generated upon a falling edge 11 _B A trigger is generated upon a rising OR falling edge
ESR1DFEN	7	rw	Digital Filter Enable This bit activates ESR1 digital spike filter. 0 _B The filter is bypassed 1 _B The filter is used
ESR1EDCON	[9:8]	rw	Edge Detection Control This bit field defines the edge of a ESR1 wake-up trigger 00 _B No trigger is generated 01 _B A trigger is generated upon a rising edge 10 _B A trigger is generated upon a falling edge 11 _B A trigger is generated upon a rising OR falling edge
PINADFEN	10	rw	Digital Filter Enable This bit activates Pin A digital spike filter. 0 _B The filter is bypassed 1 _B The filter is used
PINAEDCON	[12:11]	rw	Edge Detection Control This bit field defines the edge of a Pin A wake-up trigger 00 _B No trigger is generated 01 _B A trigger is generated upon a rising edge 10 _B A trigger is generated upon a falling edge 11 _B A trigger is generated upon a rising OR falling edge
PINBDFEN	13	rw	Digital Filter Enable This bit activates Pin B digital spike filter. 0 _B The filter is bypassed 1 _B The filter is used

Field	Bits	Type	Function
PINBEDCON	[15:14]	rw	Edge Detection Control This bit field defines the edge of a Pin B wake-up trigger 00 _B No trigger is generated 01 _B A trigger is generated upon a rising edge 10 _B A trigger is generated upon a falling edge 11 _B A trigger is generated upon a rising OR falling edge
SCREN	16	rw	Standby Controller Enable request 0 _B 8 bit Standby Controller is disabled 1 _B 8 bit Standby Controller is enabled <i>Note: The SCU cold PORST reset value is 0. Firmware initializes the value to 1 and carries out SCR-XRAM initialization after cold PORST event. In case of unsuccessful boot, Firmware resets value to 0.</i>
STBYRAMSEL	[18:17]	rw	Standby RAM supply in Standby Mode 00 _B Standby RAM is not supplied. 01 _B Standby RAM (CPU0 DSPR) is supplied. <i>Note: All other bit combinations are reserved. All other bit combinations are reserved. In case Standby RAM supply is activated, Standby RAM is not initialised after wakeup or PORST irrespective of configuration in PROCOND. RAMINSEL / RAMIN.</i>
SCRCLKSEL	19	rw	Default Clock selection on Standby Mode Entry 0 _B Only 100 KHz Clock Source is active in Standby Mode. 1 _B Both 100 KHz Clock Source and 100 MHz Clock to SCR are active in Standby Mode. SCR CCU divides the clock internally to run at lower frequency.
SCRWKEN	20	rw	Standby Controller Wake-up enable from Standby 0 _B System wake-up via 8 bit Standby Controller is disabled. 1 _B System wake-up is enabled via 8 bit Standby Controller.

Field	Bits	Type	Function
TRISTEN	21	w	Bit protection for Tristate request bit (TRISTREQ) Setting this bit enables that bit TRISTREQ can be changed by a write operation. 0 _B TRISTREQ keeps the previous state and cannot be changed. 1 _B TRISTREQ bit can be changed with a write operation.
TRISTREQ	22	rw	Tristate enable This bit decides whether pads behave as inputs with weak pull-up or tristate on reset assertion/deassertion or Standby- Wake-up transition. 0 _B No request to switch the input pad state of all the pads to tristate from pull-up (default reset state) 1 _B Pad domain in tristate.
PORSTDF	23	rw	PORST Digital Filter enable This bit field enables additional $\overline{\text{PORST}}$ digital filter to provide enhanced immunity against spurious spikes. 0 _B $\overline{\text{PORST}}$ recognition delay = Analog $\overline{\text{PORST}}$ pad filter delay (default reset state). 1 _B $\overline{\text{PORST}}$ recognition delay = Analog $\overline{\text{PORST}}$ pad filter delay + Digital filter delay.
0	24	r	Reserved Read as 0; should be written with 0.
DCDCSYNC	25	rw	DC-DC Synchronisation Enable This bitfield enables the synchronisation clock output to synchronize an external DC DC regulator with respect to the internal SMPS EVR13 regulator. 0 _B DC-DC Synchronisation signal not available. 1 _B DC-DC Synchronisation signal available.
0	[28:26]	r	Reserved Read as 0; should be written with 0.
ESR0TRIST	29	rw	ESR0 Tristate enable This bit configures ESR0 pin behavior either as reset output or tristate during Standby mode. 0 _B ESR0 configured as reset output and is held low during Standby state (default reset state) 1 _B ESR0 in tristate during Standby state.

Field	Bits	Type	Function
0	30	r	Reserved Read as 0; should be written with 0.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0_B The register is unlocked and can be updated 1_B The register is locked and cannot be updated

Additional PORST digital filter activated via PMSWCR0.PORSTDF bit provides additional filtering of atleast 500 ns to provide enhanced immunity against spurious spikes. This is in addition to the inherent analog PORST filter delay of the PORST pad / pin as documented in the datasheet. After cold PORST this delay is by default inactive.

PMSWCR1
Standby and Wake-up Control Register 1 (0E8_H)
Reset Value: 0100 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	STBYEV		STBYEV EN	CPUSEL			SCRCFG								
rh	rw		w	rw			rw								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		IRADIS	0	CPUIDLSEL			0					SCRSTR EQ	SCRSTEN		
r		rw	r	rw			r					w	w		

Field	Bits	Type	Description
SCRSTEN	0	w	Standby Controller Reset request enable 0_B Bit SCRSTREQ is not updated 1_B Bit SCRSTREQ can be updated
SCRSTREQ	1	w	Standby Controller Reset request 0_B No request for reset of the 8 bit Standby Controller. 1_B 8 bit Standby Controller reset request.

Field	Bits	Type	Description
0	[7:2]	r	Reserved Read as 0; should be written with 0.
CPUIDLSEL	[10:8]	rw	CPU selection for Idle mode This bit field allows a CPUx to issue Idle request to other CPUs in addition to itself. A request for Idle via PMCSRx.REQSLP=01 by CPUx will also trigger Idle requests to all other CPUs. 000 _B Entry to the respective Idle mode is decided by each individual CPU. 001 _B CPU0 Idle request will send all CPUs in Idle. 010 _B CPU1 Idle request will send all CPUs in Idle. <i>Note: All other CPUIDLSEL bit combinations are reserved.</i>
0	11	r	Reserved Read as 0; should be written with 0.
IRADIS	12	rw	Idle-Request-Acknowledge Sequence Disable This bit enables SCU Idle Request Acknowledge sequence to all modules on Standby entry. 0 _B Idle-Request-Acknowledge Sequence issued on Standby entry. 1 _B Idle-Request-Acknowledge Sequence skipped on Standby entry. This bit shall be set before Standby entry to disable Idle request acknowledge sequence so that standby request is not blocked by a pending reset request / sequence.
0	[15:13]	r	Reserved Read as 0; should be written with 0.

Field	Bits	Type	Description
SCRCFG	[23:16]	rw	<p>Hardware configuration of the 8 bit controller.</p> <p>00_H 8 bit XRAM is not programmed (default)</p> <p>01_H User Mode (Execution from 0000_H XRAM address)</p> <p>02_H OCDS Mode (SCR DAP0_0/DAP1_0 pin mode)</p> <p>03_H OCDS Mode (SCR DAP0_1/DAP1_1 pin mode)</p> <p>04_H OCDS Mode (SCR SPD_0 pin mode)</p> <p>05_H OCDS Mode (SCR SPD_0 pin mode)</p> <p>06_H OCDS Mode (SCR SPD_1 pin mode)</p> <p>07_H OCDS Mode (SCR SPD_1 pin mode)</p> <p>0A_H OCDS Mode (SOC DAP mode)</p> <p>0B_H OCDS Mode (SOC DAP mode)</p> <p>0C_H OCDS Mode (SOC SPD mode)</p> <p>0D_H OCDS Mode (SOC SPD mode)</p> <p>0E_H OCDS Mode (SOC SPD mode)</p> <p>0F_H OCDS Mode (SOC SPD mode)</p> <p><i>Note: Any change in SCRCFG is followed by a SCRSTREQ reset request of the 8 bit controller to start off in the chosen mode. All other bit combinations are reserved.</i></p>
CPUSEL	[26:24]	rw	<p>CPU selection for Sleep and Standby mode</p> <p>001_B Only CPU0 can trigger power down modes.</p> <p>010_B Only CPU1 can trigger power down modes.</p> <p>111_B Entry to power down modes is unanimously decided by all the CPUs.</p> <p><i>Note: All other CPUSEL bit combinations are reserved.</i></p>
STBYEVEN	27	w	<p>Standby Entry Event configuration enable</p> <p>0_B Bit STBYEV is not updated.</p> <p>1_B Bit STBYEV can be updated.</p>
STBYEV	[30:28]	rw	<p>Standby Entry Event Configuration</p> <p>000_B Standby Entry triggered by setting PMCSRx.REQSLP register bit (Default).</p> <p>100_B Standby Entry triggered on $\overline{\text{ESR1}}$/NMI assertion.</p> <p><i>Note: All other bit combinations are reserved.</i></p>

Field	Bits	Type	Description
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. Only bits [23:16] are locked for transfer to the SCR. 0 _B The register is unlocked and can be updated 1 _B The register is locked and cannot be updated

PMSWCR2
Standby and Wake-up Control Register 2 (0EC_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK	0			RST	SMU RST	TCIN TRE Q	TCINT								
rh	r			rh	rh	rw	rw								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			SCR RST	SCR WDT	SCR ECC	BUS Y	SCRINT								
r			rh	rh	rh	rh	rh								

Field	Bits	Type	Description
SCRINT	[7:0]	r	Data exchange from Standby Controller to SCU. This bit field allows fast data exchange from SCR to SCU/CPUx. The data maybe read by CPUx consequent to an interrupt from the SCR to decode the interrupt. SCRINTEXCHG data from SCR is transferred to this register bitfield except incase of an SCR reset where the old value is retained.
BUSY	8	rh	SCR Arbiter Busy status flag 0 _B SCR RAM is accessible via QSPI interface. 1 _B SCR RAM is not accessible via QSPI as the arbiter is currently busy.

Field	Bits	Type	Description
SCRECC	9	rh	SCR RAM ECC error / reset flag 0 _B No ECC error / reset reported by SCR. 1 _B ECC error / reset was detected in SCR RAM. <i>Note: The flag is cleared when SCR is reset or when RSTST.ECCRST bit is cleared in SCR.</i>
SCRWDT	10	rh	SCR Watchdog Timer error / reset flag 0 _B No WDT error / reset reported by SCR. 1 _B WDT timer error / reset reported by SCR. <i>Note: The flag is cleared when SCR is reset or when RSTST.WDTRST bit is cleared in SCR.</i>
SCRRST	11	rh	SCR Software reset flag 0 _B No SW reset occurred in SCR. 1 _B A SW reset has occurred in SCR. <i>Note: The flag is cleared when SCR is reset or when RSTST.SOFRST bit is cleared in SCR.</i>
TCINT	[23:16]	rw	Data exchange from SCU to Standby Controller. This bitfield allows fast data exchange from SCU to SCR. The data may be read by SCR consequent to an interrupt request (TCINTREQ) from SCU/CPUx to SCR to decode the interrupt.
TCINTREQ	24	rw	SW Interrupt request from SCU to Standby Controller. Setting this bit triggers an interrupt to the 8 bit Standby controller.
SMURST	25	rh	SMU Reset indication flag 0 _B No reset was issued by SMU. 1 _B SMU issued an application or system reset.
RST	26	rh	Application or System Reset indication flag 0 _B No application or system reset occurred. 1 _B An application or system reset has occurred.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus side has no effect. 0 _B The register is unlocked and can be updated 1 _B The register is locked and cannot be updated

Field	Bits	Type	Description
0	[30:27], [15:12]	r	Reserved Read as 0; should be written with 0.

PMSWSTAT
Standby and Wake-up Status Flag Register(0CC_H)
Reset Value: 0001 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	0		0	ESR OTRIST		0		0	PINB WKEN	PINA WKEN	ESR 1WKEN	SCR WKEN	SCR	SCR WKP	SCR ST
	r		r	rh		r		r	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIST	STBYRAM		HWCFGEVR		POR STDF	0	PINB OVRUN	PINB WKP	PINA OVRUN	PINA WKP	ESR 1OVRUN	ESR 1WKP	0	0	
rh	rh		rh		rh	r	rh	rh	rh	rh	rh	rh	r	r	

Field	Bits	Type	Description
0	[1:0]	r	Reserved Read as 0; should be written with 0.
ESR1WKP	2	rh	ESR1 Wake-up flag 0 _B No wake-up event detected on ESR1 input. 1 _B An event as defined by PMSWCR0. ESR1EDCON detected on ESR1 input.
ESR1OVRUN	3	rh	ESR1 Overrun status flag This flag indicates that an ESR1 wake-up event occurred while the last one was not yet serviced. 0 _B No overrun condition detected on ESR1 input. 1 _B An overrun condition detected on ESR1 input.
PINAWKP	4	rh	Pin A (P14.1) Wake-up flag 0 _B No wake-up event detected on Pin A input. 1 _B An event as defined by PMSWCR0. PINAEDCON detected on Pin A input.

Field	Bits	Type	Description
PINAOVRUN	5	rh	Pin A Overrun status flag This flag indicates that an PIN A wake-up event occurred while the last one was not yet serviced. 0_B No overrun condition detected on Pin A input. 1_B An overrun condition detected on Pin A input.
PINBWKP	6	rh	Pin B (P15.1) Wake-up flag 0_B No wake-up event occurred on the Pin B input. 1_B An event as defined by PMSWCR0.PINBEDCON detected on Pin B input.
PINBOVRUN	7	rh	Pin B Overrun status flag This flag indicates that an PIN B wake-up event occurred while the last one was not yet serviced. 0_B No overrun condition detected on Pin B input. 1_B An overrun condition detected on Pin B input.
0	8	r	Reserved Read as 0; should be written with 0.
PORSTDF	9	rh	PORST Digital Filter status This bit field indicates whether additional $\overline{\text{PORST}}$ digital filter is activated. This bit is updated when PMSWCR0.PORSTDF is set. 0_B $\overline{\text{PORST}}$ recognition delay = Analog $\overline{\text{PORST}}$ pad filter delay (default reset state). 1_B $\overline{\text{PORST}}$ recognition delay = Analog $\overline{\text{PORST}}$ pad filter delay + Digital filter delay.
HWCFGGEVR	[12:10]	rh	EVR Hardware Configuration This bit field indicates the supply configuration latched by the EVR from HWCFG[2:0] during a cold startup. The latched configuration is used during STANDBY-RUN transition to reselect EVR mode. HWCFG[0] is tied to "1" in case of TC22x. .
STBYRAM	[14:13]	rh	Standby RAM Supply status This bit field indicates whether Standby RAM is supplied during Standby Mode and to infer status after a wake-up event. This bit is updated when PMSWCR0.STBYRAMSEL is set. 00_B Standby RAM is not supplied. 01_B Standby RAM (CPU0 DMI) is supplied. $1x_B$ Reserved, do not use this combination.

Field	Bits	Type	Description
TRIST	15	rh	<p>Pad Tristate / Pull-up status</p> <p>This bit indicates whether pads are configured as inputs with weak pull-up or as tristate during/after reset or after wake-up. At start-up, the value latched from HWCFG[6] pin decides the default state. This bit is updated when PMSWCR0.TRISTREQ is set.</p> <p>0_B Pads configured as inputs with weak pull-up. 1_B Pads are in tristate.</p>
SCRST	16	rh	<p>Standby Controller Reset Indication flag</p> <p>This bit is set after a power-on reset as SCR is in reset state.</p> <p>0_B No reset of Standby controller took place. 1_B Reset of Standby controller took place.</p>
SCRWKP	17	rh	<p>SCR Wake-up flag</p> <p>In case wake-up is triggered by SCR of the main controller, this flag is set.</p> <p>0_B No wake-up event has taken place. 1_B A wake-up event occurred.</p>
SCR	18	rh	<p>Standby Controller status</p> <p>This bit indicates whether SCR is enabled. This bit is updated when PMSWCR0.SCREN bit is set.</p> <p>0_B 8 bit Standby Controller is disabled 1_B 8 bit Standby Controller is enabled</p>
SCRWKEN	19	rh	<p>Standby Controller Wake-up Enable status</p> <p>This bit indicates that SCR is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.SCRWKEN bit is set.</p> <p>0_B Wake-up from Standby via SCR is disabled. 1_B Wake-up from Standby via SCR is enabled.</p>
ESR1WKEN	20	rh	<p>ESR1 Wake-up enable status</p> <p>This bit indicates that ESR1 is enabled to trigger wake-up from Standby. This bit is updated when PMSWCR0.ESR1WKEN bit is set.</p> <p>0_B Wake-up from Standby via ESR1 is disabled. 1_B Wake-up from Standby via ESR1 is enabled.</p>

Field	Bits	Type	Description
PINAWKEN	21	rh	Pin A Wake-up enable status This bit indicates that PINA is enabled to trigger wake-up from Standby. This bit is updated when PMSWCRO.PINAWKEN bit is set. 0 _B Wake-up from Standby via PINA is disabled. 1 _B Wake-up from Standby via PINA is enabled.
PINBWKEN	22	rh	Pin B Wake-up enable status This bit indicates that PINB is enabled to trigger wake-up from Standby. This bit is updated when PMSWCRO.PINBWKEN bit is set. 0 _B Wake-up from Standby via PINB is disabled. 1 _B Wake-up from Standby via PINB is enabled.
0	[26:23]	r	Reserved Read as 0; should be written with 0.
ESR0TRIST	27	rh	ESR0 pin status during Standby This bit indicates if ESR0 pin is configured as reset output or tristate during Standby mode & transitions. This bit is updated when PMSWCRO.ESR0TRIST is set. 0 _B ESR0 configured as reset output and is held low during Standby state (default reset state) 1 _B ESR0 in tristate during Standby state.
0	28	r	Reserved Read as 0; should be written with 0.
0	[31:29]	r	Reserved Read as 0; should be written with 0.

PMSWSTATCLR
Standby and Wake-up Status Clear Register(0D0_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0								0				0	SCR WKP CLR	SCR STC LR		
r								r				r	w	w		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0								0	PINB OVR UNC LR	PINB WKP CLR	PINA OVR UNC LR	PINA WKP CLR	ESR 1OV RUN CLR	ESR 1WK PCL R	0	0
r								r	w	w	w	w	w	r	r	

Field	Bits	Type	Description
0	[1:0]	r	Reserved Read as 0; should be written with 0.
ESR1WKPCLR	2	w	ESR1 Wake-up indication flag clear 0 _B No action 1 _B PMSWSTAT.ESR1WKP bit cleared.
ESR1OVRUNC LR	3	w	ESR1 Overrun status indication flag clear 0 _B No action 1 _B PMSWSTAT.ESR1OVRUN bit cleared.
PINAWKPCLR	4	w	PINA Wake-up indication flag clear 0 _B No action 1 _B PMSWSTAT.PINAWKP bit cleared.
PINAOVRUNC LR	5	w	PINA Overrun status indication flag clear 0 _B No action 1 _B PMSWSTAT.PINAOVRUN bit cleared.
PINBWKPCLR	6	w	PINB Wake-up indication flag clear 0 _B No action 1 _B PMSWSTAT.PINBWKP bit cleared.
PINBOVRUNC LR	7	w	PINB Overrun status indication flag clear 0 _B No action 1 _B PMSWSTAT.PINBOVRUN bit cleared.
0	8	r	Reserved Read as 0; should be written with 0.

Field	Bits	Type	Description
0	[15:9]	r	Reserved Read as 0; should be written with 0.
SCRSTCLR	16	w	Standby controller SCRST indication flag clear 0 _B No action 1 _B PMSWSTAT.SCRSTCLR bit cleared.
SCRWKPCLR	17	w	SCR Wake-up indication flag clear 0 _B No action 1 _B PMSWSTAT.SCRWKP bit cleared.
0	18	r	Reserved Read as 0; should be written with 0.
0	[23:19]	r	Reserved Read as 0; should be written with 0.
0	[31:24]	r	Reserved Read as 0; should be written with 0.

7.3.3 Power Management Register Address

Table 7-22 Registers Address Spaces - PMC Kernel Registers

Module	Base Address	End Address	Note
SCU	F003 6000 _H	F003 63FF _H	-

7.3.4 PMC Kernel Registers

This section describes the kernel registers of the PMC module.

PMC Kernel Register Overview

Table 7-23 Overview of PMC Registers (Offset from SCU Register Base)

Short Name	Long Name	Offset Addr. 1)	Access Mode		Reset	Description See
			Read	Write		
EVR_RSTCON	EVR Reset Control Register	06C _H	U, SV	SV, SE, P	Cold Power-on Reset ²⁾	Page 7-169
EVR_STAT	EVR Status Register	0B0 _H	U, SV	BE	Cold Power-on Reset	Page 7-165
EVR_DVSTAT	EVR Status Register for Voltage Scaling	0B4 _H	U, SV	BE	Cold Power-on Reset	Page 7-181
-	Reserved	0B4 _H	-	BE	-	-
EVR13CON	EVR13 Control Register	0B8 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-177
EVR33CON	EVR33 Control Register	0BC _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-178
PMSWCR0	Standby and Wakeup Control Register 0	0C8 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-207
PMSWSTAT	Standby and Wakeup Status	0CC _H	U, SV	BE	Cold Power-on Reset	Page 7-216

Table 7-23 Overview of PMC Registers (Offset from SCU Register Base)

Short Name	Long Name	Offset Addr. 1)	Access Mode		Reset	Description See
			Read	Write		
PMSWSTATCLR	Standby and Wakeup Status Clear	0D0 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-220
PMCSR0	CPU0 Power Management Control and Status Register	0D4 _H	U, SV	SE, CE0, SV, P	Application Reset	Page 7-205
PMCSR1	CPU1 Power Management Control and Status Register	0D8 _H	U, SV	SE, CE1, SV, P	Application Reset	Page 7-205
PMSWCR1	Standby and Wakeup Control Register 1	0E8 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-211
PMSWCR2	Standby and Wakeup Control Register 2	0EC _H	U, SV	SV, P	Power-on Reset	Page 7-214
EVRTRIM	EVR Trim Register	198 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-179
EVRADCSTAT	EVR ADC Status Register	19C _H	U, SV	BE	Power-on Reset	Page 7-168
–	Reserved	198 _H	nE	nE	–	–
EVRUVMON	EVR Undervoltage Monitor	1A0 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-173
EVROVMON	EVR Overvoltage Monitor	1A4 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-172
EVRMONCTRL	EVR Monitor Control Register	1A8 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-174
–	Reserved	1AC _H	nE	nE	–	–

Table 7-23 Overview of PMC Registers (Offset from SCU Register Base)

Short Name	Long Name	Offset Addr. 1)	Access Mode		Reset	Description See
			Read	Write		
EVRSDCTRL1	EVR SD Control Register 1	1B0 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-182
EVRSDCTRL2	EVR SD Control Register 2	1B4 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-183
EVRSDCTRL3	EVR SD Control Register 3	1B8 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-184
EVRSDCTRL4	EVR SD Control Register 4	1BC _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-185
EVRSDCOEFF1	EVR SD Coefficient Register 1	1C0 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-186
EVRSDCOEFF2	EVR SD Coefficient Register 2	1C4 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-187
EVRSDCOEFF3	EVR SD Coefficient Register 3	1C8 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-188
EVRSDCOEFF4	EVR SD Coefficient Register 4	1CC _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-189
EVRSDCOEFF5	EVR SD Coefficient Register 5	1D0 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-190
EVRSDCOEFF6	EVR SD Coefficient Register 6	1D4 _H	U, SV	SV, SE, P	Cold Power-on Reset	Page 7-191
–	Reserved	300 _H	nE	nE	–	–
–	Reserved	304 _H - 37C _H	BE	BE	–	–

- 1) The absolute register address is calculated as follows:
Module Base Address + Offset Address (shown in this column)
- 2) All EVR registers excluding status registers have corresponding shadow registers in the EVR Standby domain. After a cold PORST, a read of these registers may return the reset value or the value updated by the Firmware. The reset value reflects the default isolation value in the shadow register. Otherwise, a read will provide the value of the most recent write operation.

7.4 System Control Unit (SCU)

The System Control Unit (SCU) of the TC26x B-Step contains miscellaneous control registers associated with other system functions.

This includes:

- External Request and Cross-triggering Unit (ERU) (see [Section 7.4.1](#))
- CPU Lockstep Comparator Logic (LCL) (see [Section 7.4.2](#))
- Die Temperature Sensor (DTS) (see [Section 7.4.3](#))
- Watchdog Timers (WDTx) (see [Section 7.4.4](#))
- Emergency Stop (EMS) (see [Section 7.4.5](#))
- Logic Built-in-Self-Test (LBIST) (see [Section 7.4.6](#))
- Overlay Control (OVC) (see [Section 7.4.7](#))
- Miscellaneous System Control Registers (see [Section 7.4.8](#))
- SCU register overview table (see [Table 7-28](#))

7.4.1 External Request Unit (ERU)

The External Request Unit (ERU) is a versatile event and pattern detection unit. Its major task is the **generation of interrupts based on selectable trigger events at different inputs**, e.g. to generate external interrupt requests if an edge occurs at an input pin. The detected events can also be used by other modules to trigger or to gate module-specific actions.

7.4.1.1 Introduction

The ERU of the TC26x B-Step can be split in three main functional parts:

- 8 independent **Input Channels x** for input selection and conditioning of trigger or gating functions
- Event distribution: A **Connecting Matrix** defines the events of the Input Channel x that lead to a reaction of an Output Channel y.
- 8 independent **Output Channels y** for combination of events, definition of their effects and distribution to the system (interrupt generation, ...)

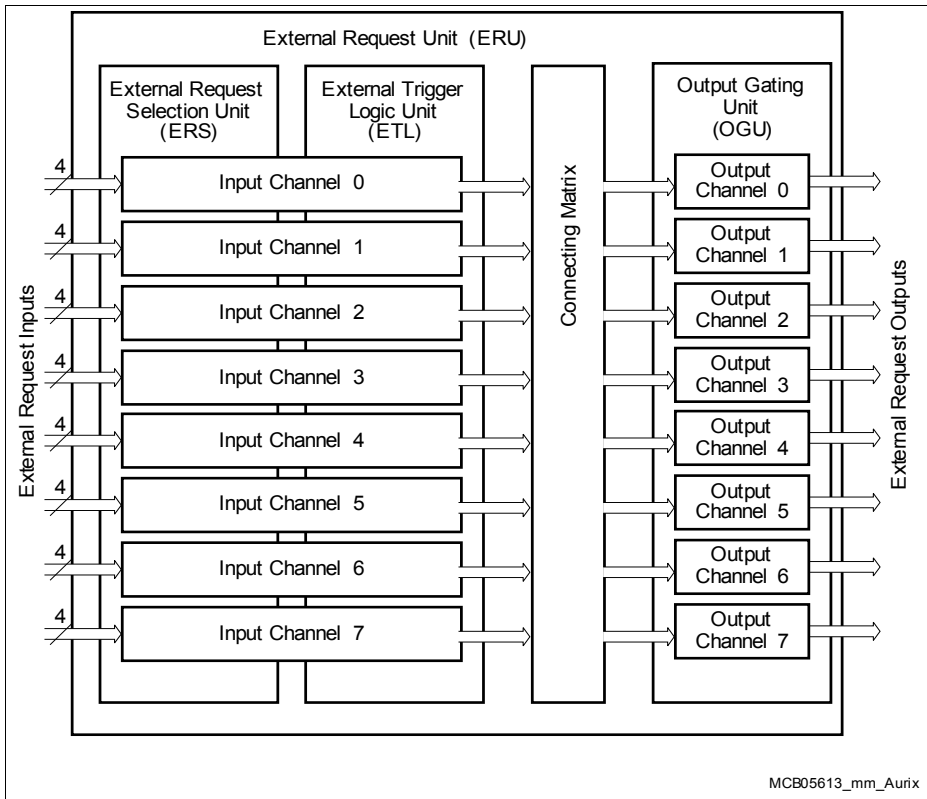


Figure 7-39 External Request Unit Overview

These tasks are handled by the following building blocks:

- An **External Request Select Unit (ERSx)** per Input Channel allows the selection of one input vector out of the 4 possible available inputs.
- An **Event Trigger Logic (ETLx)** per Input Channel allows the definition of the transition (edge selection, or by software) that lead to a trigger event and can also store this status. Here, the input levels of the selected signals are translated into events (event detected = event flag becomes set, independent of the polarity of the original input signals).
- The **Connecting Matrix** distributes the events and status flags generated by the Input Channels to the Output Channels.
- An **Output Gating Unit (OGUy)** per Output Channel that combines the available trigger events and status information from the Input Channels. An event of one Input Channel can lead to reactions of several Output Channels, or also events of several

Input Channels can be combined to a reaction of one Output Channel (pattern detection).

Different types of reactions are possible, e.g. interrupt generation (based on signals ERU_IOUTy).

7.4.1.2 ERU Input Pin Connections

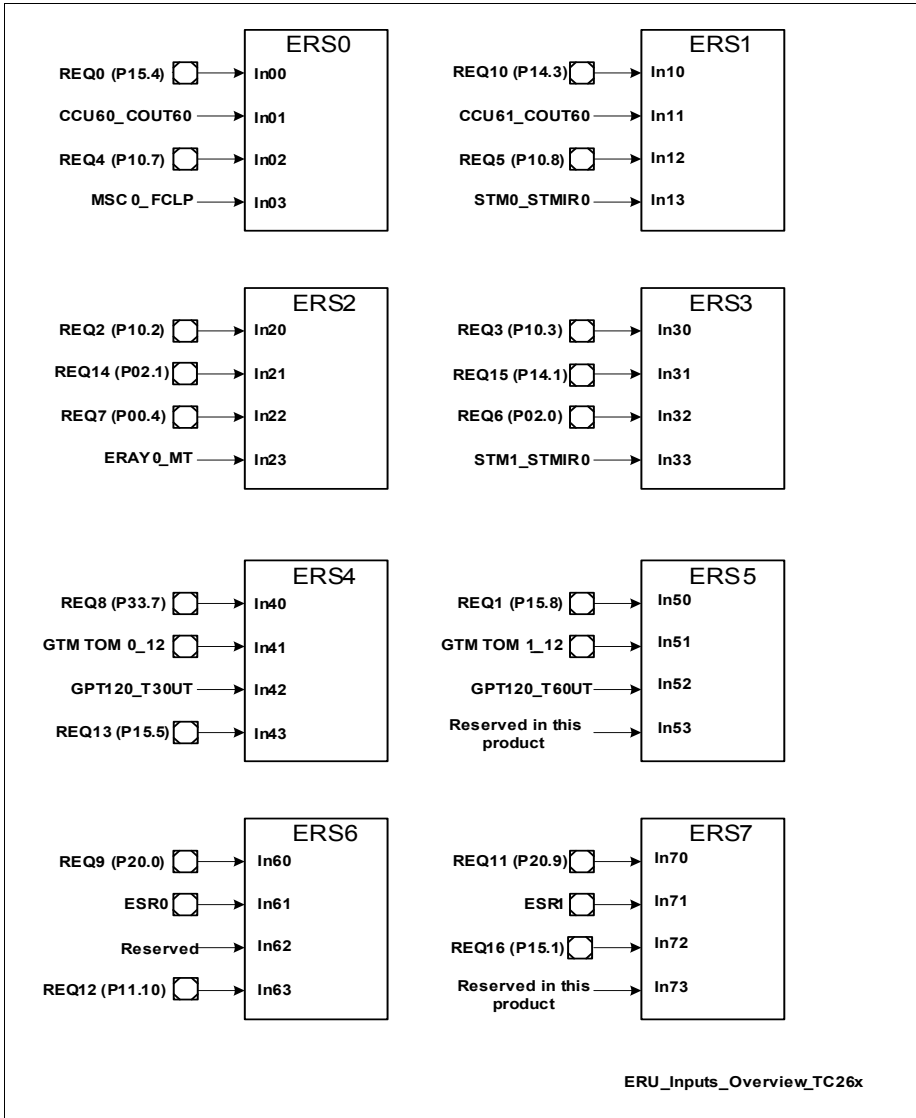


Figure 7-40 External Request Unit Input Connections for TC26x

The inputs to the ERU can be selected from a large number of input signals. 16 of these inputs come directly from input REQx ports, but other inputs come from various peripheral module status signals. Usually, such inputs would be selected for an ERU function when the module input function is not used by the application, or when the module is not used at all. However, it is also possible to select an input which is also used by the other module, to also be used in the ERU as a trigger or to be combined with other signals (e.g. to generate an interrupt trigger when a start-of-frame is detected on a selected communication interface input).

7.4.1.3 External Request Selector Unit (ERS)

Each ERS selects one of four inputs as the one input signal of the respective input channel. **Figure 7-41** shows the structure of this block.

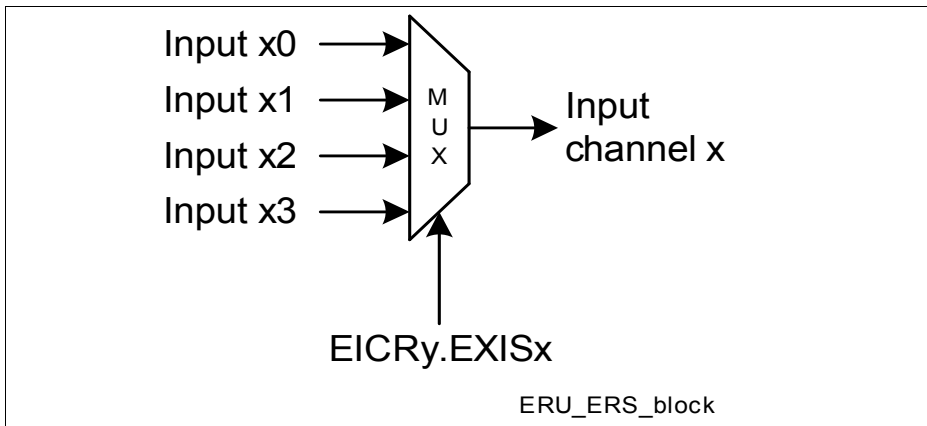


Figure 7-41 External Request Select Unit Overview

The ERS unit for channel x is controlled via bit field EICRy.EXISx.

7.4.1.4 Event Trigger Logic (ETL)

For each Input Channel x, an event trigger logic ETLx derives a trigger event and a status from the input channel x delivered by the associated ERSx unit. Each ETLx is based on an edge detection block, where the detection of a rising or a falling edge can be individually enabled. Both edges lead to a trigger event if both enable bits are set (e.g. to handle a toggling input).

Each pair of the four ETL units has an associated EICRy register, that controls all options of an ETL (the register also holds control bits for the associated ERS unit pair, e.g. EICR0 to control ESR0 and ESR1 plus and ETL0 and ETL1).

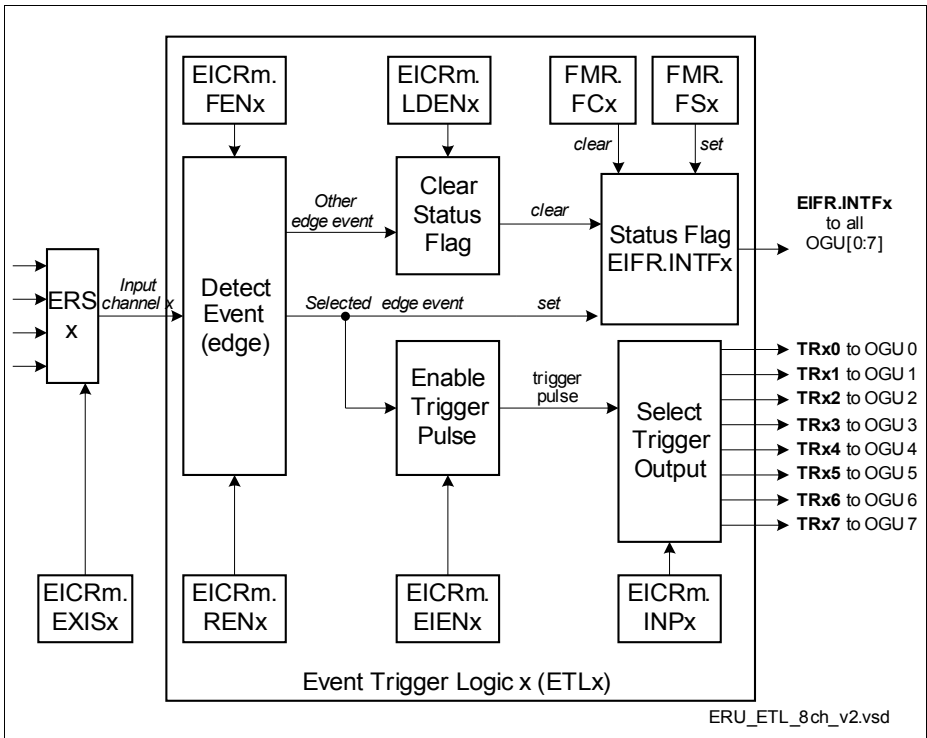


Figure 7-42 Event Trigger Logic Overview

When the selected event (edge) is detected, the status flag EIFR.INTFx becomes set. The status flag is cleared automatically if the “opposite” event is detected, if so enabled via bit EICRy.LDENx = 1. For example, if only the falling edge detection is enabled to set the status flag, it is cleared when the rising edge is detected. In this mode, it can be used for pattern detection where the actual status of the input is important (enabling both edge detections is not useful in this mode).

The output of the status flag is connected to all following Output Gating Units (OGUz) in parallel (see [Figure 7-43](#)) to provide **pattern detection capability of all OGUz** units based on different or the same status flags.

In addition to the modification of the status flag, a trigger pulse output TRxz of ETLx can be enabled (by bit EICRy.EIENx) and selected to **trigger actions in one of the OGUz** units. The target OGUz for the trigger is selected by bit field EICRy.INPx.

The trigger becomes active when the selected edge event is detected, independently from the status flag EIFR.INTFx.

7.4.1.5 Connecting Matrix

The connecting matrix distributes the trigger signals (TRxy) and status signals (EIFR.INPFx) from the different ETLx units between the OGUy units. [Figure 7-43](#) provides a complete overview of the connections between the ETLx and the OGUz units.

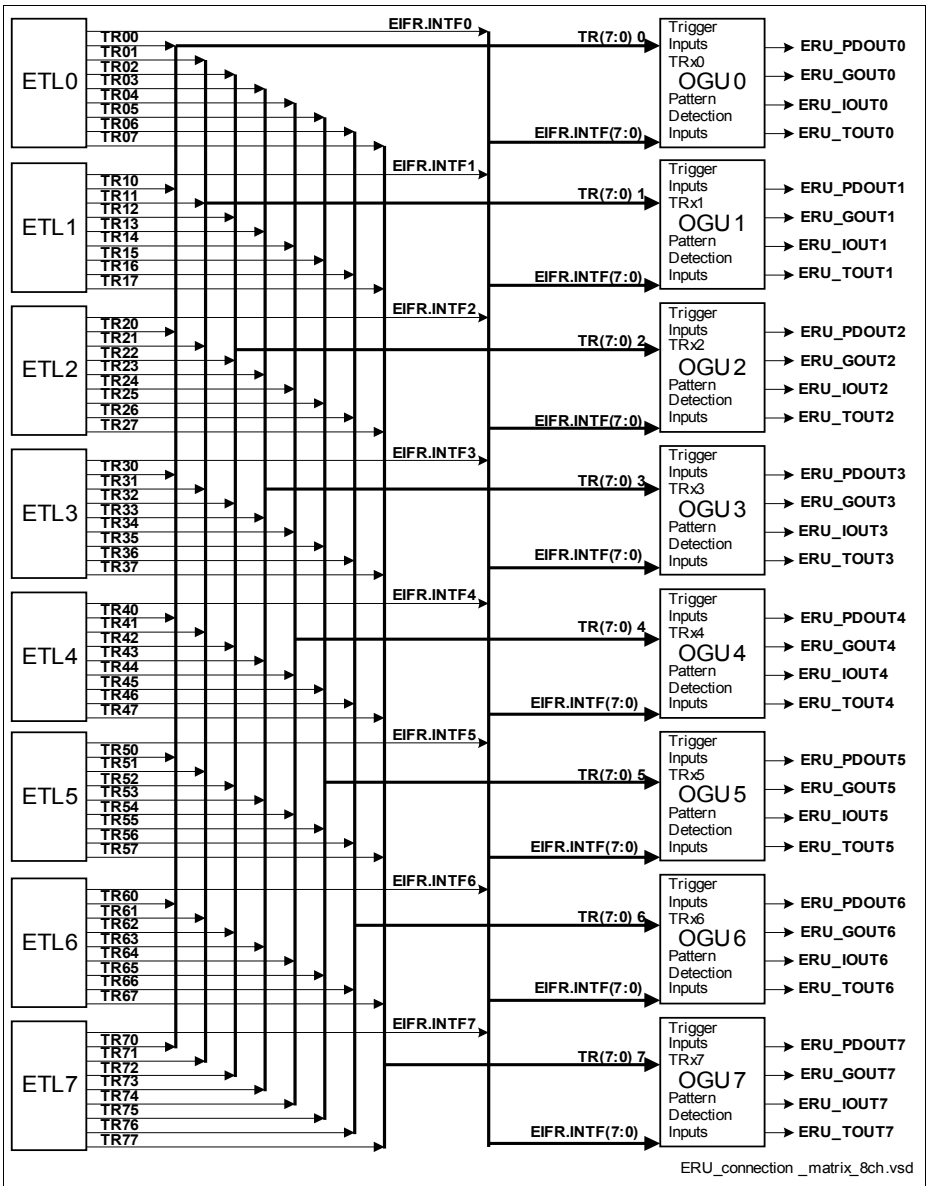


Figure 7-43 Connecting Matrix between ETLx and OGUy

7.4.1.6 Output Gating Unit (OGU)

Each OGUy unit combines the available trigger events and status flags from the Input Channels and distributes the results to the system. [Figure 7-44](#) illustrates the logic blocks within an OGUy unit. All functions of an OGUy unit are controlled by the associated IGCRm registers, one for each pair of output channels e.g. IGCR1 for OGU2 and OGU3. The function of an OGUy unit can be split into two parts:

- **Trigger combination:**
All trigger signals TRxy from the Input Channels that are enabled and directed to OGUy and a pattern change event (if enabled) are logically OR-combined.
- **Pattern detection:**
The status flags EIFR.INTFx of the Input Channels can be enabled to take part in the pattern detection. A pattern match is detected while all enabled status flags are set.

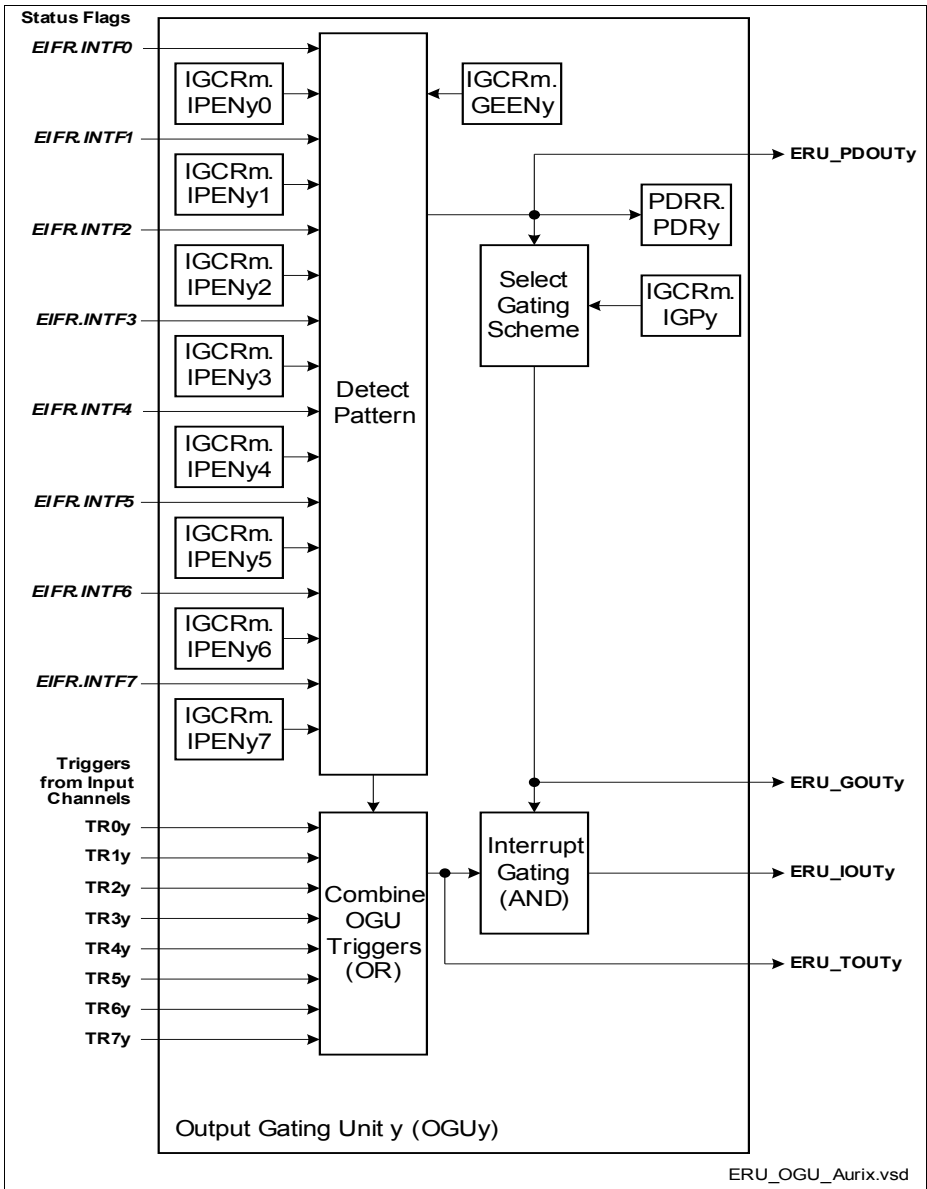


Figure 7-44 Output Gating Unit for Output Channel *y*

Each OGUy units generates 4 output signals that are distributed to the system.

- **ERU_PDOUTy** to directly output the pattern match information for gating purposes in other modules (pattern match = 1).
- **ERU_GOUTy** to output the pattern match or pattern miss information (inverted pattern match), or a permanent 0 or 1 under software control for gating purposes in other modules.
- **ERU_TOUTy** as combination of a peripheral trigger, a pattern detection result change event, or the ETLx trigger outputs TRxy to trigger actions in other modules.
- **ERU_IOUTy** as gated trigger output (ERU_GOUTy logical AND-combined with ERU_TOUTy) to trigger interrupts (e.g. the interrupt generation can be gated to allow interrupt activation during a certain time window).

Trigger Combination

The trigger combination logically OR-combines different trigger inputs to form a common trigger ERU_TOUTy. Possible trigger inputs are:

- In each ETLx unit of the **Input Channels**, the trigger output TRxy can be enabled and the trigger event can be directed to one of the OGUy units.
- In the case that at least one **pattern detection** input is enabled (IGCRm.IPENxy) and a change of the pattern detection result from pattern match to pattern miss (or vice-versa) is detected, a trigger event is generated to indicate a pattern detection result event (if enabled by IGCRm.GEENy).

The trigger combination offers the possibility to program different trigger criteria for several input signals (independently for each Input Channel) or peripheral signals, and to combine their effects to a single output, e.g. to generate an interrupt or to start an ADC conversion. This combination capability allows the generation of an interrupt per OGU that can be triggered by several inputs (multitude of request sources -> one reaction).

Pattern Detection

The pattern detection logic allows the combination of the status flags of all ETLx units. Each status flag can be individually included or excluded from the pattern detection for each OGUy, via control bits IGCRm.IPENxy. The pattern detection block outputs the following pattern detection results:

- **Pattern match** (PDRR.PDRy = 1 and ERU_PDOUTy = 1):
A pattern match is indicated while all status flags that are included in the pattern detection are 1.
- **Pattern miss** (PDRR.PDRy = 0 and ERU_PDOUTy = 0):
A pattern miss is indicated while at least one of the status flags that are included in the pattern detection is 0.

In addition, the pattern detection can deliver a trigger event if the pattern detection result changes from match to miss or vice-versa (if enabled by IGCRm.GEENy = 1). The

pattern result change event is logically OR-combined with the other enabled trigger events to support interrupt generation or to trigger other module functions (e.g. in an ADC). The event is indicated when the pattern detection result changes and PDRR.PDRy becomes updated.

The interrupt generation in the OGUy is based on the trigger ERU_TOUTy that can be gated (masked) with the pattern detection result ERU_PDOUTy. This allows an automatic and reproducible generation of interrupts during a certain time window, where the request event is elaborated by the trigger combination block and the time window information (gating) is given by the pattern detection. For example, interrupts can be issued on a regular time base while a combination of input signals occurs (pattern detection based on ETLx status bits).

Only four interrupt/service requests are generated by the ERU. The mapping of interrupt requests to OGUy blocks is shown in the ERU connection diagram.

A programmable gating scheme introduces flexibility to adapt to application requirements and allows the generation of interrupt requests ERU_IOUTy under different conditions:

- **Pattern match** (IGCRm.IGPy = 10_B):
An interrupt request is issued when a trigger event occurs while the pattern detection shows a pattern match.
- **Pattern miss** (IGCRm.IGPy = 11_B):
An interrupt request is issued when the trigger event occurs while the pattern detection shows a pattern miss.
- **Independent** of pattern detection (IGCRm.IGPy = 01_B):
In this mode, each occurring trigger event leads to an interrupt request. The pattern detection output can be used independently from the trigger combination for gating purposes of other peripherals (independent use of ERU_TOUTy and ERU_PDOUTy with interrupt requests on trigger events).
- **No interrupts** (IGCRm.IGPy = 00_B, default setting)
In this mode, an occurring trigger event does not lead to an interrupt request. The pattern detection output can be used independently from the trigger combination for gating purposes of other peripherals (independent use of ERU_TOUTy and ERU_PDOUTy without interrupt requests on trigger events).

7.4.1.7 ERU Output Pin Connections

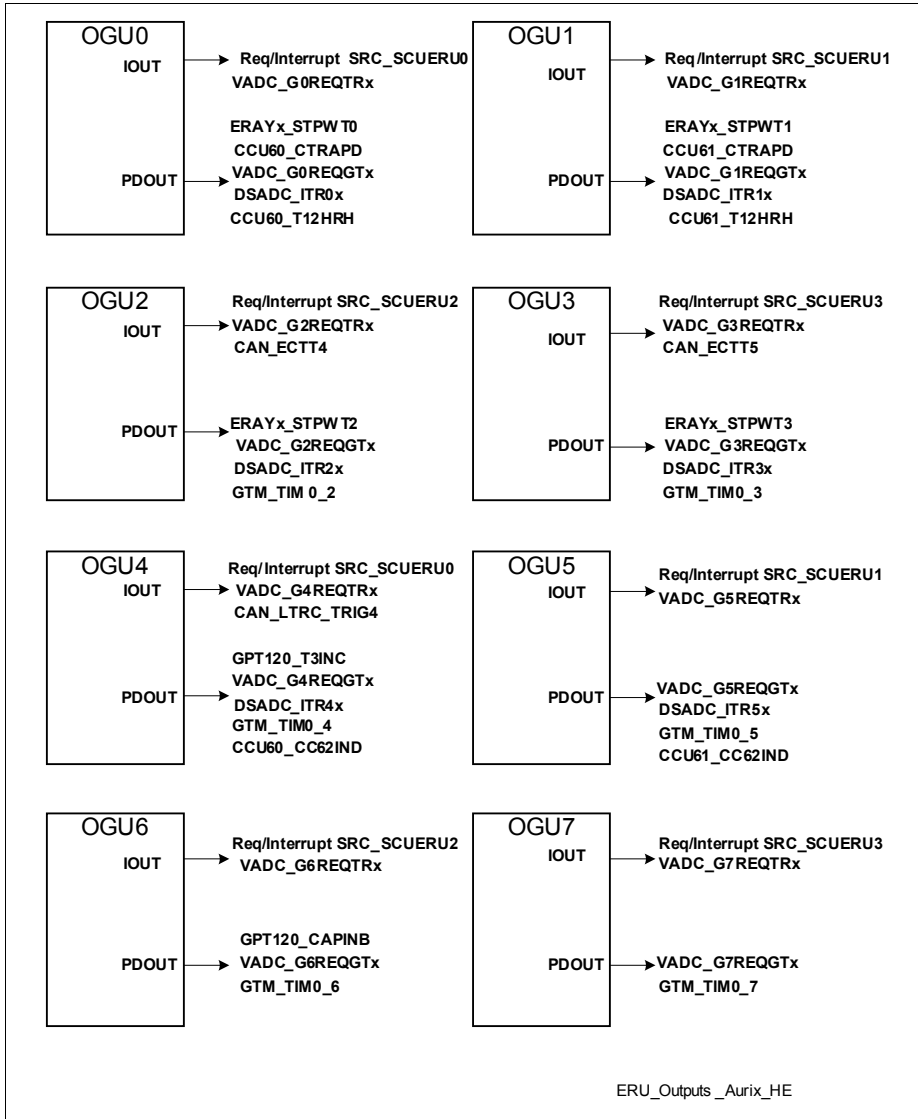


Figure 7-45 External Request Unit Output Connections for TC26x

7.4.1.8 External Request Unit Registers

The External Input Channel Registers EICR_i (i=0 to 3) for the 4 external input channels contain bits to configure the external request selection ERS and the event trigger logic ETL.

EICR0

External Input Channel Register 0 (210_H) Reset Value: 0000 0000_H

EICR1

External Input Channel Register 1 (214_H) Reset Value: 0000 0000_H

EICR2

External Input Channel Register 2 (218_H) Reset Value: 0000 0000_H

EICR3

External Input Channel Register 3 (21C_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	INP1			EI EN1	LD EN1	R EN1	F EN1	0	EXIS1			0			
r	rw			rw	rw	rw	rw	r	rw			r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	INP0			EI EN0	LD EN0	R EN0	F EN0	0	EXIS0			0			
r	rw			rw	rw	rw	rw	r	rw			r			

Field	Bits	Type	Description
EXIS0	[6:4]	rw	External Input Selection 0 This bit field determines which input line is selected for Input Channel (2i). 000 _B Input (2i) 0 is selected 001 _B Input (2i) 1 is selected 010 _B Input (2i) 2 is selected 011 _B Input (2i) 3 is selected 100 _B Reserved 101 _B Reserved 110 _B Reserved 111 _B Reserved

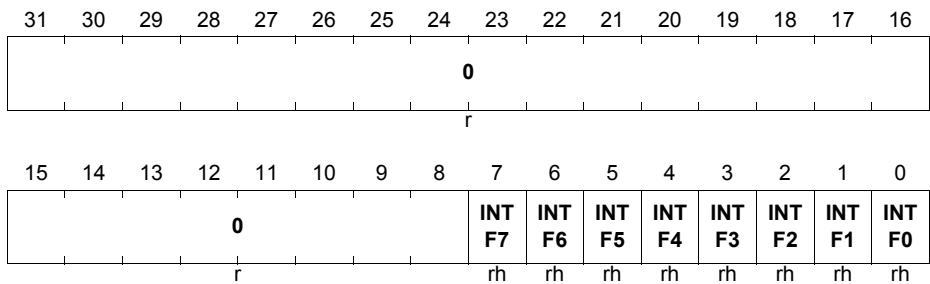
Field	Bits	Type	Description
FEN0	8	rw	<p>Falling Edge Enable 0</p> <p>This bit determines if the falling edge of Input Channel (2i) is used to set bit INTF(2i).</p> <p>0_B The falling edge is not used</p> <p>1_B The detection of a falling edge of Input Channel 0 generates a trigger event. INTF(2i) becomes set.</p>
REN0	9	rw	<p>Rising Edge Enable 0</p> <p>This bit determines if the rising edge of Input Channel (2*i) is used to set bit INTF(2i).</p> <p>0_B The rising edge is not used</p> <p>1_B The detection of a rising edge of Input Channel (2*i) generates a trigger event. INTF(2*i) becomes set</p>
LDEN0	10	rw	<p>Level Detection Enable 0</p> <p>This bit determines if bit INTF(2i) is cleared automatically if an edge of the input Input Channel (2i) is detected, which has not been selected (rising edge with REN0 = 0 or falling edge with FEN0 = 0).</p> <p>0_B Bit INTF(2i) will not be cleared</p> <p>1_B Bit INTF(2i) will be cleared</p>
EIEN0	11	rw	<p>External Input Enable 0</p> <p>This bit enables the generation of a trigger event for request channel (2i) (e.g. for interrupt generation) when a selected edge is detected.</p> <p>0_B The trigger event is disabled</p> <p>1_B The trigger event is enabled</p>

Field	Bits	Type	Description
INP0	[14:12]	rw	Input Node Pointer This bit field determines the destination (output channel) for trigger event (2i) (if enabled by EIEN(2i)). 000 _B An event from input ETL 2i triggers output OGU0 (signal TR(2i) 0) 001 _B An event from input ETL 2i triggers output OGU1 (signal TR(2i) 1) 010 _B An event from input ETL 2i triggers output OGU2 (signal TR(2i) 2) 011 _B An event from input ETL 2i triggers output OGU3 (signal TR(2i) 3) 100 _B An event from input ETL 2i triggers output OGU4 (signal TR(2i) 0) 101 _B An event from input ETL 2i triggers output OGU5 (signal TR(2i) 0) 110 _B An event from input ETL 2i triggers output OGU6 (signal TR(2i) 0) 111 _B An event from input ETL 2i triggers output OGU7 (signal TR(2i) 0)
EXIS1	[22:20]	rw	External Input Selection 1 This bit field determines which input line is selected for Input Channel (2i+1). 000 _B Input (2i+1) 0 is selected 001 _B Input (2i+1) 1 is selected 010 _B Input (2i+1) 2 is selected 011 _B Input (2i+1) 3 is selected 100 _B Reserved 101 _B Reserved 110 _B Reserved 111 _B Reserved
FEN1	24	rw	Falling Edge Enable 1 This bit determines if the falling edge of Input Channel (2i+1) is used to set bit INTF(2i+1). 0 _B The falling edge is not used 1 _B The detection of a falling edge of Input Channel 1 generates a trigger event. INTF(2i+1) becomes set

Field	Bits	Type	Description
REN1	25	rw	Rising Edge Enable 1 This bit determines if the rising edge of Input Channel (2i+1) is used to set bit INTF(2i+1). 0 _B The rising edge is not used 1 _B The detection of a rising edge of Input Channel 1 generates a trigger event . INTF(2i+1) becomes set
LDEN1	26	rw	Level Detection Enable 1 This bit determines if bit INTF(2i+1) is cleared automatically if an edge of the input Input Channel (2i+1) is detected, which has not been selected (rising edge with REN1 = 0 or falling edge with FEN1 = 0). 0 _B Bit INTF(2i+1) will not be cleared 1 _B Bit INTF1(2i+1) will be cleared
EIEN1	27	rw	External Input Enable 1 This bit enables the generation of a trigger event for request channel (2i+1) (e.g. for interrupt generation) when a selected edge is detected. 0 _B The trigger event is disabled 1 _B The trigger event is enabled
INP1	[30:28]	rw	Input Node Pointer This bit field determines the destination (output channel) for trigger event (2i+1) (if enabled by EIEN(2i+1)). 000 _B An event from input ETL 2i+1 triggers output OGU0 (signal TR(2i+1) 0) 001 _B An event from input ETL 2i+1 triggers output OGU1 (signal TR(2i+1) 1) 010 _B An event from input ETL 2i+1 triggers output OGU2 (signal TR(2i+1) 2) 011 _B An event from input ETL 2i+1 triggers output OGU3 (signal TR(2i+1) 3) 100 _B An event from input ETL 2i+1 triggers output OGU4 (signal TR(2i+1) 0) 101 _B An event from input ETL 2i+1 triggers output OGU5 (signal TR(2i+1) 0) 110 _B An event from input ETL 2i+1 triggers output OGU6 (signal TR(2i+1) 0) 111 _B An event from input ETL 2i+1 triggers output OGU7 (signal TR(2i+1) 0)

Field	Bits	Type	Description
0	[3:0], 7, [19:15], 23, 31	r	Reserved Read as 0; should be written with 0.

The External Input Flag Register EIFR contains all status flags for the external input channels. The bits in this register can be cleared by software by setting FMR.FCx, and set by setting FMR.FSx.

EIFR
External Input Flag Register
(220_H)
Reset Value: 0000 0000_H


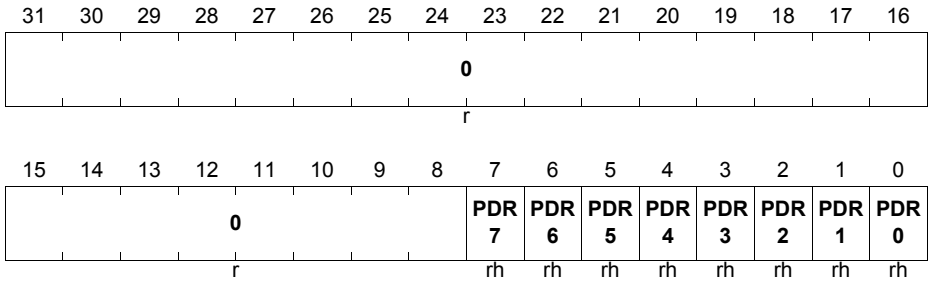
Field	Bits	Type	Description
INTFx (x = 0-7)	x	rh	External Event Flag of Channel x This bit monitors the status flag of the event trigger condition for the input channel x. This bit is automatically cleared when the selected condition (see RENx, FENx) is no longer met (if LDENx = 1) or remains set until it is cleared by software (if LDENx = 0).
0	[31:8]	r	Reserved Read as 0; should be written with 0.

FMR
Flag Modification Register
(224_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								FC7	FC6	FC5	FC4	FC3	FC2	FC1	FC0
r								w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								FS7	FS6	FS5	FS4	FS3	FS2	FS1	FS0
r								w	w	w	w	w	w	w	w

Field	Bits	Type	Description
FSx (x = 0-7)	x	w	Set Flag INTFx for Channel x Setting this bit will set the corresponding bit INTFx in register EIFR. Reading this bit always delivers a 0. If both FSx and FCx are set in the same access then the bit x in register EIFR is not modified. 0 _B The bit x in register EIFR is not modified 1 _B The bit x in register EIFR is set
FCx (x = 0-7)	16 + x	w	Clear Flag INTFx for Channel x Setting this bit will clear the corresponding bit INTFx in register EIFR. Reading this bit always delivers a 0. If both FSx and FCx are set in the same access then the bit x in register EIFR is not modified. 0 _B The bit x in register EIFR is not modified 1 _B The bit x in register EIFR is cleared
0	[15:8], [31:24]	r	Reserved Read as 0; should be written with 0.

The Pattern Detection Result Register monitors the combinatorial output status of the pattern detection units.

PDRR
Pattern Detection Result Register
(228_H)
Reset Value: 0000 00FF_H


Field	Bits	Type	Description
PDR_y (y = 0-7)	y	rh	Pattern Detection Result of Channel y This bit monitors the output status of the pattern detection for the output channel y.
0	[31:8]	r	Reserved Read as 0; should be written with 0.

The Interrupt Gating Control Registers IGCR_j (j=0 to 3) contain bits to enable the pattern detection and to control the gating for the output channels.

IGCR0
Flag Gating Register 0 (22C_H) **Reset Value: 0000 0000_H**
IGCR1
Flag Gating Register 1 (230_H) **Reset Value: 0000 0000_H**
IGCR2
Flag Gating Register 2 (234_H) **Reset Value: 0000 0000_H**
IGCR3
Flag Gating Register 3 (238_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IGP1		GE EN1		0				IPEN 17	IPEN 16	IPEN 15	IPEN 14	IPEN 13	IPEN 12	IPEN 11	IPEN 10
rw		rw		r				rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IGP0		GE EN0		0				IPEN 07	IPEN 06	IPEN 05	IPEN 04	IPEN 03	IPEN 02	IPEN 01	IPEN 00
rw		rw		r				rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
IPEN0x (x = 0-7)	x	rw	Flag Pattern Enable for Channel 0 Bit IPEN0x determines if the flag INTF _x of channel x takes part in the pattern detection for the gating of the requests for the output signals GOUT(2 ^{*j}) and IOUT(2 ^{*j}). 0 _B The bit INTF _x does not take part in the pattern detection 1 _B The bit INTF _x is taken into consideration for the pattern detection
GEEN0	13	rw	Generate Event Enable 0 Bit GEEN0 enables the generation of a trigger event for output channel (2 ^{*j}) when the result of the pattern detection changes. When using this feature, a trigger (e.g. for an interrupt) is generated during the first clock cycle when a pattern is detected or when it is no longer detected. 0 _B The trigger generation at a change of the pattern detection result is disabled 1 _B The trigger generation at a change of the pattern detection result is enabled

Field	Bits	Type	Description
IGPO	[15:14]	rw	<p>Interrupt Gating Pattern 0</p> <p>In each register IGCR_j, bit field IGPO determines how the pattern detection influences the output lines GOUT(2j) and IOUT(2j).</p> <p>00_B IOUT(2j) is inactive. The pattern is not considered.</p> <p>01_B IOUT(2j) is activated in response to a trigger event. The pattern is not considered.</p> <p>10_B The detected pattern is considered. IOUT(2j) is activated if a trigger event occurs while the pattern is present.</p> <p>11_B The detected pattern is considered. IOUT(2j) is activated if a trigger event occurs while the pattern is not present.</p>
IPEN1x (x = 0-7)	16+x	rw	<p>Interrupt Pattern Enable for Channel 1</p> <p>Bit IPEN(2j+1)x determines if the flag INTF_x of channel (2j+1) takes part in the pattern detection for the gating of the requests for the output signals GOUT(2j+1) and IOUT(2j+1).</p> <p>0_B The bit INTF_x does not take part in the pattern detection</p> <p>1_B The bit INTF_x is taken into consideration for the pattern detection</p>
GEEN1	29	rw	<p>Generate Event Enable 1</p> <p>Bit GEEN1 enables the generation of a trigger event for output channel (2j+1) when the result of the pattern detection changes. When using this feature, a trigger (e.g. for an interrupt) is generated during the first clock cycle when a pattern is detected, or when it is no longer detected.</p> <p>0_B The trigger generation at a change of the pattern detection result is disabled</p> <p>1_B The trigger generation at a change of the pattern detection result is enabled</p>

Field	Bits	Type	Description
IGP1	[31:30]	rw	<p>Interrupt Gating Pattern 1</p> <p>In each register IGCR_j, bit field IGP1 determines how the pattern detection influences the output lines GOUT(2j+1) and IOUT(2j+1).</p> <p>00_B IOUT(2j+1) is inactive. The pattern is not considered.</p> <p>01_B IOUT(2j+1) is activated in response to a trigger event. The pattern is not considered.</p> <p>10_B The detected pattern is considered. IOUT(2j+1) is activated if a trigger event occurs while the pattern is present.</p> <p>11_B The detected pattern is considered. IOUT(2j+1) is activated if a trigger event occurs while the pattern is not present.</p>
0	[12:8], [28:24]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

7.4.2 Lockstep CPU Configuration

This section contains the registers which are used to configure and enable CPU Lockstep Mode. These registers are only writeable by start-up firmware and are configured from the BMI . See the Firmware and Lockstep Control Logic chapters for further details.

7.4.2.1 Logic Monitor Control Registers

LCL CPU0 Control

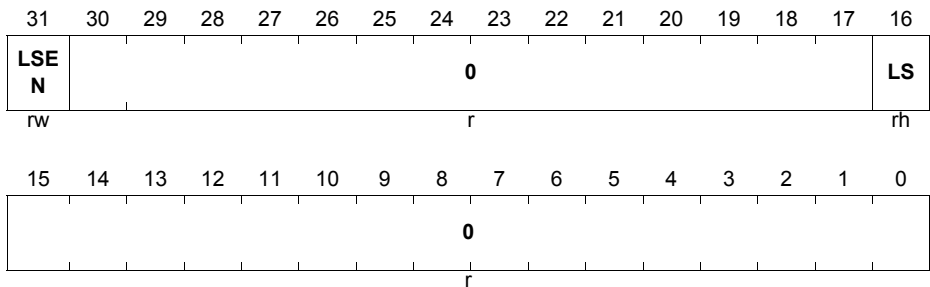
Provides control for CPU0 lockstep compare logic block.

LCLCON0

LCL CPU0 Control Register

(134_H)

Reset Value: 8000 0000_H



Field	Bits	Type	Description
0	[15:0]	r	Reserved will be read as 0 _B , should be written as 0 _B
LS	16	r	Reserved in this product will be read as 0 _B , should be written as 0 _B
0	[30:17]	r	Reserved will be read as 0 _B , should be written as 0 _B
LSEN	31	rw	Reserved in this product will be read as 1 _B , should be written as 1 _B

LCL Test Register

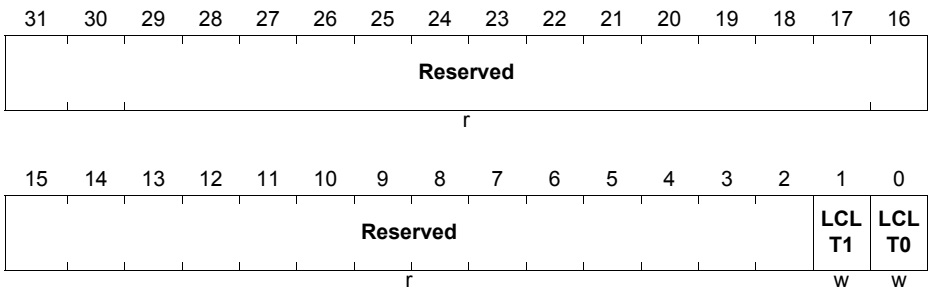
Provides the capability for software to inject a fault condition into the comparators of each lockstep compare logic block. The implementation should generate a single cycle fault each time the bit is written with '1'.

LCLTEST

LCL Test Register

(13C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
0	[31:2]	r	Reserved
LCLT0	0	r	Reserved in this product
LCLT1	1	rwh	LCL1 Lockstep Test Fault injection for LCL1. Reads as zero. 0 _B No action 1 _B Inject single fault in LCL1

Critical Register Monitoring

Safety critical SCU registers are protected by the Safety Endinit signal against accidental software writes. In addition to this, a continuous monitoring mechanism is provided to check the content of these safety critical registers have not been affected by random single bit flip-flop failures. This mechanism is controlled by the SMU.

The System Control registers with write access mode "SE" listed in [Table 7-28](#) are covered by this mechanism. This includes:

- CCU registers OSCCON, PLLCONx, PLLERAYCONx, CCUCONx, FDR, EXTCON
- RCU registers RSTCON, ARSTDIS, SWRSTCON, RSTCONx, ESRCFGx, ESROCFG, PDISC
- EVR registers EVRRSTCON, EVRSDCOEFF*, EVRSDCTRL*, EVRTRIM, EVR13CON, EVR33CON
- PMC registers PMSWCRx, PMCSRx, PMSWSTATCLR
- Emergency Stop register EMSR
- Registers controlling safety monitors WDTSCONx1, WDTCPU*CON1, LBISTCTRLx, EVRUVMON, EVROVMON
- MTU registers MEMTEST, MEMMAP, MEMSTAT, ECCS and RAR

Any single bit flipflop failures will be reported via an SMU alarm.

7.4.3 Die Temperature Measurement

The Die Temperature Sensor (DTS) generates a measurement result that indicates directly the current temperature. The result of the measurement is displayed via bit field DTSSTAT.RESULT. In order to start one measurement bit DTSCON.START needs to be set.

The DTS has to be enabled before it can be used via bit DTSCON.PWD. When the DTS is powered after the start-up time of the DTS (defined in the Data Sheet) a temperature measurement can be started.

Note: If bit field DTSSTAT.RESULT is read before the first measurement was finished it will return 0x000.

When a measurement is started the result is available after the measurement time passed. If the DTS is ready to start a measurement can be checked via bit DTSSTAT.RDY. If a started measurement is finished or still in progress is indicated via the status bit DTSSTAT.BUSY. The measurement time is also defined in the Data Sheet.

In order to adjust production variations bit field DTSCON.CAL is available. DTSCON.CAL is automatically programmed with a predefined value.

Note: The first measurement after the DTS was powered delivers a result without calibration adjustment and should be ignored therefore.

The formula to calculate the die temperature is defined in the Data Sheet.

Note: The maximum resolution is only achieved for a measurement that is part of multiple continuous measurements (recommended to ignore are two here).

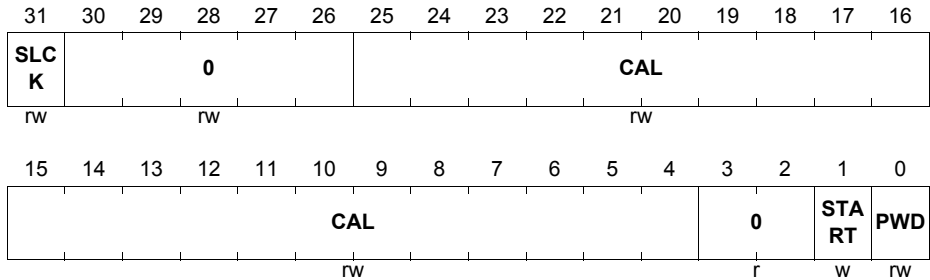
An interrupt service request (SRC_SCUDTS) is generated when DTS busy indication changes from 1 to 0.

7.4.3.1 Die Temperature Sensor Register

DTSCON

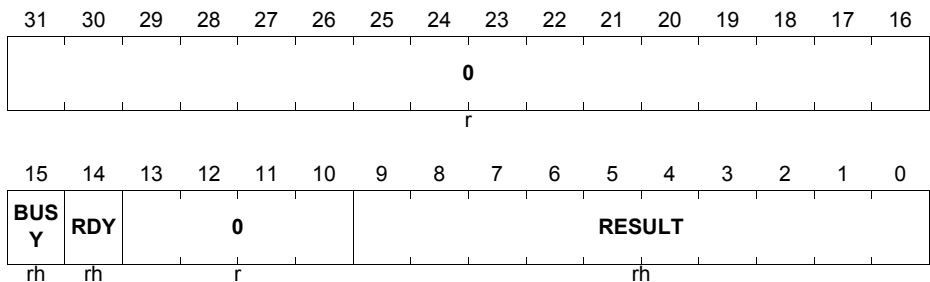
Die Temperature Sensor Control Register(0E4_H)

Reset Value: 0000 0001_H



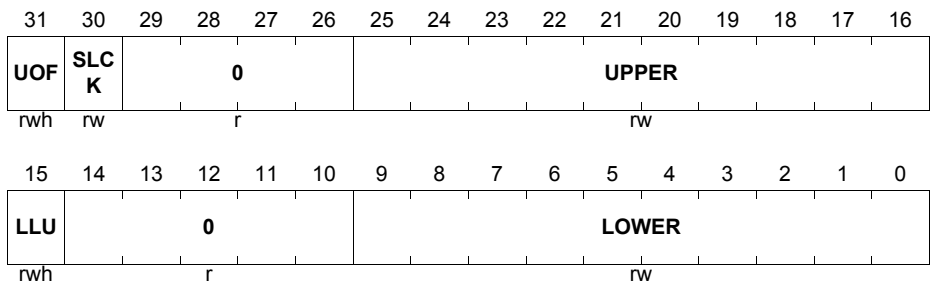
Field	Bits	Type	Description
PWD	0	rw	Sensor Power Down This bit defines the DTS power state. 0 _B The DTS is powered 1 _B The DTS is not powered
START	1	w	Sensor Measurement Start This bit starts a measurement of the DTS. 0 _B No DTS measurement is started 1 _B A DTS measurement is started If set this bit is automatically cleared. This bit always reads as zero.
CAL	[25:4]	rw	Calibration Value This bit field interfaces the calibration values to the DTS.

Field	Bits	Type	Description
SLCK	31	rw	Security Lock 0_B No lock active 1_B Lock is active If this bit is set, no bits in this register (except START) are writeable. Attempted writes when LCK is set will cause an SMU SPB error alarm trigger. This bit can not be cleared by software. This bit can only be set by an access from the HSM master ($TAG = 001101_B$). A set operation performed by any other master is ignored and the bit is not modified.
0	[3:2], [30:24]	r	Reserved Read as 0; should be written with 0.

DTSSTAT
Die Temperature Sensor Status Register(0E0_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RESULT	[9:0]	rh	Result of the DTS Measurement This bit field shows the result of the DTS measurement. The value given is directly related to the die temperature. Only the bit [9:2] have to be evaluated.
RDY	14	rh	Sensor Ready Status This bit indicate the DTS is ready or not. 0_B The DTS is not ready 1_B The DTS is ready

Field	Bits	Type	Description
BUSY	15	rh	Sensor Busy Status This bit indicate the DTS is currently busy or not. If the sensor is busy currently a measurement is running and the result should not be used. 0_B The DTS is not busy 1_B The DTS is busy <i>Note: This bit is updated 2 cycles after bit DTSCON.START is set.</i>
0	[13:10] , [31:16]	r	Reserved Read as 0; should be written with 0.

DTSLIM
Die Temperature Sensor Limit Register(240_H)
Reset Value: 03FF 0000_H


Field	Bits	Type	Description
LOWER	[9:0]	rw	Lower Limit This bit field defines the lower limit of the DTS temperature check. The DTS measurement result is compared against this value and if the measurement result is smaller than LOWER bit LLU is set.

Field	Bits	Type	Description
LLU	15	rwh	<p>Lower Limit Underflow</p> <p>0_B No temperature underflow was detected 1_B A temperature underflow was detected When this bit is set an HSM temperature underflow trigger is generated. When this bit is set an temperature SMU DTS alarm trigger is generated. This bit is cleared by writing a zero. Writing a one has no effect. This bit is updated when either a DTS measurement is finished or if bit field LOWER is changed.</p>
UPPER	[25:16]	rw	<p>Upper Limit</p> <p>This bit field defines the upper limit of the DTS temperature check. The DTS measurement result is compared against this value and if the measurement result is greater than UPPER bit UOF is set</p>
SLCK	30	rw	<p>Security Lock</p> <p>0_B No lock active 1_B Lock is active If this bit is set both bit fields LOWER and UPPER can not longer be written. Write requests when LCK is set will cause a SMU SPB error alarm trigger. This bit can not be cleared by software. This bit can only be set by an access from the HSM master (TAG = 001101_B). A set operation performed by any other master is ignored and the bit is keep as cleared.</p>
UOF	31	rh	<p>Upper Limit Overflow</p> <p>0_B No temperature overflow was detected 1_B A temperature overflow was detected When this bit is set an HSM temperature overflow trigger is generated. When this bit is set an SMU DTS alarm trigger is generated. This bit is cleared by writing a zero. Writing a one has no effect. This bit is updated when either a DTS measurement is finished or if bit field UPPER is changed.</p>

Field	Bits	Type	Description
0	[14:10] , [29:26]	r	Reserved Read as 0; should be written with 0.

7.4.4 Watchdog Timers

This section describes the TC26x B-Step Watchdog Timers (WDTs). Topics include an overview of the WDT functions and descriptions of the registers, the password-protection scheme, accessing registers, modes, and initialization.

The TC26x B-Step contains the following Watchdog Timers:

- Safety Watchdog
- CPU0 Watchdog
- CPU1 Watchdog

7.4.4.1 Watchdog Timers Overview

The WDTs provide a highly reliable and secure way to detect and recover from software or hardware failure. The WDTs help to abort an accidental malfunction of a CPU or TC26x B-Step system within a user-specified time period.

In addition to this standard “Watchdog” function, each of the WDTs incorporates an End-of-Initialization (ENDINIT) feature which can protect critical registers from accidental writes.

Servicing the Watchdogs and modifying the ENDINIT bits are critical functions that must not be allowed in case of a system malfunction. To protect these functions a sophisticated scheme is implemented that requires a password and guard bits during accesses to the WDT control registers. Any write access that does not deliver the correct password or the correct value for the guard bits is regarded as a malfunction of the system, and results in a watchdog alarm. In addition, even after a valid access has been performed and an ENDINIT bit has been cleared to provide access to the critical registers, the Watchdog imposes a time limit for this access window. If the ENDINIT bit has not been properly set again before this limit expires, the system is assumed to have malfunctioned, and a Watchdog alarm is reported to the Safety Management Unit (SMU). These stringent requirements, although not guaranteed, nonetheless provide a high degree of assurance of the robustness of system operation.

Configuration options are available which enable a Watchdog service to additionally check code execution sequence and intermediate code execution time. If these checks are enabled then any incorrect sequence or out-of-limit execution time will also result in an SMU alarm request.

Expiry of any of the TC26x B-Step WDTs leads to an SMU alarm request. It is possible to program the SMU to provide an interrupt or NMI to provide some time for recovery or status recording before further action is taken (eg reset of the device or stopping of the CPU via idle request).

Safety Watchdog

The Safety Watchdog Timer provides an overall system level watchdog which is independent from the CPU watchdogs and also provides temporal protection against

unintended writes to critical system registers which could have impact on safety-critical systems. When the Safety WDT is enabled, it can cause an SMU alarm request if it is not serviced within a user-programmable time period. A CPU must service the Safety WDT within this time interval to prevent this. The response to a Safety WDT timeout is configurable within the SMU. Hence, periodic servicing of the Safety WDT confirms that the system is functioning as expected.

Typically the SCU write protection (ACCEN) will be configured so that only restricted "Safety" CPU(s) can configure system-critical functionality. This includes the ability to service the Safety Watchdog. In addition, Safety Watchdog disable/enable/configuration function requires a Safety ENDINIT password.

The registers marked "SE" in [Table 7-28](#) and other module Register Tables are considered to be static properties of a safety-critical system and are all write-protected by the Safety ENDINIT feature so that they cannot be changed without unlocking the Safety Watchdog.

CPU Watchdogs

The individual CPU Watchdog Timers provide the ability to monitor separate CPU execution threads without the need for software to co-ordinate the shared use of a common watchdog. Each CPUx watchdog timer also incorporates a local CPUx_Endinit feature which provides temporal protection against unintended writes to critical local CPU registers and also some system registers which require protection but are not already covered by the Safety ENDINIT.

When a CPU WDT is enabled it can cause an SMU alarm request if it is not correctly serviced within a user-programmable time period. The CPU must service its CPU WDT within this time interval to prevent this. The response to each CPUx watchdog timeout is configurable within the SMU. Hence, periodic servicing of a CPU WDT confirms that the corresponding CPU is executing a software sequence as expected.

After a reset, CPU0 runs and CPU0 watchdog timer starts automatically. Other CPUs are initially in a HALT state and therefore their corresponding Watchdog Timers are disabled. The other CPU Watchdogs are not configured to generate a time-out reset by default, but this can be enabled (See SMU section). A CPU Watchdog may only be configured, enabled or disabled by its corresponding CPU.

7.4.4.2 Features of the Watchdog Timers

The main features of the WDTs are summarized here.

- 16-bit Watchdog counter
- Selectable input frequency: $f_{SPB}/64$, $f_{SPB}/256$ or $f_{SPB}/16384$
- 16-bit user-definable reload value for normal Watchdog operation, fixed reload value for Time-Out Mode
- Incorporation of the corresponding ENDINIT bit and monitoring of its modifications
- Sophisticated Password Access mechanism with user-definable password fields
- Access Error Detection: Invalid password (during first access) or invalid guard bits (during second access) trigger an alarm request to the SMU
- Temporal and logical monitoring capabilities:
 - Optional Code Sequence checking. An incorrect code sequence signature identification will trigger an alarm request to the SMU
 - Optional Code Execution Time checking. Code execution times out of expected limits will trigger an alarm request to the SMU.
- Overflow Error Detection: An overflow of the WDT counter triggers an alarm request to the SMU
- Watchdog function can be disabled; access protection and ENDINIT bit monitor function remain enabled
- Configurable mechanism to prevent Watchdog reloads after an unserviced safety warning alarm to ensure that unserviced warnings lead to an SMU hardware response
- External “Alive heartbeat” indication to show that WDT is running and SMU is not in alarm mode

7.4.4.3 The Endinit Functions

There are a number of registers in the TC26x B-Step that are usually programmed only once during the initialization sequence of the system or application. Modification of such registers during normal application run can have a severe impact on the overall operation of modules or the entire system.

While the Supervisor Mode and the Access Protection scheme provide a certain level of protection against unintentional modifications, they may not be sufficient to protect against all unintended accesses to system-critical registers.

The TC26x B-Step provides one more level of protection for such registers via the various Endinit features. This is a highly secure write-protection scheme that makes unintentional modifications of registers protected by this feature nearly impossible.

The Endinit feature consists of an ENDINIT bit incorporated in each WDT control register. Registers protected via an Endinit determine whether or not writes are enabled. Writes are only enabled if the corresponding ENDINIT = 0 AND Supervisor Mode is active. Write attempts if this condition is not true will be discarded and the register

contents will not be modified in this case. The BCU controls the further operation following a discarded write access.

To get the highest level of security, writes to the ENDINIT bits are protected by a highly secure access protection scheme implemented in the WDTs. This is a complex procedure, that makes it nearly impossible for ENDINIT bits to be modified unintentionally. In addition, each WDT monitors ENDINIT bit modifications by starting a time-out sequence each time software opens access to the critical registers through clearing the corresponding ENDINIT bit. If the time-out period ends before the corresponding ENDINIT bit is set again, a malfunction of the software is assumed and a Watchdog fault response is generated.

The access-protection scheme and the Endinit time-out operation of the WDTs is described in the following sections. In the register overview tables for each module (including the SCU itself) the registers which are protected via each Endinit type are identified in the column describing write accesses as follows:

- “CE0” - writeable only when CPU0 ENDINIT bit is zero
- “CE1” - writeable only when CPU1 ENDINIT bit is zero
- “E” - writeable when any (one or more) CPUx ENDINIT bit is zero
- “SE” - writeable only when Safety ENDINIT bit is zero
- None of the above - accessible at any time

Note: The clearing of an ENDINIT bit takes some time. Accesses to Endinit-protected registers after the clearing of an ENDINIT bit must only be done when the ENDINIT bit is really cleared. As a solution the ENDINIT bit should be read back once before Endinit-protected registers are accessed the first time after bit ENDINIT has been cleared.

Password Access to WDTxCON0

A correct password must be written to register WDTxCON0 (x=S,CPU0,CPU1) in order to unlock it for modifications. Software must either know the correct password in advance or compute it at runtime. The passwords for each of the Watchdogs (x=S,CPU0,CPU1) can be different in order to provide independent watchdog functionality program flows to have independent watchdog functions.

The Safety Watchdog password register WDTSCON0 is protected by the generic SCU protection scheme which allows only configured master(s) to have write access (See [ACCENO](#)).

CPU-specific Watchdog password registers WDTCPUyCON0 are individually protected such that they may only be written by the corresponding CPUy

A watchdog may be used within a safety application to provide a recovery time period during which software might attempt to recover from a safety alarm warning. To ensure that a CPU fault could not allow a fault to be ignored an option is provided to prevent watchdog unlocking if the Safety Management Unit (SMU) is not in the RUN state. This option may be enabled by bit WDTxCON0.UR.

If the password is valid and the SMU state meets the requirements of the WDTxCON0.US bit then WDTxCON0 will be unlocked as soon as the Password Access is completed. The unlocked condition will be indicated by WDTxCON0.LCK = 0. To ensure the correct servicing sequence, a password access is only permitted when the WDTxCON0.LCK bit was set prior to the access.

If an improper password value is written to WDTxCON0 during the Password Access, a Watchdog Access Error condition exists. Bit WDTxSR.AE is set and an alarm request is sent to the Safety Management Unit (SMU).

The 14-bit user-definable password, WDTxCON0.PW, provides additional options for adjusting the password requirements to the application's needs. It can be used, for instance, to detect unexpected software loops, or to monitor the execution sequence of routines.

[Table 7-24](#) summarizes the requirements for the password. Various options exist, which are described in more detail below

Table 7-24 Password Access Bit Pattern Requirements

Bit Position	Required Value
[1:0]	Fixed; must be written to 01 _B
[15:2]	<p>If PAS=0: WDTxCON0.PW[7:2] must be written with inverted current value read from WDTxCON0.PW[7:2] WDTxCON0.PW[15:8] must be written with non-inverted current value read from WDTxCON0.PW[15:8]</p> <p>If PAS=1: Must be written with Expected Next Sequence Password</p>
[31:16]	<p>If TCS=0: Must be written with current value of user-definable reload value, WDTxCON0.REL</p> <p>If TCS=1: Must be written with inverted estimate of the WDT count value, WDTxSR.TIM. This value must be within +/- WDTxSR.TCT of the actual value</p>

Static Password

In the static password mode (WDTxSR.PAS=0) the password can only be changed by a valid Modify Access. The Password Access has been designed such that it is not possible simply to read the register and re-write it. Some of the password read bits must be inverted (toggled) before being re-written. This prevents a simple malfunction from accidentally unlocking the WDT through a simple read/write sequence.

Automatic Password Sequencing

If automatic password sequencing is enabled (WDTxSR.PAS=1) then the password changes automatically after each password check (i.e. Password Access or Check Access). The Expected Next Password follows a pseudo-random sequence based upon a 14-bit Fibonacci LFSR (Linear Feedback Shift Register) with characteristic polynomial $x^{14}+x^{13}+x^{12}+x^2+1$. An initial password (or subsequent manual password updates) can also be provided by Modify Accesses.

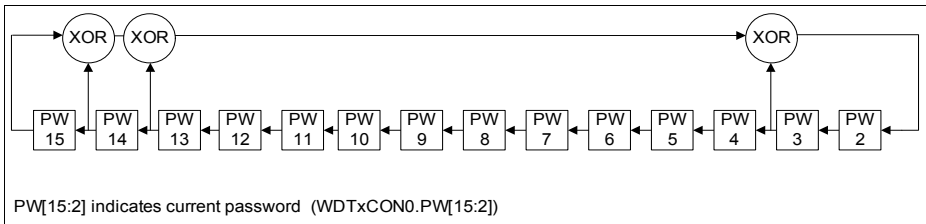


Figure 7-46 Password Sequencing LFSR

Time-Independent Password

If time checking is not enabled (WDTxSR.TCS=0) then the REL field of the WDTxCON0 register must be simply re-written with the existing reload value during the Password Access.

Time Check Password

If time checking is enabled (WDTxSR.TCS=1) the the REL field of the WDTxCON0 register must be written with an inverted (bitflipped) estimate of the current WDT count value. The acceptable margin of error of this estimate (in WDT clock periods) is specified by the value of WDTxSR.TCT. If the written estimate is outside the range WDTxSR.TIM +/- WDTxSR.TCT, then an SMU alarm condition is indicated. This mechanism can provide a check of the elapsed program execution time since the last WDT restart. Note that a Time Check comparison is still required for a Password or Check Access while the WDT is operating in Time-Out mode (e.g. After accessing ENDINIT-protected registers).

Check Access to WDTxCON0

A Check Access is identical to a Password Access except that the lock bit is not cleared and a subsequent Modify Access is therefore not allowed. A Check Access does not trigger an SMU Alarm Request provided that the write data requirements are satisfied. A Check Access may only be performed when the LCK bit is set.

This type of access is used for intermediate checkpoints between WDT services. This may be used (e.g. in conjunction with the timestamp count checking feature or sequence passwords) for task sequence or execution time monitoring.

Table 7-25 Check Access Bit Pattern Requirements

Bit Position	Required Value
[1:0]	Fixed; must be written to 11 _B
[15:2]	<p>If PAS=0: WDTxCON0.PW[7:2] must be written with inverted current value read from WDTxCON0.PW[7:2] WDTxCON0.PW[15:8] must be written with non-inverted current value read from WDTxCON0.PW[15:8]</p> <p>If PAS=1: Must be written with Expected Next Sequence Password</p>
[31:16]	<p>If TCS=0: Must be written with current value of user-definable reload value, WDTxCON0.REL</p> <p>If TCS=1: Must be written with inverted estimate of the WDT counter WDTxSR.TIM. This value must be within +/- WDTxSR.TCT of the actual value</p>

If an improper value is written to WDTxCON0 during the Check Access, a Watchdog Access Error condition exists. Bit WDTxSR.AE is set and an alarm request is sent to the Safety Management Unit (SMU).

Modify Access to WDTxCON0

If a WDTxCON0 is successfully unlocked by a Password Access, the following write access to WDTxCON0 can modify it. However, this access must also meet certain requirements in order to be accepted and regarded as valid. [Table 7-26](#) lists the required bit patterns. If the access does not follow these rules, a Watchdog Access Error condition is detected, bit WDTxSR.AE is set, and an alarm request is sent to the Safety Management Unit (SMU).

Table 7-26 Modify Access Bit Pattern Requirements

Bit Position	Value
0	User-definable; desired value for bit WDTxCON0.ENDINIT.
1	Fixed; must be written with 1 _B .
[15:2]	User-definable; desired value of user-definable password field, WDTxCON0.PW.
[31:16]	User-definable; desired value of user-definable reload value, WDTxCON0.REL.

After the Modify Access has completed, WDTxCON0.LCK is set again, automatically re-locking WDTxCON0. Before the register can be modified again, a valid Password Access must be executed again.

External WDT “Alive Heartbeat” Indication

The current state of the SAFECON.HBT bit may optionally be routed to external pin(s). This register may only be written during a Safety WDT service (between a Modify Access with Safety ENDINIT=0 and the subsequent Modify Access with Safety ENDINIT=1). Periodic toggling of this bit by software can therefore be used to indicate to an external monitor that the CPU is operational and the Safety Watchdog Timer is still running. If the Unlock Restriction (UR bit) is also enabled then any other SMU safety alarm condition will prevent a WDT refresh and hence stop the heartbeat automatically.

Access to Endinit-Protected Registers

If write access to Endinit-protected registers is required during run time, write access can be re-enabled for a limited time period. To re-enable access, WDTxCON0 must first be unlocked with a valid Password Access. In the subsequent valid Modify Access, ENDINIT can be cleared. Access to Endinit-protected registers is now open again. However, when WDTxCON0 is unlocked, the WDT is automatically switched to Time-Out Mode. The access window is therefore time-limited. Time-Out Mode is only terminated after ENDINIT has been set again, requiring another Valid Password and Valid Modify Access to WDTxCON0.

If the WDT is not used in an application and is therefore disabled (WDTxSR.DS = 1), the above described case is the only occasion when WDTxCON0 must be accessed again after the system is initialized. If there are no further changes to critical system registers needed, no further accesses to WDTxCON0, WDTxCON1, or WDTxSR are necessary. However, it is recommended that the WDT be used in an application for safety reasons.

For debugging support the Cerberus module can override all the ENDINIT controls of all WDTs to ease the debug flow. If bit CBS_OSTATE.ENIDIS is set all ENDINIT protection is disabled independent of the current status configured by the WDTs. If CBS_OSTATE.ENIDIS is cleared the complete control is within the WDTs.

7.4.4.4 Timer Operation

The timers for Safety Watchdog and CPU0 are automatically active after an Application Reset. Each 16-bit counter implementing the timer functionality is either triggered with $f_{SPB} / 64$, $f_{SPB} / 256$ or $f_{SPB} / 16384$. The three possible counting rates are controlled via bit WDTxCON1.IRx.

Determining WDT Periods

All WDTs use the SPB clock f_{SPB} . A clock divider in front of each WDT provides three WDT counter frequencies, $f_{SPB} / 64$, $f_{SPB} / 256$ and $f_{SPB} / 16384$.

The general formula to calculate a Watchdog timeout period is:

(7.26)

$$\text{period} = \frac{(2^{16} - \text{startvalue}) \cdot \text{divider}}{f_{SPB}}$$

The parameter startvalue represents the fixed value $FFFC_H$ for the calculation of the Time-Out Period, and the user-programmable reload value WDTxCON0.REL for the calculation of the Normal Period.

The parameter divider represents the user-programmable source clock division selected by WDTxCON0.IRx, which can be 64, 256 or 16384.

Timer Modes

Each Watchdog Timer can operate in one of three different operating modes:

- Time-Out Mode
- Normal Mode
- Disable Mode

The following overview describes these modes and how the WDT changes from one mode to the other.

Time-Out Mode

The Time-Out Mode is entered after an Application Reset (Except CPU1 WDTs which are disabled after Application Reset) or when a valid Password Access to register WDTxCON0 is performed. The Time-Out Mode is indicated by bit WDTxSR.TO = 1. The timer is set to $FFFC_H$ and starts counting upwards. Time-Out Mode can only be exited properly by setting ENDINIT = 1 with a correct access sequence. If an improper access to the WDT is performed, or if the timer overflows before ENDINIT is set, and an alarm request is sent to the Safety Management Unit (SMU).

A proper exit from Time-Out Mode can either be to the Normal or the Disable Mode, depending on the state of the disable request bit WDTxCON1.DR.

Normal Mode

In Normal Mode (DR = 0), the WDT operates in a standard Watchdog fashion. The timer is set to WDTxCON0.REL, and begins counting up. It has to be serviced before the counter overflows. Servicing is performed through a proper access sequence to the control register WDTxCON0. This enters the Time-Out Mode.

If the WDT is not serviced before the timer overflows, a system malfunction is assumed. Normal Mode is terminated and an alarm request is sent to the Safety Management Unit (SMU)

Disabled Mode

Disabled Mode is provided for applications which truly do not require the WDT function. It can be requested from Time-Out Mode when the disable request bit WDTxCON1.DR is set. The Disabled Mode is entered when was requested AND bit WDTxCON0.ENDINIT is set. The timer is stopped in this mode. However, disabling the WDT only stops it from performing the standard Watchdog function, eliminating the need for timely service of the WDT. It does not disable Time-Out mode. If an access to register WDTxCON0 is performed in Disabled Mode, Time-Out Mode is entered if the access was valid, and an alarm request is sent to the Safety Management Unit (SMU) if the access was invalid. Thus, the ENDINIT monitor function as well as (a part of) the system malfunction detection will still be active.

WDT Alarm Request

SMU alarm requests are generated for three cases:

- Invalid write data during access to register WDTxCON0
- Not executing a password access before a timer overflow occurs in the Time-Out Mode
- Not serving the WDT before a timer overflow occurs in the Normal Mode

The resulting alarm response(eg NMI, Reset etc) is programmable within the Safety Management Unit (SMU).

Note: The WDT itself is reset by any Application Reset .

Servicing the Watchdog Timer

If the WDT is used in an application and is enabled ($WDTxSR.DS = 0$), it must be regularly serviced to prevent it from overflowing.

Service is performed in two steps, a valid Password Access followed by a valid Modify Access. The valid Password Access to $WDTxCON0$ automatically switches the WDT to Time-Out Mode. Thus, the Modify Access must be performed before the time-out expires or an alarm request will be sent to the Safety Management Unit (SMU).

During the next Modify Access, the strict requirement is that $WDTxCON0.ENDINIT$ is written with 1.

Note: $ENDINIT$ must be written with 1 to perform a proper service, even if it is already set to 1.

Changes to the reload value $WDTxCON0.REL$, or the user-definable password $WDTxCON0.PW$, are not required.

When a WDT service is properly executed, Time-Out Mode is terminated, and the WDT switches back to its former mode of operation, and the WDT service is complete.

Check Accesses are optional and may be performed at any time when the WDT is locked.

WDT Operation During Power-Saving Modes

If one or more of the CPU are in Idle Mode, they cannot service a WDT because no software is running. Excluding the case where the system is running normally, a strategy for managing WDTs is needed while a CPU is in Idle Mode. There are two ways to manage a WDT in these cases. Firstly, the Watchdog can be disabled before idling the CPU. The disadvantage of this is that the system will no longer be monitored during the idle period.

A better approach to this problem relies upon the wake-up feature of the WDTs. Whenever $CPUx$ is put in Idle or Sleep Mode and the WDT is not disabled, the $CPUx$ is woken at regular intervals. When $WDTx$ changes its count value ($WDTxSR.TIM$) from $7FFF_H$ to 8000_H , $CPUx$ is woken and continues execution at the instruction following the instruction that was executed before entering the Idle or Sleep Mode.

Note: Before switching into a non-running power-management mode, software should perform a Watchdog service sequence. At the Modify Access, the Watchdog reload value, $WDTxCON0.REL$, should be programmed such that the wake-up occurs after a period which best meets application requirements. The maximum period between two CPU wake-ups is one-half of the maximum WDT period.

Suspend Mode Support

During a debug session the Watchdog functionality might lead to unintended alarms.

By default the WDTs are disabled when OCDS is enabled. However, it is possible to enable the WDTs also when debug is enabled. Please refer to the manual of your debugger tool for details.

7.4.4.5 Watchdog Timer Registers

Watchdog Timer Control Register 0

This register is written with check data in Check Accesses and Password Accesses. It also stores the timer reload value, password update and the corresponding End-of-Initialization (ENDINIT) control bit during a Modify Access.

There is a WDTxCON0 register for each Watchdog x.

WDTSCON0

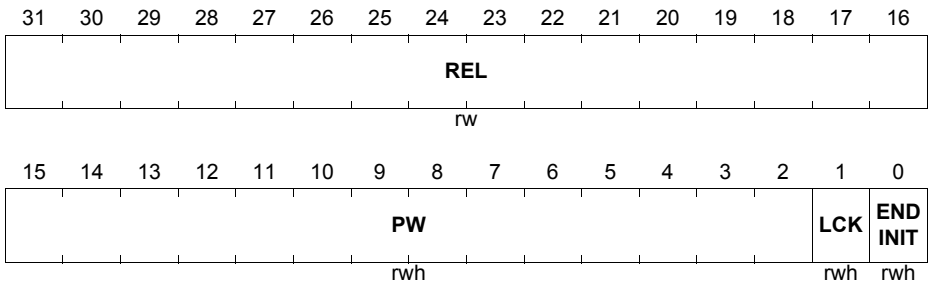
Safety WDT Control Register 0 (0F0_H) **Reset Value: FFFC 000E_H**

WDTCPU0CON0

CPU0 WDT Control Register 0 (100_H) **Reset Value: FFFC 000E_H**

WDTCPU1CON0

CPU1 WDT Control Register 0 (10C_H) **Reset Value: FFFC 000F_H**



Field	Bits	Type	Description
ENDINIT	0	rwh	<p>End-of-Initialization Control Bit</p> <p>0_B Access to Endinit-protected registers is permitted.</p> <p>1_B Access to Endinit-protected registers is not permitted.</p> <p>This bit must be written with a '1' during a Password Access or Check Access (although this write is only used for the password-protection mechanism and is not stored). This bit must be written with the required ENDINIT update value during a Modify Access.</p>

Field	Bits	Type	Description
LCK	1	rwh	<p>Lock Bit to Control Access to WDTxCON0</p> <p>0_B Register WDTxCON0 is unlocked 1_B Register WDTxCON0 is locked (default after ApplicationReset)</p> <p>The current value of LCK is controlled by hardware. It is cleared after a valid Password Access to WDTxCON0 when WDTxSR.US is 0 (or when WDTxSR.US is 1 and the SMU is in RUN mode), and it is automatically set again after a valid Modify Access to WDTxCON0. During a write to WDTxCON0, the value written to this bit is only used for the password-protection mechanism and is not stored.</p> <p>This bit must be cleared during a Password Access to WDTxCON0, and set during a Modify Access to WDTxCON0.</p> <p>A Check Access does not clear LCK.</p>
PW	[15:2]	rwh	<p>User-Definable Password Field for Access to WDTxCON0</p> <p>This bit field is written with an initial password value during a Modify Access.</p> <p>A read from this bitfield returns this initial password, but bits [7:2] are inverted (toggled) to ensure that a simple read/write is not sufficient to service the WDT.</p> <p>If corresponding WDTxSR.PAS = 0 then this bit field must be written with its current contents during a Password Access or Check Access.</p> <p>If corresponding WDTxSR.PAS = 1 then this bit field must be written with the next password in the LFSR sequence during a Password Access or Check Access</p> <p>The default password after Application Reset is 00000000111100_B</p>

Field	Bits	Type	Description
REL	[31:16]	rw	<p>Reload Value for the WDT (also Time Check Value)</p> <p>The reload value can be changed during a Modify Access to WDTxCON0 (Default after ApplicationReset is FFFC_H). If the Watchdog Timer is enabled and in Normal Timer Mode, it will start counting from this value after a correct Watchdog service.</p> <p>A read from this bitfield always returns the current reload value.</p> <p>During a Password Access or a Check Access this bitfield may be used for additional checks. Writes during such checks have no effect upon the reload value.</p> <p>If corresponding WDTxSR.TCS=0 then this bit field must be written with its current contents during a Password Access or Check Access.</p> <p>If corresponding WDTxSR.TCS=1 then this bit field must be written with an inverted estimate of the current WDTxSR.TIM value during a Password Access or Check Access.</p>

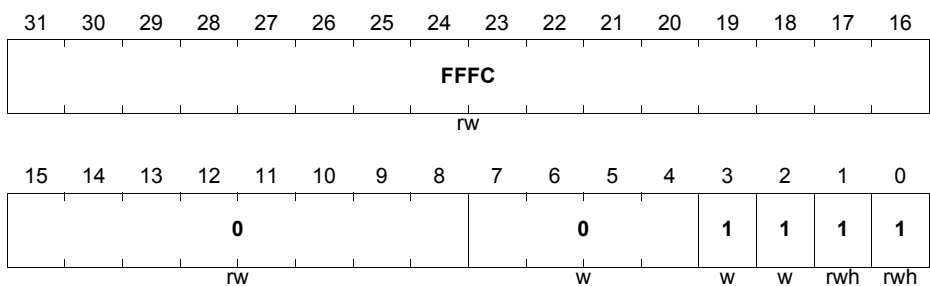
The following register is implemented but is not functional in this product.

WDTCPU2CON0

Reserved Register

(118_H)

Reset Value: FFFC 000F_H



Field	Bits	Type	Description
FFFC	[31:16]	rw	Reserved Not functional in this device.
0	[15:8]	rw	Reserved Not functional in this device.
0	[7:4]	w	Reserved Not functional in this device.
1	[3:2]	w	Reserved Not functional in this device.
1	[1:0]	rw	Reserved Not functional in this device.

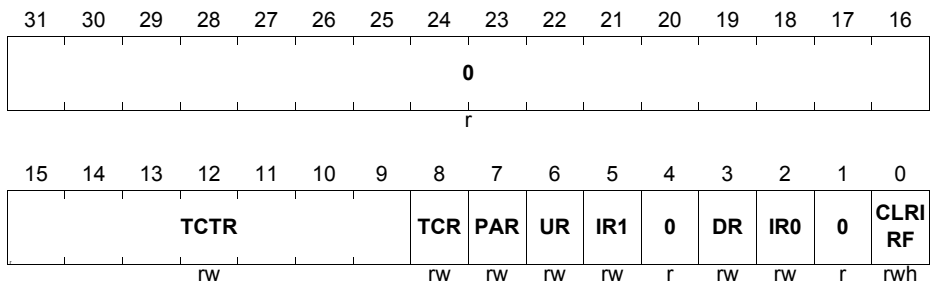
Watchdog Timer Control Register 1

The register WDTxCON1 for each Watchdog manages operation of the WDT. It includes the disable request, password configuration and frequency selection bits. Each WDTxCON1 register is protected by the corresponding ENDINIT which it controls.

WDTSCON1 has an additional bit CLRIRF which can be used to clear the internal reset status used to detect double SMU (eg WDT) resets.

WDTSCON1

Safety WDT Control Register 1 **(0F4_H)** **Reset Value: 0000 0000_H**

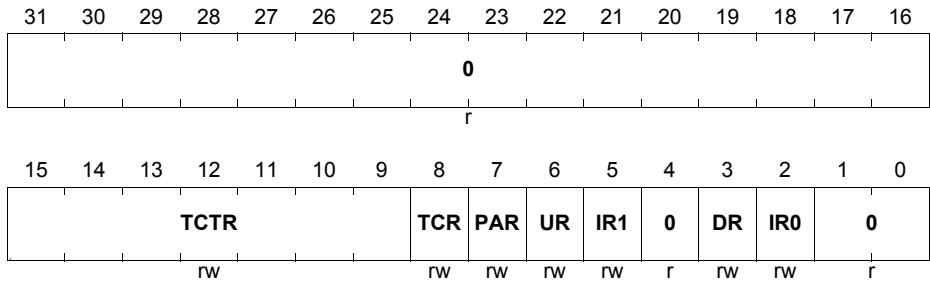


Field	Bits	Type	Description
CLRIRF	0	rw	<p>Clear Internal Reset Flag</p> <p>This bit is used to request a clear of the internal flag which indicates whether a previous SMU reset has already been requested</p> <p>0_B No action 1_B Request to clear the internal previous-SMU-reset flag</p> <p>This bit can only be modified if WDTSCON0.ENDINIT is cleared. The internal flag is cleared when ENDINIT is set again. As long as ENDINIT is cleared, the internal flag is unchanged and continues to determine the response to a further SMU reset request. When ENDINIT is set again with a valid Modify Access, the internal flag is cleared together with this bit.</p>
IR1, IR0	5,2	rw	<p>Input Frequency Request Control</p> <p>00_B Request to set input frequency to $f_{SPB}/16384$. 01_B Request to set input frequency to $f_{SPB}/256$. 10_B Request to set input frequency to $f_{SPB}/64$. 11_B Reserved. Do not use.</p> <p>Bit IR0 and IR1 should be programmed together to determine the WDT timer frequency.</p> <p>These bits can only be modified if the corresponding WDTSCON0.ENDINIT is cleared. WDTSSR.ISx are updated by these bit only when ENDINIT is set again. As long as ENDINIT is cleared, WDTSSR.ISx controls the current input frequency of the Watchdog Timer. When ENDINIT is set again, WDTSSR.ISx is updated with the values of IRx.</p>
DR	3	rw	<p>Disable Request Control Bit</p> <p>0_B Request to enable the WDT 1_B Request to disable the WDT</p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTSSR.DS is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTSSR.DS controls the current enable/disable status of the WDT. When the ENDINIT is set again with a Valid Modify Access, WDTSSR.DS is updated with the state of DR.</p>

Field	Bits	Type	Description
UR	6	rw	<p>Unlock Restriction Request Control Bit</p> <p>0_B Request to disable SMU restriction of WDT unlock</p> <p>1_B Request to enable SMU restriction of WDT unlock</p> <p>This bit can only be modified if the corresponding WDTSCON0.ENDINIT is cleared. WDTxSR.US is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTSSR.US controls whether unlocking is possible at all times or only when the SMU is in the RUN state. When the ENDINIT is set again with a Valid Modify Access, WDTSSR.US is updated with the state of UR.</p>
PAR	7	rw	<p>Password Auto-sequence Request Bit</p> <p>0_B Request no automatic change of password after each Modify Access or Check Access</p> <p>1_B Request automatic sequence of password after each Modify Access or Check Access</p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.PAS is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.PAS controls password sequencing. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.PAS is updated with the state of PAR.</p>

Field	Bits	Type	Description
TCR	8	rw	<p>Counter Check Request Bit</p> <p>0_B Request to check only that REL field is written with existing REL value during Modify Access or Check Access</p> <p>1_B Request to check that REL field is written with correct TIM Count (within tolerance of WDTxSR.TCT) during Modify Access or Check Access</p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.TCS is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.TCS controls whether counter check is enabled. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.TCS is updated with the state of TCR</p>
TCTR	[15:9]	rw	<p>Timer Check Tolerance Request</p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.TCT is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.TCT controls the tolerance of timer checks. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.TCT is updated with the state of TCTR</p>
0	1,4, [31:16]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

The register WDTxCON1 for each CPU Watchdog manages operation of the WDT. It includes the disable request, password configuration and frequency selection bits. Each WDTxCON1 register is protected by the corresponding ENDINIT which it controls.

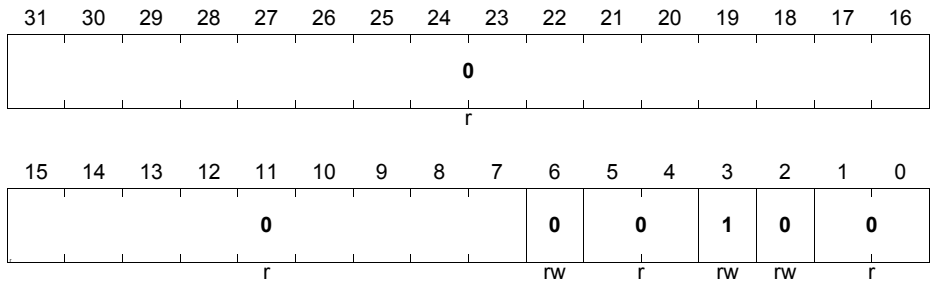
WDTCPU0CON1
CPU0 WDT Control Register 1 (104_H) Reset Value: 0000 0000_H
WDTCPU1CON1
CPU1 WDT Control Register 1 (110_H) Reset Value: 0000 0008_H


Field	Bits	Type	Description
IR1, IR0	5,2	rw	<p>Input Frequency Request Control</p> <p>00_B Request to set input frequency to $f_{SPB}/16384$.</p> <p>01_B Request to set input frequency to $f_{SPB}/256$.</p> <p>10_B Request to set input frequency to $f_{SPB}/64$.</p> <p>11_B Reserved. Do not use.</p> <p>Bit IR0 and IR1 should be programmed together to determine the WDT timer frequency.</p> <p>These bits can only be modified if the corresponding WDTSCON0.ENDINIT is cleared. WDTSSR.ISx are updated by these bit only when ENDINIT is set again. As long as ENDINIT is cleared, WDTSSR.ISx controls the current input frequency of the Watchdog Timer.</p> <p>When ENDINIT is set again, WDTSSR.ISx is updated with the values of IRx.</p>

Field	Bits	Type	Description
DR	3	rw	<p>Disable Request Control Bit</p> <p>0_B Request to enable the WDT</p> <p>1_B Request to disable the WDT</p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.DS is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.DS controls the current enable/disable status of the WDT. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.DS is updated with the state of DR.</p>
UR	6	rw	<p>Unlock Restriction Request Control Bit</p> <p>0_B Request to disable SMU restriction of WDT unlock</p> <p>1_B Request to enable SMU restriction of WDT unlock</p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.US is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.US controls whether unlocking is possible at all times or only when the SMU is in the RUN state. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.US is updated with the state of UR.</p>
PAR	7	rw	<p>Password Auto-sequence Request Bit</p> <p>0_B Request no automatic change of password after each Modify Access or Check Access</p> <p>1_B Request automatic sequence of password after each Modify Access or Check Access</p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.PAS is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.PAS controls password sequencing. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.PAS is updated with the state of PAR.</p>

Field	Bits	Type	Description
TCR	8	rw	<p>Counter Check Request Bit</p> <p>0_B Request to check only that REL field is written with existing REL value during Modify Access or Check Access</p> <p>1_B Request to check that REL field is written with correct TIM Count (within tolerance of WDTxSR.TCT) during Modify Access or Check Access</p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.TCS is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.TCS controls whether counter check is enabled. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.TCS is updated with the state of TCR</p>
TCTR	[15:9]	rw	<p>Timer Check Tolerance Request</p> <p>This bit can only be modified if the corresponding WDTxCON0.ENDINIT is cleared. WDTxSR.TCT is updated when ENDINIT is set again. As long as the ENDINIT is cleared, bit WDTxSR.TCT controls the tolerance of timer checks. When the ENDINIT is set again with a Valid Modify Access, WDTxSR.TCT is updated with the state of TCTR</p>
0	0,1,4, [31:16]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

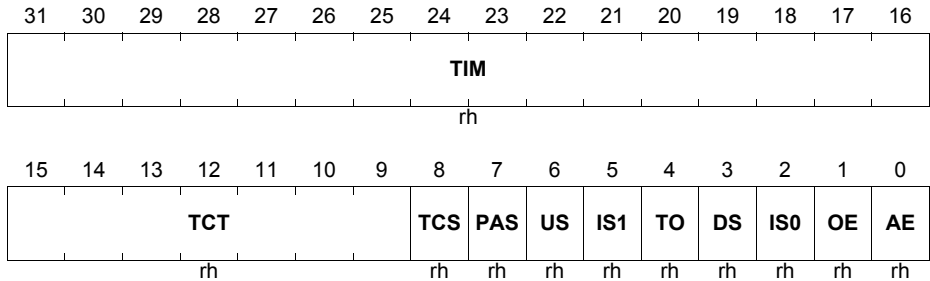
The following registers are implemented for reuse but are not functional in the TC26x B-Step.

WDTCPU2CON1
Reserved Register
(11C_H)
Reset Value: 0000 0008_H


Field	Bits	Type	Description
0	[31:7], 5,4,1,0	r	Reserved Not functional in this device.
0	6,2	rw	Reserved Not functional in this device.
1	3	rw	Reserved Not functional in this device.

Watchdog Timer Status Register

The WDTxSR registers show the current state of each WDT. Status include bits indicating Time-Out, enable/disable status, input clock status, and access error status.

WDTSSR
Safety WDT Status Register (0F8_H) Reset Value: FFFC 0010_H
WDTCPU0SR
CPU0 WDT Status Register (108_H) Reset Value: FFFC 0010_H
WDTCPU1SR
CPU1 WDT Status Register (114_H) Reset Value: FFFC 0008_H


Field	Bits	Type	Description
AE	0	rh	Watchdog Access Error Status Flag 0 _B No Watchdog access error 1 _B A Watchdog access error has occurred This bit is set when an illegal Password Access or Modify Access to register WDTxCON0 was attempted. This bit is only cleared when WDTxCON0.ENDINIT is set during a valid Modify Access
OE	1	rh	Watchdog Overflow Error Status Flag 0 _B No Watchdog overflow error 1 _B A Watchdog overflow error has occurred This bit is set when the WDT overflows from FFFF _H to 0000 _H . This bit is only cleared when WDTxCON0.ENDINIT is set to 1 during a valid Modify Access.

Field	Bits	Type	Description
IS1, IS0	5,2	rh	<p>Watchdog Input Clock Status</p> <p>00_B WDT counter frequency is $f_{SPB}/16384$.</p> <p>01_B WDT counter frequency is $f_{SPB}/256$.</p> <p>10_B WDT counter frequency is $f_{SPB}/64$.</p> <p>11_B Reserved. Do not use.</p> <p>Bit IS0 and IS1 should be programmed together. These bits are updated with the state of bits WDTxCON1.IRx after WDTxCON0.ENDINIT is written with 1 during a valid Modify Access to register WDTxCON0.</p>
DS	3	rh	<p>Watchdog Enable/Disable Status Flag</p> <p>0_B WDT is enabled (default after ApplicationReset)</p> <p>1_B WDT is disabled</p> <p>This bit is updated with the state of bit WDTxCON1.DR (after WDTxCON0.ENDINIT is set during a Valid Modify Access to register WDTxCON0) and it is cleared when Time-Out mode is entered.</p>
TO	4	rh	<p>Watchdog Time-Out Mode Flag</p> <p>0_B The Watchdog is not operating in Time-Out Mode</p> <p>1_B The Watchdog is operating in Time-Out Mode (default after ApplicationReset)</p> <p>This bit is set when Time-Out Mode is entered. It is automatically cleared when Time-Out Mode is left.</p>
US	6	rh	<p>SMU Unlock Restriction Status Flag</p> <p>0_B WDT unlock permitted at any time</p> <p>1_B WDT unlock only permitted when the SMU is in the RUN state.</p> <p>WDTxCON0.LCK will not be unlocked by a valid Password Access if this bit is '1' and the SMU is not in the RUN state</p>

Field	Bits	Type	Description
PAS	7	rh	<p>Password Auto-sequence Status Flag</p> <p>0_B No change of password after each Modify Access or Check Access</p> <p>1_B Automatically sequence the password after each Modify Access or Check Access</p> <p>This bit is updated with the state of bit WDTxCON1.PAR after WDTxCON0.ENDINIT is written with 1 during a valid Modify Access to register WDTxCON0.</p>
TCS	8	rh	<p>Timer Check Status Flag</p> <p>0_B Check only that REL field is written with existing REL value during Modify Access or Check Access</p> <p>1_B Check that REL field is written with inverted estimated TIM value (within +/- TCT value) during Password Access or Check Access</p> <p>This bit is updated with the state of bit WDTxCON1.TCR after WDTxCON0.ENDINIT is written with 1 during a Valid Modify Access to register WDTxCON0.</p>
TCT	[15:9]	rh	<p>Timer Check Tolerance</p> <p>This field determines the tolerance of the timer check during Password or Check Access (See TCS). This bit is updated with the state of bit WDTxCON1.TCTR after WDTxCON0.ENDINIT is written with 1 during a Valid Modify Access to register WDTxCON0.</p>
TIM	[31:16]	rh	<p>Timer Value</p> <p>Reflects the current content of the WDT.</p>

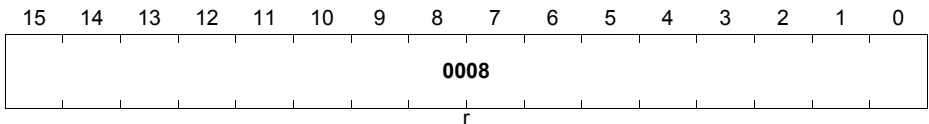
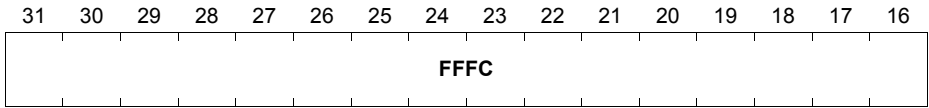
The following registers are implemented for reuse but are not functional in the TC26x B-Step.

WDTCPU2SR

CPU2 WDT Status Register

(120_H)

Reset Value: FFFC 0008_H



Field	Bits	Type	Description
FFFC	[31:16]	r	Reserved Not functional in this device
0008	[15:0]	r	Reserved Not functional in this device

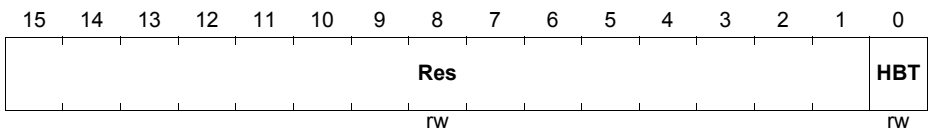
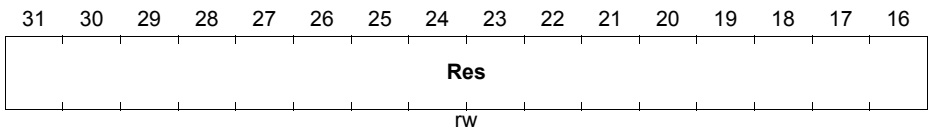
Safety Heartbeat Register

SAFECON

Safety Heartbeat Register

(150_H)

Reset Value: XXXX XXXX_H



Field	Bits	Type	Description
HBT	0	rw	<p>Heartbeat</p> <p>This bit can be used by software to provide a “heartbeat” indication to an external safety monitoring device.</p> <p>It directly controls the output on the heartbeat ports. WDTxLCK P20.9, P20.8, P20.7 and/or P20.6 (Subject to port configuration).</p> <p>0_B Port drives low (reset value) 1_B Port drives high</p> <p>Writes are only possible when Safety ENDINIT=0</p>
Res	[31:1]	rw	<p>Reserved</p> <p>Content must not be changed. Write back only the original read value.</p>

7.4.5 Emergency Stop Output Control

The emergency stop feature of the TC26x B-Step provides a fast reaction to an emergency event without the intervention of software. In response to the emergency event, selected output ports can be immediately placed into a defined state (for more information see the port chapter).

An emergency stop may be triggered by either of the following emergency events:

- A transition on the port which is configured as the Emergency Stop input
- An SMU alarm event or SMU command which is enabled and configured to generate a port emergency stop (PES). See SMU Chapter for details.

Figure 7-47 shows a diagram of the emergency stop input logic. This logic is controlled by the SCU Emergency Stop Register EMSR.

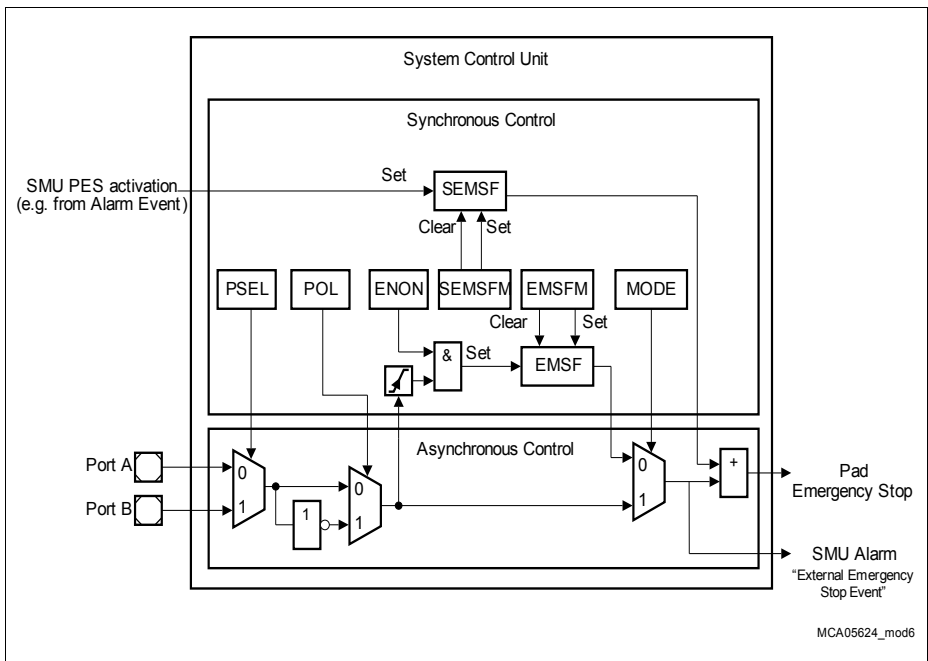


Figure 7-47 Emergency Stop Control

7.4.5.1 Port Triggered Emergency Stop

This can be configured to trigger on either transition edge of either one of two ports.

The two input port options for TC26x B-Step are (A) P 33.8 and (B) P21.2.

The emergency stop control logic for the ports can basically operate in two modes:

- Synchronous Mode (default after reset):
Emergency case is activated by hardware and released by software.
- Asynchronous Mode:
Emergency case is activated and released by hardware.

In Synchronous Mode (selected by `EMSR.MODE = 0`), the port signal is sampled for a inactive-to-active level transition, and an emergency stop flag `EMSR.EMSFS` is set if the transition is detected. The setting of `EMSR.EMSFS` activates the emergency stop. A port triggered emergency state can only be terminated by clearing `EMSR.EMSFS` via software (Write `EMSR.EMSFSM` with 10_{B}). The synchronous control logic is clocked by the system bus clock f_{SPB} . This results in a small delay between the port signal and emergency stop signal generation.

In Asynchronous Mode (selected by `EMSR.MODE = 1`), the occurrence of an active level at the port input immediately activates the emergency stop signal. Of course, a valid-to-invalid transition of the port input (emergency case is released) also immediately deactivates the emergency stop signal.

The `EMSR.POL` bit determines the active level of the input signal from the port. The `EMSR.MODE` bit selects Synchronous or Asynchronous Mode for emergency stop signal generation and the `EMSR.PSEL` bit selects which of the two ports is used as the emergency stop trigger.

7.4.5.2 SMU Event Triggered Emergency Stop

The Safety Alarm(s) which can trigger an Emergency Stop are configured and enabled within the Safety Management Unit (SMU). All SMU triggered Emergency Stop cases are in Synchronous Mode, regardless of the state of `EMSR.MODE`. The safety emergency stop flag `EMSR.SEMSF` is set when a configured and enabled SMU Safety Alarm occurs. The setting of `EMSR.SEMSF` activates the emergency stop. An SMU triggered emergency state can only be terminated by clearing the `EMSR.SEMSF` via software (Write `EMSR.SEMSFM` with 10_{B}). The synchronous control logic is clocked by the system bus clock f_{SPB} .

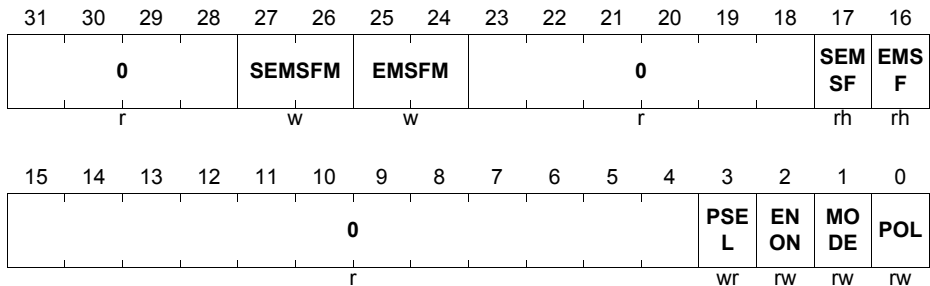
7.4.5.3 Emergency Stop Register

The Emergency Stop Register EMSR contains control and status bits/flags of the emergency stop input logic.

EMSR

Emergency Stop Register

 (0FC_H)

 Reset Value: 0000 0001_H


Field	Bits	Type	Description
POL	0	rw	Input Polarity This bit determines the polarity of the configured Emergency Stop input. 0 _B Input is active high 1 _B Input is active low
MODE	1	rw	Mode Selection This bit determines the operating mode of the emergency stop signal. 0 _B Synchronous Mode selected; emergency stop is derived from the state of flag EMSF 1 _B Asynchronous Mode selected; emergency stop is directly derived from the state of the input signal
ENON	2	rw	Enable ON This bit enables the setting of flag EMSF by an inactive-to-active level transition of input signal. 0 _B Setting of EMSF is disabled 1 _B Setting of EMSF is enabled

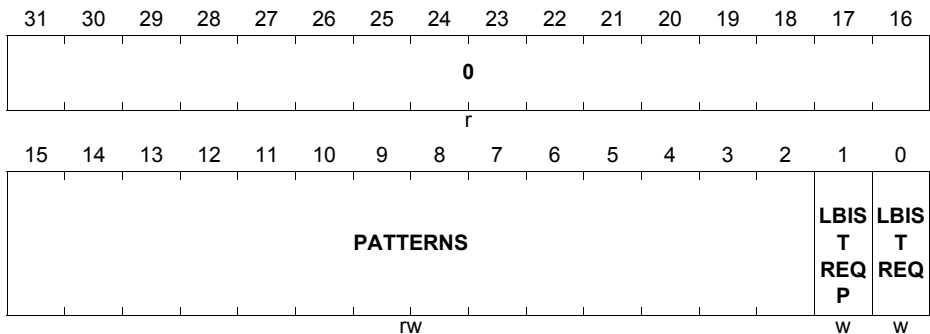
Field	Bits	Type	Description
PSEL	3	rw	PORT Select This bit selects which one of the two Emergency Stop port options is monitored. 0 _B Port A is used as Emergency Stop input 1 _B Port B is used as Emergency Stop input
0	[15:4]	r	Reserved Read as 0; should be written with 0.
EMSF	16	rh	Emergency Stop Flag This bit indicates that a synchronous mode port-triggered emergency stop condition has occurred. 0 _B An emergency stop has not occurred 1 _B An emergency stop has occurred and emergency stop state becomes active (if MODE = 0)
SEMSF	17	rh	SMU Emergency Stop Flag This bit indicates that an SMU Safety Alarm triggered emergency stop condition has occurred. 0 _B An emergency stop has not occurred 1 _B An emergency stop has occurred and emergency stop state becomes active
0	[23:18]	r	Reserved Read as 0; should be written with 0.
EMSMF	[25:24]	w	Emergency Stop Flag Modification This bit field sets or clears flag EMSF via software. 00 _B EMSF remains unchanged 01 _B EMSF becomes set 10 _B EMSF becomes cleared 11 _B EMSF remains unchanged EMSMF is always read as 00 _B .
SEMSFM	[27:26]	w	SMU Emergency Stop Flag Modification This bit field sets or clears flag SEMSF via software. 00 _B SEMSF remains unchanged 01 _B SEMSF becomes set 10 _B SEMSF becomes cleared 11 _B SEMSF remains unchanged SEMSFM is always read as 00 _B .
0	[31:28]	r	Reserved Read as 0; should be written with 0.

7.4.6 LBIST Support

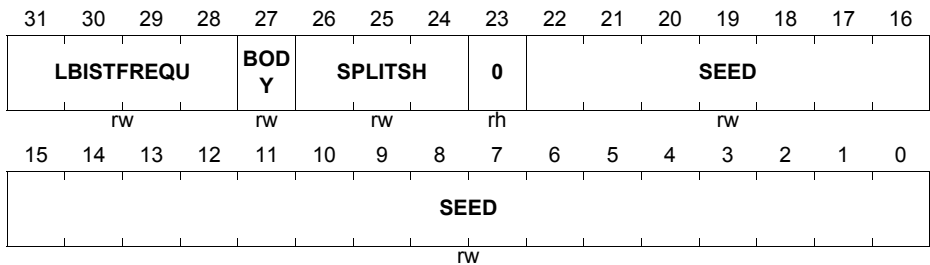
For TC26x B-Step an automatic executed scan-test mechanism is implemented, allowing the structural verification of the silicon in an application system. This process is controlled and monitored by the LBIST registers.

7.4.6.1 LBIST Control Register

The LBISTCTRL Control Register provides the link between software and the LBIST-controller.

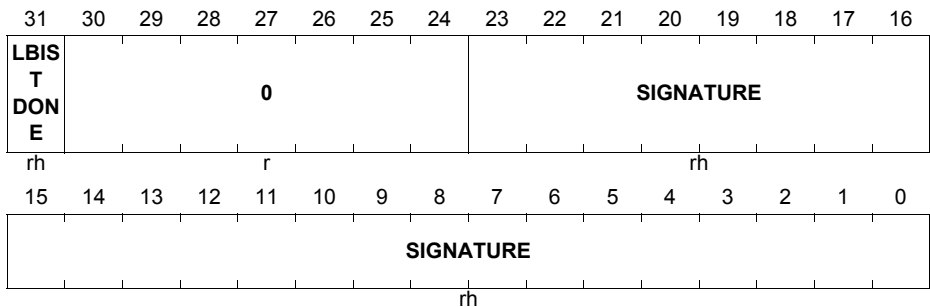
LBISTCTRL0 - Logic BIST Control 0 Register
LBISTCTRL0
Logic BIST Control 0 Register
(164_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
LBISTREQ	0	w	LBIST Request If written high this bit requests the execution of an automatic scan-test procedure. Whether the request is successful or not depends on whether the LBIST procedure has already been executed since the last power-on-reset. If read this bit always returns a '0'. <i>Note: LBIST execution time depends on the number of scan-loads as defined in the PATTERNS field.</i>
LBISTREQP	1	w	LBIST Request Protection Bit Protect LBISTREQ against unintended changes. 0 _B LBISTREQ is not changed. 1 _B LBISTREQ is updated by this write access.
PATTERNS	[15:2]	rw	LBIST Pattern Number This field defines the number of scan-patterns (i.e. scan-loads), which will be executed during the LBIST-procedure. The two LSBs of scan pattern number are always assumed as '1' so LBIST patterns number can only be controlled with an accuracy of 4.
0	[31:16]]	r	Reserved Read as 0; should be written with 0.

LBISTCTRL1 - Logic BIST Control 1 Register
LBISTCTRL1
Logic BIST Control 1 Register
(168_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SEED	[22:0]	rw	LBIST Seed This field determines which pattern is applied during LBIST execution.
0	23	r	Reserved Read as 0; should be written with 0.
SPLITSH	[26:24]	rw	LBIST Split-Shift Selection The value of this bit will allow to run LBIST with partitioned scan-shift operation in order to reduce the power consumption. 0x _B , Concurrent scan-shift is selected. 1x0 _B , Partitioned scan-shift is selected (four scan partitions). 1x1 _B , Partitioned scan-shift is selected (two scan partitions).
BODY	27	rw	Body Application Indicator The value of this bit will determine the static reset behavior of all GPIOs during LBIST execution. If set to low GPIOs will show a weak pull-up behavior, if set to high GPIOs are constrained to tri-state. A high value must be written to this bit in case LBIST shall be executed for body applications.

Field	Bits	Type	Description
LBISTFREQ	[31:28]	rw	LBIST Frequency Selection Through this register-field a pre-scaler factor between 1..16 is selectable for LBIST operation clock (derived from EVR-oscillator). This will allow to determine the LBIST scan-shift frequency. Value of these bits will be mirrored inside of LBIST-controller and become effective if a new LBIST-procedure has been successfully initiated via LBISTCTRL0.LBISTREQ.

LBISTCTRL2 - Logic BIST Control 1 Register
LBISTCTRL2
Logic BIST Control 2 Register
(16C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SIGNATURE	[23:0]	rh	LBIST Signature This field reflects the MISR signature from the last LBIST execution. It is mirrored from LBIST-controller and is only valid if LBISTCTRL0.LBISTDONE = 1.
0	[30:24]	r	Reserved Read as 0; should be written with 0.

Field	Bits	Type	Description
LBISTDONE	31	rh	LBIST Execution Indicator This bit indicates if since the last power-on-reset an automatic scan-test procedure has been executed: 0 _B No LBIST was executed since last power-on-reset. 1 _B LBIST was executed since last power-on-reset.

7.4.7 Global Overlay Controls

The following registers control the Global Overlay functionality.

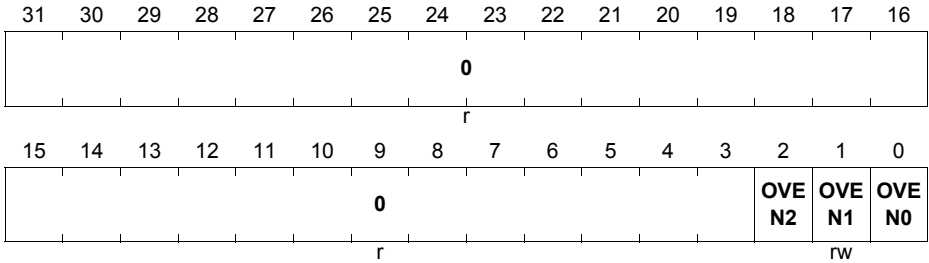
7.4.7.1 Global Overlay Control

Two registers globally control the overlay operation for all CPUs:

- Overlay Enable Register OVCENABLE disables or enables data access overlay individually for each CPU.
- Overlay Control Register OVCCON can be used to perform the following actions on a selected set of CPUs:
 - concurrently enable / disable selected overlay blocks,
 - concurrently disable overlay blocks,
 - invalidate data cache.

All overlay control registers are reset to their default values with the Application Reset . A special debug reset is not considered.

The external overlay feature is not available in product variants offering ADAS functionality.

OVCENABLE
Overlay Enable Register
(1E0H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
OVEN0	0	rw	Overlay Enable 0 0 _B OVC is disabled on CPU0. All Overlay redirections are disabled regardless of the state of OVC0_RABRy.OVEN. 1 _B OVC is enabled on CPU0.
OVEN1	1	rw	Overlay Enable 1 0 _B OVC is disabled on CPU1. All Overlay redirections are disabled regardless of the state of OVC1_RABRy.OVEN. 1 _B OVC is enabled on CPU1.
OVEN2	2	rw	Reserved in this Product
0	[31:3]	r	Reserved Read/write 0.

OVCCON
Overlay Control Register
(1E4H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						POV CON F	OV CON F	0					DC IN VAL	OV STP	OV ST RT
r						w	rw	r					w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												CSE L2	CSE L1	CSE L0	
r												w	w	w	

Field	Bits	Type	Description
CSEL0	0	w	CPU Select 0 0 _B CPU0 not affected, 1 _B Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU0. Return 0 if read.
CSEL1	1	w	CPU Select 1 0 _B CPU1 not affected, 1 _B Action selected by OVSTRT, OVSTP, DCINVAL bits, set by the same register write access, is applied to CPU1. Return 0 if read.
CSEL2	2	r	Reserved in this Product Return 0 if read.
0	[15:3]	r	Reserved Read/write 0.

Field	Bits	Type	Description
OVSTRT	16	w	Overlay Start 0 _B No action 1 _B For each CPU selected with CSEL, all the blocks selected with OVCx_OSEL will be activated. In the selected CPUs all the blocks deselected with OVCx_OSEL will be deactivated. CPUs which are not selected are not affected. No action is taken if OVSTP is also set. Return 0 if read.
OVSTP	17	w	Overlay Stop 0 _B No action 1 _B For CPUs selected with CSEL, all the overlay blocks are deactivated. OVCx_RABRy.OVEN bits are cleared. CPUs which are not selected are not affected No action is taken if OVSTRT is also set. Return 0 if read.
DCINVAL	18	w	Data Cache Invalidate No function in devices without data cache in CPU. 0 _B No action 1 _B Data Cache Lines in DMI are invalidated ¹⁾ Data Cache is affected only in the CPUs selected with CSEL. Return 0 if read.
0	[23:19]	r	Reserved Read/write 0.
OVCONF	24	rw	Overlay Configured Overlay configured status bit 0 _B Overlay is not configured or it has been already started 1 _B Overlay block control registers are configured and ready for overlay start This bit may be used as handshake bit between a debug device (via JTAG interface and Cerberus) and CPU(s).

Field	Bits	Type	Description
POVCONF	25	w	<p>Write Protection for OVCONF</p> <p>0_B OVCONF remains unchanged. 1_B OVCONF can be changed with write access to register OVCCON</p> <p>This bit enables OVCONF write during OVCCON write. Return 0 if read.</p>
0	[31:26]	r	<p>Reserved Read/write 0.</p>

- 1) Dirty (modified) cache lines are not effected by this operation. If data cache contains modified data, it is not invalidated, and has to be written-back and invalidated by the user. Therefore, it is highly recommended to either: access overlaid data in read-only mode, or use only non-cached access.

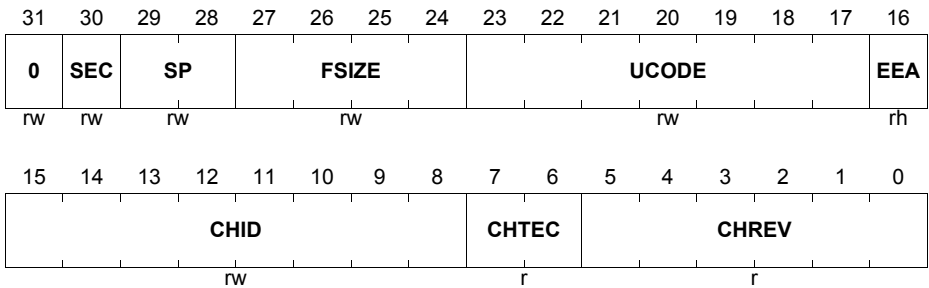
Field	Bits	Type	Description
0	[15:9],7,6,1	rw	Reserved Write only the read value
0	5	r	Reserved Read as 0
0	[31:16]	r	Reserved Read as 0

7.4.8.2 Identification Registers

The CHIPID register allows the user to determine the product, package option, FSI microcode version and chip revision (silicon step).

CHIPID

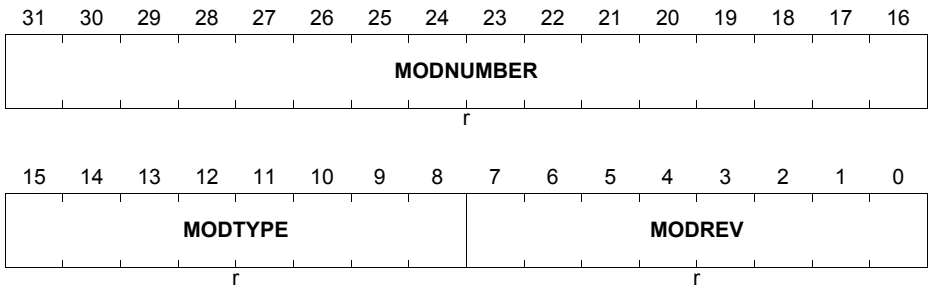
Chip Identification Register (140_H) **Reset Value: XXXX XXXX_H**



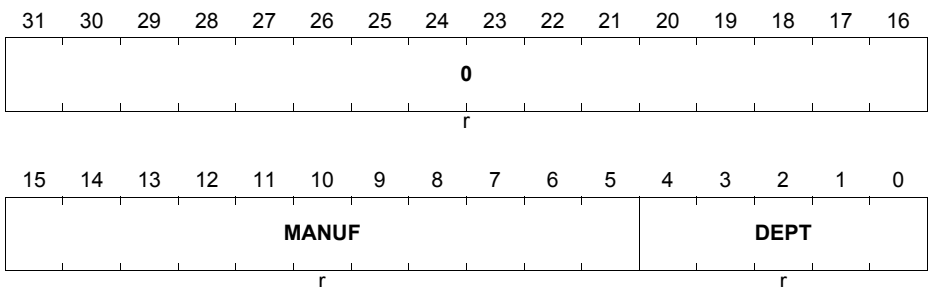
Field	Bits	Type	Description
CHREV	[5:0]	r	<p>Chip Revision Number</p> <p>This bit field indicates the revision number of the TC26x B-Step device. The value of this bit field is defined in the TC26x B-Step Data Sheet.</p> <p>Bits [3:0] are used to indicate the steps. These updates can be done with any metal-fix or FW ROM change.</p> <p>Bits [5:4] define the major silicon design steps (A, B, C, D). These bits can be changed only with a major redesign.</p> <p>Note that this specification refers to the latest available productive design step. Earlier silicon functionality may vary. Consult the appropriate delta documentation for details.</p> <p>00XXXX_B A-step silicon 01XXXX_B B-step silicon 10XXXX_B C-step silicon 11XXXX_B D-step silicon</p>

Field	Bits	Type	Description
CHTEC	[7:6]	r	Chip Family These bits indicate the product family and are changed only with a redesign. 00 _B Reserved 01 _B Aurix Gen1 Family 10 _B Reserved 11 _B Reserved
CHID	[15:8]	rw	Chip Identification Number This bit field defines the product by a unique number. This number may be used by SW to identify the system architecture (e.g. number of available CPUs). See Product Datasheet Addendum for more details.
EEA	16	rh	Emulation Extension Available Indicates if the emulation extension is available or not. 0 _B EEC is not available (TC2xxx_ or TC2xxxP) 1 _B EEC is available (TC2xxxE or TC2xxxF)
UCODE	[23:17]	rw	µCode Version This bit field displays the Version X.Y of the flash µCode.

Field	Bits	Type	Description
FSIZE	[27:24]	rw	<p>Program Flash Size</p> <p>This bit field indicates available program flash size for this device. Detailed information is shown in the Data Sheet.</p> <p>0000_B 256 KByte Program Flash (TC2xxx-4Fx) 0001_B 0.5 MByte Program Flash (TC2xxx-8Fx) 0010_B 1.0 MByte Program Flash (TC2xxx-16Fx) 0011_B 1.5 MByte Program Flash (TC2xxx-24Fx) 0100_B 2.0 MByte Program Flash (TC2xxx-32Fx) 0101_B 2.5 MByte Program Flash (TC2xxx-40Fx) 0110_B 3.0 MByte Program Flash (TC2xxx-48Fx) 0111_B 4.0 MByte Program Flash (TC2xxx-64Fx) 1000_B 5.0 MByte Program Flash (TC2xxx-80Fx) 1001_B 6.0 MByte Program Flash (TC2xxx-96Fx) 1010_B 7.0 MByte Program Flash (TC2xxx-112Fx) 1011_B 8.0 MByte Program Flash (TC2xxx-128Fx) 1100_B Reserved, do not use this combination 1101_B Reserved, do not use this combination 1110_B Reserved, do not use this combination 1111_B Reserved, do not use this combination</p>
SP	[29:28]	rw	<p>Speed</p> <p>This bit field indicates the maximum allowed frequency for the application CPU(s)</p> <p>00_B Default speed grade</p> <p>01_B Speed grade two</p> <p>10_B Speed grade three</p> <p>11_B Speed grade four</p>
SEC	30	rw	<p>Security Device</p> <p>This bit field indicates whether the product has a Hardware Security Module (HSM)</p> <p>0_B No Hardware Security Module (TC2xxxE or TC2xxx or TC2xxxA products) 1_B Hardware Security Module is available (TC2xxxF or TC2xxxP or TC2xxxB products)</p>
0	31	rw	<p>Reserved</p> <p>Must not be written.</p>

ID
Identification Register
(008_H)
Reset Value: 00C4 C081_H


Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number This bit field indicates the revision number of the Aurix SCU module (81 _H).
MODTYPE	[15:8]	r	Module Type This bit field is C0 _H . It defines a 32-bit module
MODNUMBER	[31:16]	r	Module Number This bit field defines the module identification number. The identification number for the Aurix SCU is 00C4 _H

MANID
Manufacturer Identification Register (144_H)
Reset Value: 0000 1820_H


Field	Bits	Type	Description
DEPT	[4:0]	r	Department Identification Number = 00 _H ; indicates the ATV Microcontroller department within Infineon Technologies.
MANUF	[15:5]	r	Manufacturer Identification Number This is a JEDEC normalized manufacturer code. MANUF = C1 _H stands for Infineon Technologies.
0	[31:16]	r	Reserved Read as 0.

7.4.8.3 SCU Access Restriction Registers

The Access Enable Register 0 restricts write access to all SCU registers so that they may only be written by specified bus masters (eg CPUs). See the Bus chapter for the mapping of TAG ID to specific system masters and CPUs).

ACCEN0

Access Enable Register 0

(3FC_H)

Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the SCU kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

ACCEN1

Access Enable Register 1

(3F8_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

7.4.9 SCU Register Address

Table 7-27 Registers Address Spaces - SCU Kernel Registers

Module	Base Address	End Address	Note
SCU	F003 6000 _H	F003 63FF _H	-

7.4.10 SCU Kernel Registers

This section describes the kernel registers of the SCU module. Most of SCU kernel register names described in this section will be referenced in other parts of the TC26x B-Step User's Manual by the module name prefix "SCU_".

SCU Kernel Register Overview

Table 7-28 Register Overview of SCU (Offset from Main Register Base)

Short Name	Long Name	Offset Addr. 1)	Access Mode		Reset	Description See
			Read	Write		
-	Reserved	000 _H - 00C _H	BE	BE	-	-
SYSCON	System Control Register	07C _H	U, SV	U, SV, P	System Reset	Page 7-307
-	Reserved	090 _H - 098 _H	BE	BE	-	-
DTSSTAT	Die Temperature Sensor Status Register	0E0 _H	U, SV	BE	Application Reset	Page 7-259
DTSCON	Die Temperature Sensor Control Register	0E4 _H	U, SV	U, SV, P	Application Reset	Page 7-258
WDTSCON0	Safety WDT Control Register 0	0F0 _H	U, SV	U, SV, 32, P	Application Reset	Page 7-277
WDTSCON1	Safety WDT Control Register 1	0F4 _H	U, SV	SV, SE, P	Application Reset	Page 7-280
WDTSSR	Safety WDT Status Register	0F8 _H	U, SV	BE	Application Reset	Page 7-288

Table 7-28 Register Overview of SCU (Offset from Main Register Base)

Short Name	Long Name	Offset Addr. 1)	Access Mode		Reset	Description See
			Read	Write		
EMSR	Emergency Stop Register	0FC _H	U, SV	SV, SE, P	Application Reset	Page 7-295
WDTCPU0CON0	CPU0 WDT Control Register 0	100 _H	U, SV	U, SV, 32,(CPU 0 ²)	Application Reset	Page 7-277
WDTCPU0CON1	CPU0 WDT Control Register 1	104 _H	U, SV	SV, CE0, P	Application Reset	Page 7-284
WDTCPU0SR	CPU0 WDT Status Register	108 _H	U, SV	BE	Application Reset	Page 7-288
WDTCPU1CON0	CPU1 WDT Control Register 0	10C _H	U, SV	U, SV, 32,(CPU 1)	Application Reset	Page 7-277
WDTCPU1CON1	CPU1 WDT Control Register 1	110 _H	U, SV	SV, CE1, P	Application Reset	Page 7-284
WDTCPU1SR	CPU1 WDT Status Register	114 _H	U, SV	BE	Application Reset	Page 7-288
–	Reserved	118 _H -120 _H	U, SV	–	–	
LCLCON0	CPU0 Lockstep Control Register	134 _H	U, SV	BE	Cold Power-On Reset	Page 7-253
LCLCON1	CPU1 Lockstep Control Register	138 _H	U, SV	BE	Cold Power-On Reset	Page 7-254
LCLTEST	Lockstep Test Register	13C _H	U,SV	U,SV, P	System Reset	Page 7-255
CHIPID	Chip Identification Register	140 _H	U, SV	BE	System Reset	Page 7-309
MANID	Manufacture Identification Register	144 _H	U, SV	BE	System Reset	Page 7-312
–	Reserved	148 _H - 160 _H	BE	BE	–	–

Table 7-28 Register Overview of SCU (Offset from Main Register Base)

Short Name	Long Name	Offset Addr. 1)	Access Mode		Reset	Description See
			Read	Write		
LBISTCTRL0	Logic BIST Control 0	164 _H	U, SV	SV, SE, P	System Reset	Page 7-298
LBISTCTRL1	Logic BIST Control 1	168 _H	U, SV	SV, SE, P	System Reset	Page 7-299
LBISTCTRL2	Logic BIST Control 2	16C _H	U, SV	BE	System Reset	Page 7-300
–	Reserved	194 _H	BE	BE	–	–
–	Reserved	1DC _H	BE	BE	–	–
OVCENABLE	Overlay Enable	1E0 _H	U,SV	SV,SE, P	Application Reset	Page 7-303
OVCCON	Overlay Control	1E4 _H	U,SV	SV, P	Application Reset	Page 7-304
–	Reserved	1E8 _H - 20C _H	BE	BE	–	–
EICR0	External Input Channel Register 0	210 _H	U, SV	U, SV, P	Application Reset	Page 7-241
EICR1	External Input Channel Register 1	214 _H	U, SV	U, SV, P	Application Reset	Page 7-241
EICR2	External Input Channel Register 3	218 _H	U, SV	U, SV, P	Application Reset	Page 7-241
EICR3	External Input Channel Register 4	21C _H	U, SV	U, SV, P	Application Reset	Page 7-241
EIFR	External Input Flag Register	220 _H	U, SV	BE	Application Reset	Page 7-245
FMR	Flag Modification Register	224 _H	U, SV	U, SV, P	Application Reset	Page 7-246
PDRR	Pattern Detection Result Register	228 _H	U, SV	BE	Application Reset	Page 7-247
IGCR0	Interrupt Gating Register 0	22C _H	U, SV	U, SV, P	Application Reset	Page 7-248
IGCR1	Interrupt Gating Register 1	230 _H	U, SV	U, SV, P	Application Reset	Page 7-248

Table 7-28 Register Overview of SCU (Offset from Main Register Base)

Short Name	Long Name	Offset Addr. 1)	Access Mode		Reset	Description See
			Read	Write		
IGCR2	Interrupt Gating Register 2	234 _H	U, SV	U, SV, P	Application Reset	Page 7-248
IGCR3	Interrupt Gating Register 3	238 _H	U, SV	U, SV, P	Application Reset	Page 7-248
–	Reserved	23C _H	BE	BE	–	–
DTSLIM	Die Temperature Sensor Limit Register	240 _H	U, SV	U, SV, P	Application Reset	Page 7-260
–	Reserved	244 _H - 3F4 _H	BE	BE	–	–
ACCEN1	Access Enable Register 1	3F8 _H	U, SV	SV, SE	Application Reset	Page 7-314
ACCEN0	Access Enable Register 0	3FC _H	U, SV	SV, SE	Application Reset	Page 7-314

1) The absolute register address is calculated as follows:

Module Base Address + Offset Address (shown in this column)

2) Some registers marked (CPUx) are writeable only from the specified CPU. For these registers the setting of the ACCEN bits have no effect. For all other registers the ACCEN bits control access. Attempted writes by other masters or CPUs will not be successful, but no bus error will result.

8 Memory Test Unit (MTU)

All of the internal memories within the TC26x B-Step have a unified interface for the control of ECC (Error Correction), BIST (Built-in-Self-Test) and redundancy features. This interface can also be used to fill the memories with a predefined data value for initialization. The Memory Test Unit (MTU) controls and monitors these memory test, initialization and data integrity checking functions.

The MTU comprises two sections:

- MTU configuration registers.
- Memory Controller related registers.

The configuration registers are used to enable and disable memory test functionality. For some memories, a special sequence must be followed in order to execute memory tests without compromising data security. This sequence is ensured by simple state machines which automatically prevent test access to the memories at times when data content may be sensitive.

The Memory Controller registers are replicated for each memory bank in the device. These registers control the individual MBIST, redundancy and ECC settings for that specific memory block.

8.1 Memory Content Initialization

In security applications it is important that the MTU does not permit reading or modification of restricted memory content via the MBIST modules. In other applications (e.g. safety) it is important that modification of memory content is possible via MBIST (e.g. for ECC error injection or runtime self-test).

The TC26x B-Step can be configured for either of these application options via FLASH0.PROCOND.

Note that if RAMs embedded inside IP modules are not configured for initialization at boot then software initialization may be required before the IP module can be used. (E.g. GTM, ERAY modules)

8.1.1 Non-Security Applications

If FLASH0.PROCOND.RAMIN=11 then no automatic initialization of RAM content is performed on a reset and no automatic initialization of RAM content is performed when MBIST modules are enabled or disabled with MTU_MEMTEST.MEMxEN registers.

This permits the use of the MBIST controller by an application (E.g. for error injection, data modification or runtime memory testing) without unwanted corruption of memory content.

8.1.2 Security Applications

If FLASH0_PROCOND.RAMIN = 00, 01 or 10 then automatic memory content initialization is enabled. PROCOND.RAMIN configures whether the initialization is triggered by cold resets, warm resets or both. In these modes, an automatic initialization of security-sensitive memories is also triggered whenever the corresponding MTU_MEMTEST.MEMxEN is changed (i.e. The MBIST controller is enabled or disabled).

The following MBIST Controller enables are considered to be security-sensitive in this product:

- FSI0
- CPU0PTEN
- CPU0PSEN (Initializes only the cache part of the memory)
- CPU1PTEN
- CPU1PSEN (Initializes only the cache part of the memory)
- CPU1DTEN
- CPU1DSEN (Initializes only the cache part of the memory)

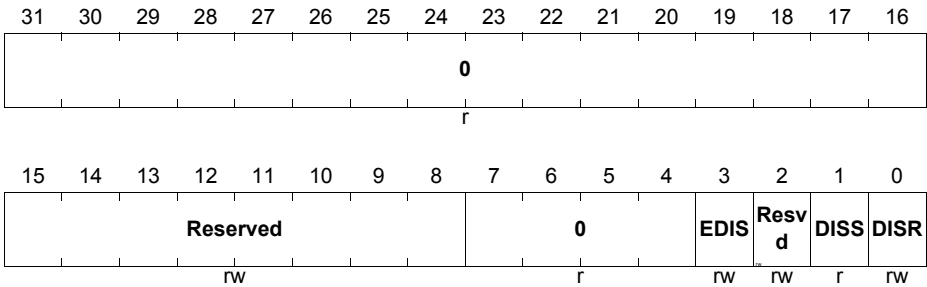
For security reasons, no MTU registers associated with HSM or FSI modules are accessible to the user after start-up.

8.2 Safety Notifications

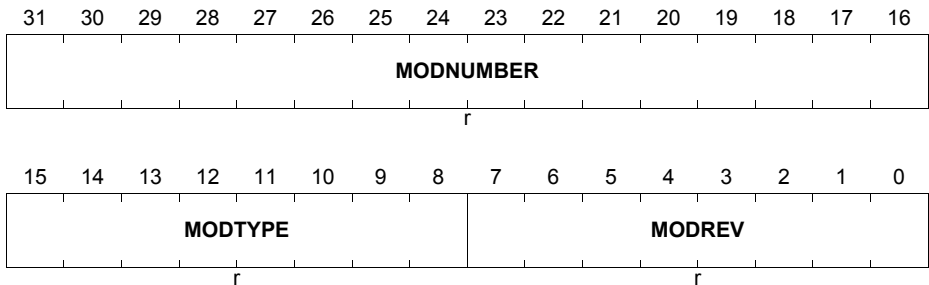
SRAM ECC errors from all system SRAMs may be reported to the Safety Management Unit (SMU). The SMU may be programmed to initiate appropriate action.

8.3 Memory Test Unit (MTU) Kernel Registers

These are the MTU control registers

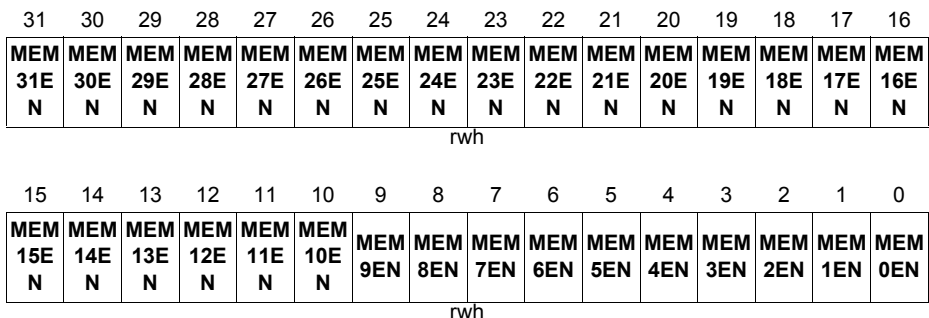
Memory Test Unit (MTU)
8.3.1 Descriptions
MTU_CLC
Identification Register
(F006 0000_H)
Reset Value: 0000 0003_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module. 0 _B Module disable is not requested 1 _B Module disable is requested
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module 0 _B Module is enabled 1 _B Module is disabled If the RMC field is implemented and if it is 0, DISS is set automatically.
Resvd	2	rw	Resvd Read as 0. Must be written with 0 _H
EDIS	3	rw	Sleep Mode Enable Control Used for module Sleep Mode control. 0 _B Sleep Mode request is regarded. Module is enabled to go into Sleep Mode on a request. 1 _B Sleep Mode request is disregarded: Sleep Mode cannot be entered on a request.
Reserved	[15:8]	rw	Reserved Read as 0. Must be written with 0 _H
0	[31:16], [7:4]	r	0 Read as 0.

Memory Test Unit (MTU)
MTU_ID
Identification Register
(F006 0008_H)
Reset Value: 00B2 C0XX_H


Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number This bit field indicates the revision number of the MTU module .
MODTYPE	[15:8]	r	Module Type This bit field is C0 _H . It defines a 32-bit module
MODNUMBER	[31:16]	r	Module Number This bit field defines the module identification number. The identification number for the Aurix MTU is 00B2 _H

The memory test register MEMTEST holds CPU configurable select bits for the various MBIST controllers.

MTU_MEMTEST0
Memory MBISTEnable Register 0 (F006 0010_H)
Reset Value: 0000 0000_H


Memory Test Unit (MTU)

Field	Bits	Type	Description
MEMxEN (x = 0-31)	x	rwh	Memory x MBIST Controller Memory Test Enable 0 _B Memory x MBIST controller is disabled 1 _B Memory x MBIST controller is enabled ¹⁾ Security Notes: For bits which represent security-sensitive memories an automatic auto-initialization of the associated memory x is triggered on every attempt to toggle the MEMxEN ²⁾ . Only after this initialization has completed will the value read back from this register bit show the updated value. Register bit MEMSTAT.AIUx provides an indication that the automatic initialization of memory x is underway. See the Implementation Section for a list of memories which are security-sensitive

1)

2)

The memory test register MEMTEST holds CPU configurable select bits for the various MBIST controllers. See the Integration Section for mapping of memory controller numbers.

MTU_MEMTEST1
Memory MBISTEnable Register 1 (F006 0014_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEM 63E N	MEM 62E N	MEM 61E N	MEM 60E N	MEM 59E N	MEM 58E N	MEM 57E N	MEM 56E N	MEM 55E N	MEM 54E N	MEM 53E N	MEM 52E N	MEM 51E N	MEM 50E N	MEM 49E N	MEM 48E N
rwh															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEM 47E N	MEM 46E N	MEM 45E N	MEM 44E N	MEM 43E N	MEM 42E N	MEM 41E N	MEM 40E N	MEM 39E N	MEM 38E N	MEM 37E N	MEM 36E N	MEM 35E N	MEM 34E N	MEM 33E N	MEM 32E N
rwh															

Memory Test Unit (MTU)

Field	Bits	Type	Description
MEMxEN (x = 32-63)	x-32	rwh	<p>Memory x MBIST Controller Memory Test Enable</p> <p>0_B Memory x MBIST controller is disabled 1_B Memory x MBIST controller is enabled¹⁾</p> <p>Security Notes: For bits which represent security-sensitive memories an automatic auto-initialization of the associated memory x is triggered on every attempt to toggle the MEMxEN ²⁾. Only after this initialization has completed will the value read back from this register bit show the updated value. Register bit MEMSTAT.AIUX provides an indication that the automatic initialization of memory x is underway. See the Implementation Section for a list of memories which are security-sensitive</p>

1)

2)

The memory test register MEMTEST holds CPU configurable select bits for the various MBIST controllers. See the Integration Section for mapping of memory controller numbers.

MTU_MEMTEST2

Memory MBISTEnable Register 2 (F006 0018_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res								MEM 87E N	MEM 86E N	MEM 85E N	MEM 84E N	MEM 83E N	MEM 82E N	MEM 81E N	MEM 80E N
r								rwh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEM 79E N	MEM 78E N	MEM 77E N	MEM 76E N	MEM 75E N	MEM 74E N	MEM 73E N	MEM 72E N	MEM 71E N	MEM 70E N	MEM 69E N	MEM 68E N	MEM 67E N	MEM 66E N	MEM 65E N	MEM 64E N
rwh															

Memory Test Unit (MTU)

Field	Bits	Type	Description
MEMxEN (x = 64-87)	x-64	rwh	Memory x MBIST Controller Memory Test Enable 0 _B Memory x MBIST controller is disabled 1 _B Memory x MBIST controller is enabled ¹⁾ Security Notes: For bits which represent security-sensitive memories an automatic auto-initialization of the associated memory x is triggered on every attempt to toggle the MEMxEN ²⁾ . Only after this initialization has completed will the value read back from this register bit show the updated value. Register bit MEMSTAT.AIUx provides an indication that the automatic initialization of memory x is underway. See the Implementation Section for a list of memories which are security-sensitive
Res	[31:24]	r	Reserved Read as 0.

1)

2)

The Memory Mapping Enable register MEMMAP has configurable control bits to select memory-mapped test mode. See the Integration Section for mapping of memory controller numbers.

MTU_MEMMAP
Memory Mapping Enable Register (F006 001C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM
31M	30M	29M	28M	27M	26M	25M	24M	23M	22M	21M	20M	19M	18M	17M	16M
AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP

rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM	MEM
15M	14M	13M	12M	11M	10M	9MA	8MA	7MA	6MA	5MA	4MA	3MA	2MA	1MA	0MA
AP	AP	AP	AP	AP	AP	P	P	P	P	P	P	P	P	P	P

rwh

Memory Test Unit (MTU)

Field	Bits	Type	Description
MEMxMAP (x = 0-31)	x	rwh	<p>MEMx Mapping Enable</p> <p>0_B Memory x functional 1_B Memory x memory-mapped (e.g. for test)</p> <p>Note that only CPU Cache Memory Mapping bits are implemented (See Implementation Section for details of used bits in this product)</p> <p>Security Notes: Caches are considered security-sensitive memories and an automatic auto-initialization of the associated memory x is triggered on every attempt to toggle the MEMxMAP bit ¹⁾. Only after this initialization has completed will the value read back from this register bit show the updated value. Register bit MEMSTAT.AIUX provides an indication that the automatic initialization of memory x is underway.</p>

1)

The memory status register MEMSTAT shows whether each MBIST controller is currently executing an automatic initialization sequence.

MTU_MEMSTAT0

Memory AutoinitializeStatus Register 0 (F006 0038_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEM 31AI	MEM 30AI	MEM 29AI	MEM 28AI	MEM 27AI	MEM 26AI	MEM 25AI	MEM 24AI	MEM 23AI	MEM 22AI	MEM 21AI	MEM 20AI	MEM 19AI	MEM 18AI	MEM 17AI	MEM 16AI
U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U

rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEM 15AI	MEM 14AI	MEM 13AI	MEM 12AI	MEM 11AI	MEM 10AI	MEM 9AIU	MEM 8AIU	MEM 7AIU	MEM 6AIU	MEM 5AIU	MEM 4AIU	MEM 3AIU	MEM 2AIU	MEM 1AIU	MEM 0AIU
U	U	U	U	U	U										

rh

Memory Test Unit (MTU)

Field	Bits	Type	Description
MEMxAIU (x = 0-31)	x	rh	Memory x MBIST AutoInitialize Underway 0 _B Memory x MBIST not running autoinitialize 1 _B Memory x MBIST running autoinitialize This bit indicates whether an automatic data initialization of Memory x has been triggered by a change of state of MEMTEST.MEMxEN or MEMxMAP but that the initialization sequence has not yet completed.

MTU_MEMSTAT1
Memory AutoinitializeStatus Register 1 (F006 003C_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEM 63AI U	MEM 62AI U	MEM 61AI U	MEM 60AI U	MEM 59AI U	MEM 58AI U	MEM 57AI U	MEM 56AI U	MEM 55AI U	MEM 54AI U	MEM 53AI U	MEM 52AI U	MEM 51AI U	MEM 50AI U	MEM 49AI U	MEM 48AI U

rh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEM 47AI U	MEM 46AI U	MEM 45AI U	MEM 44AI U	MEM 43AI U	MEM 42AI U	MEM 41AI U	MEM 40AI U	MEM 39AI U	MEM 38AI U	MEM 37AI U	MEM 36AI U	MEM 35AI U	MEM 34AI U	MEM 33AI U	MEM 32AI U

rh

Field	Bits	Type	Description
MEMxAIU (x = 63-32)	x-32	rh	Memory x MBIST AutoInitialize Underway 0 _B Memory x MBIST not running autoinitialize 1 _B Memory x MBIST running autoinitialize This bit indicates whether an automatic data initialization of Memory x has been triggered by a change of state of MEMTEST.MEMxEN or MEMxMAP but that the initialization sequence has not yet completed.

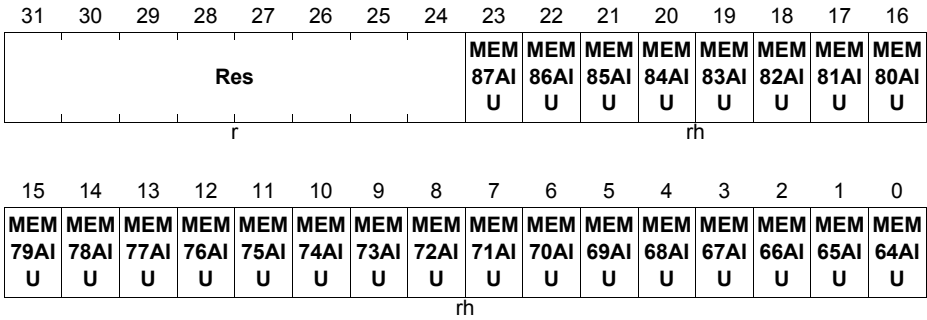
The memory status register MEMSTAT shows whether each MBIST controller is currently executing an automatic initialization sequence.

Memory Test Unit (MTU)

MTU_MEMSTAT2

Memory AutoinitializeStatus Register 2 (F006 0040_H)

Reset Value: 0000 0000_H



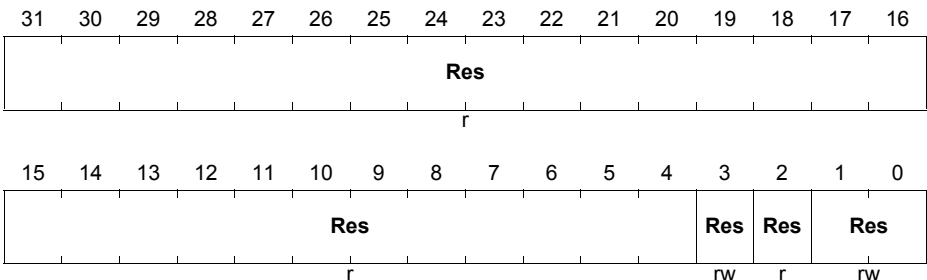
Field	Bits	Type	Description
MEMxAIU (x = 87-64)	x-64	rh	Memory x MBIST Autoinitialize Underway 0 _B Memory x MBIST not running autoinitialize 1 _B Memory x MBIST running autoinitialize This bit indicates whether an automatic data initialization of Memory x has been triggered by a change of state of MEMTEST.MEMxEN or MEMxMAP but that the intialization sequence has not yet completed.
Res	[31:24]	r	Reserved Read as zero

MTU_RES0

Reserved Register

(F006 0020_H)

Reset Value: 0000 0000_H

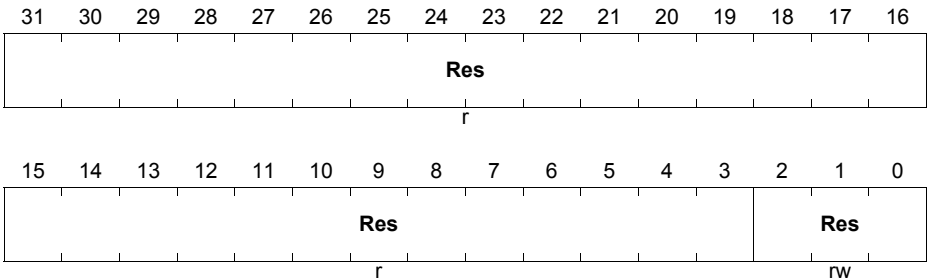


Memory Test Unit (MTU)

Field	Bits	Type	Description
Res	0,1,3	rw	Reserved in this Product
Res	[31:4],2	r	Reserved in this Product

MTU_RES1

Reserved Register (F006 0024_H) Reset Value: 0000 0000_H

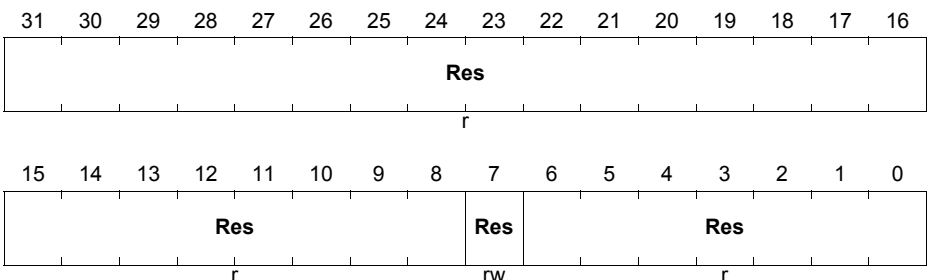


Field	Bits	Type	Description
Res	[31:3]	r	Reserved
Res	[2:0]	rw	Reserved in this Product

Note:

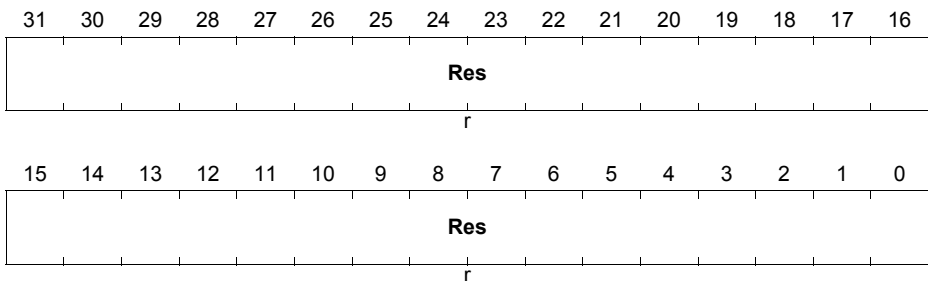
MTU_RES2

Reserved Register (F006 0030_H) Reset Value: 0000 0000_H



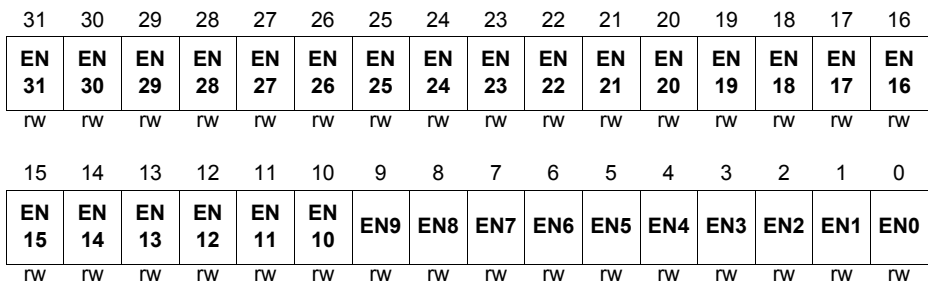
Memory Test Unit (MTU)

Field	Bits	Type	Description
Res	[31:8],[6:0]	r	Reserved
Res	7	rW	Reserved in this Product

MTU_RES3
Reserved Register
(F006 0034_H)
Reset Value: 0000 0000_H


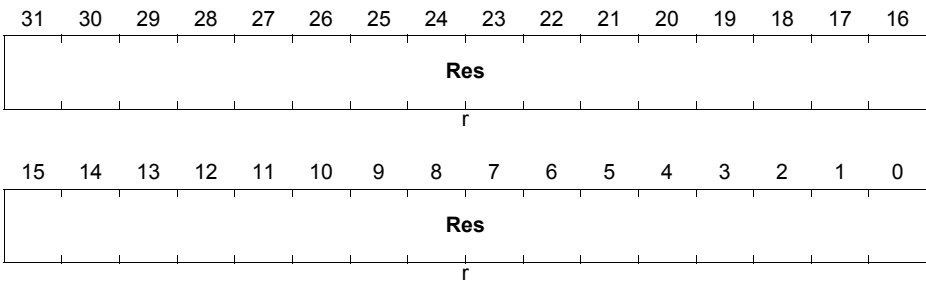
Field	Bits	Type	Description
Res	[31:0]	r	Reserved

The Access Enable Register 0 restricts write access to all MTU registers so that they may only be written by specified bus masters (eg CPUs). See the Bus chapter for the mapping of TAG ID to specific system masters and CPUs).

MTU_ACCEN0
Access Enable Register 0
(F006 00FC_H)
Reset Value: FFFF FFFF_H


Memory Test Unit (MTU)

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the MTU kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

MTU_ACCEN1
Access Enable Register 1
(F006 00F8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
Res	[31:0]	r	Reserved Read as 0; should be written with 0.

8.3.2 MTU Register Overview
Table 8-1 Register Overview of MTU Configuration register block

Short Name	Long Name	Offset Addr. ¹⁾	Access Mode		Reset	Description See
			Read	Write		
MTU_CLC	Reserved	00 _H	U,SV	SV,E,P	Applica tion Reset	
–	Reserved	04 _H	BE	BE	–	
MTU_ID	Identification Register	08 _H	U, SV	BE	–	Page 8-4

Memory Test Unit (MTU)
Table 8-1 Register Overview of MTU Configuration register block

Short Name	Long Name	Offset Addr. ¹⁾	Access Mode		Reset	Description See
			Read	Write		
–	Reserved	0C _H	BE	BE	–	
MTU_MEMTEST0	MBIST Enables 0	10 _H	U, SV	SV, SE,P	Application Reset	Page 8-4
MTU_MEMTEST1	MBIST Enables 1	14 _H	U, SV	SV, SE,P	Application Reset	Page 8-5
MTU_MEMTEST2	MBIST Enables 2	18 _H	U, SV	SV, SE,P	Application Reset	Page 8-6
MTU_MEMMAP	Memory Mapping Control	1C _H	U, SV	SV, SE,P	Application Reset	Page 8-7
–	Reserved Registers	20 _H -34 _H	-	-	–	–
MTU_MEMSTAT0	Memory AutoInit Status Register 0	38 _H	U, SV	BE	Application Reset	Page 8-8
MTU_MEMSTAT1	Memory AutoInit Status Register 1	3C _H	U, SV	BE	Application Reset	Page 8-9
MTU_MEMSTAT2	Memory AutoInit Status Register 2	40 _H	U, SV	BE	Application Reset	Page 8-10
–	Reserved	44 _H -F4 _H	BE	BE	–	
MTU_ACCESS1	Access Enable Register 1	F8 _H	U, SV	BE	Application Reset	Page 8-13
MTU_ACCESS0	Access Enable Register 0	FC _H	U, SV	SV, SE	Application Reset	Page 8-12

1) The absolute register address is calculated as follows:
Memory Module Register Base Address + Offset Address (shown in this column)

8.4 Memory Controllers

8.4.1 Control and Status Interfaces

8.4.1.1 Direct CPU Interface

There is a dedicated interface to the registers of each MBIST/ECC controller.

Memory Test Unit (MTU)

Register Mapping

Registers have an MTU system address and can be accessed through normal 16 bit SPB bus accesses. In the MTU the registers appear as if they were 16 bits wide.

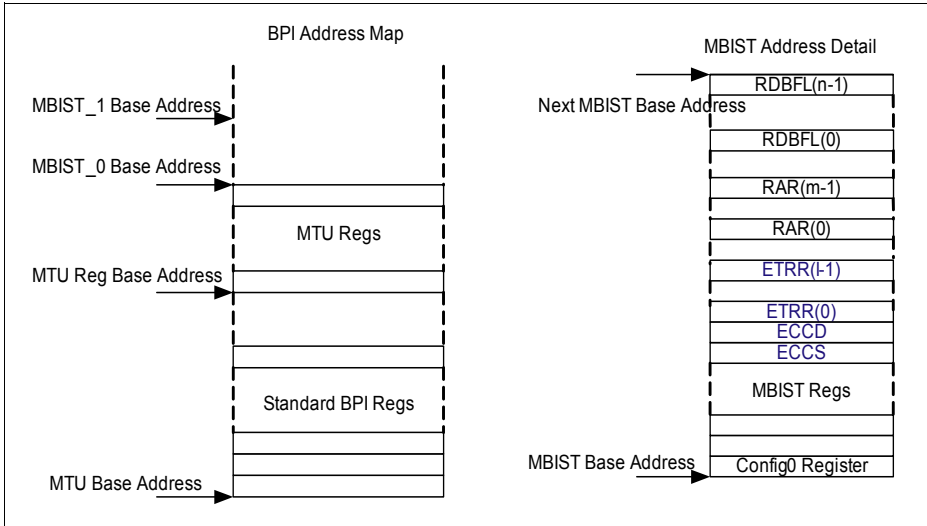


Figure 8-1 MBIST/ECC Register Mapping Scheme

8.4.2 Registers

8.4.2.1 MBIST/ECC Registers

Some register field sizes or content depend upon the physical sizes of the memories. The default settings enable fill/test of the entire physical RAM and the register descriptions show the maximum bitfield sizes.

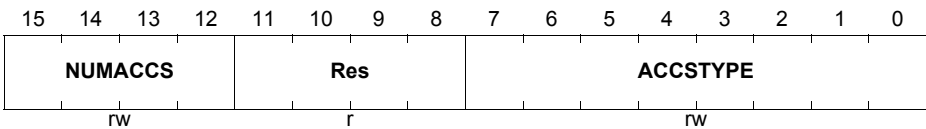
Configuration Registers

The bits in these registers can be used to control and program any march and hammer sequences. All bits concerning these test are concentrated here. All bits do not change during a test run. MCONTROL.START will start the tests defined here. MSTATUS.DONE is reset at the beginning of a test and set after completion. If no legal

Memory Test Unit (MTU)

operation was defined in CONFIG1.AG_MOD nothing will be done but the handshake of MCONTROL.START and MSTATUS.DONE is carried out.

The reset values of the CONFIG0/1 and MCONTROL registers will perform a $\{\uparrow(w0,r0)\}$ operation (direction up, write 0 to all cells and check for 0) which initializes the whole memory with 0 and checks the contents if MCONTROL.START is set. This is the start sequence of many tests.

CONFIG0
Configuration Register 0
(00_H)
Reset Value: 2002_H


Field	Bits	Type	Description
NUMACCS	15:12	rw	Number of accesses per address This field specifies the total number of accesses which are being performed to each single address in the current marching element. Range is from 0 to 15. With the sizes of ACCSPAT and ACCSTYPE given only 0 to 8 make sense. NUMACCS=0 will not access a memory.
Res	11:8	r	Reserved Reads return 0
ACCSTYPE	7:0	rw	Access type This field specifies the type of access which is being performed to each single address in the current marching element. ACCSTYPE[n] specifies n-th access of the marching element. 0 _B , write access 1 _B , read access

Memory Test Unit (MTU)

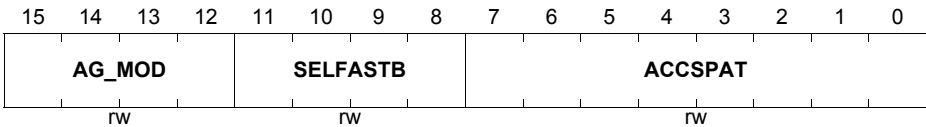
Configuration Register 1

CONFIG1

Configuration Register 1

(02_H)

Reset Value: 0000_H



Field	Bits	Type	Description
AG_MOD	15:12	rw	<p>Address Generator Mode</p> <p>These bits enable the special hardware for performing the more complex addressing schemes.</p> <p>In case RANGE.RAEN (range enable) is set to 0 (single access) linear address mode has to be selected and NUMACCS set to 1.</p> <p>0000_B, run the test with linear address generation</p> <p>0001_B, run the right half select test</p> <p>0010_B, run the test with GALPAT9 algorithm</p> <p>0011_B, run the left half select test</p> <p>0100_B, run the test with the GALPAT5 algorithm</p> <p>0101_B, run the non-destructive inversion test</p> <p>1000_B, run the write mask test</p> <p>1010_B, run the test with 2ⁱ address generation</p> <p>others, Nothing is done but the handshake of START and DONE is carried out.</p>

Memory Test Unit (MTU)

Field	Bits	Type	Description
SELFSTB	11:8	rw	<p>Select Fast Bit</p> <p>This field defines during a 2^i test the address bit position that has the Hamming distance of 1, i. e. changes fastest. Bit 0 of either column or row address is swapped with the indicated bit of either column or row according to MCONTROL.RCADR. MCONTROL.RCADR=0 -> column MCONTROL.RCADR=1 -> row 0000_B, normal addressing sequence, bit 0 in its normal position.</p> <p>others, bit 0 swapped with the indicated position.</p>
ACCSPAT	7:0	rw	<p>Access pattern</p> <p>This field specifies directly the bit pattern which is being used for an access to each single address in the current marching element. ACCSPAT[n] specifies the n-th access of the marching element. These patterns are toggled according to BITTOG and ROWTOG.</p>

Memory Test Unit (MTU)

MBIST Control Register

The bits in this register control the behavior of the MBIST, start and define tests to be performed. Any test will be performed and started when the startbit is set. The end will be signalled by MSTATUS.DONE . .

Required data (e.g. march elements, select fast bit) are taken from CONFIG0 and CONFIG1.

None of the tests are interruptable.

“Fast Row”, “Fast Column” (formerly called Fast Y and Fast X resp.) are defined as follows: “Fast Row” moves along the word-lines first and then in bit-line direction, “Fast Column” along the bit-lines first.

Note: Writes to this register are only permitted when no test is under way.

MCONTROL

MBIST Control Register

(04_H)

Reset Value: 4008_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	Res4	Res	FAIL DMP	GP_BAS E	BITT OG	ROW TOG	RCA DR	DINI T	DIR	EST F	RES UME	STA RT			
r	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
Res	15	r	Reserved Reads return 0
Res4	14:10	r	Reserved Read returns 0x4 Must always be written with 0x4

Memory Test Unit (MTU)

Field	Bits	Type	Description
FAILDMP	9	rw	<p>Fail bitmap dump</p> <p>This field enables a dump of the failing address and a fail bit map after a fault has been detected. The memory test is suspended afterwards and resumed by RESUME. MSTATUS.FDA shows that a fail dump is available. This functionality can be used only if bit LDRED = 1. In case a fail dump is available, RDBFL will contain the fail bit map and ETRR the failing address. ECCS/ECCD.AENE shouldn't be set at the same time as MCONTROL.FAILDUMP is set because wrong address errors are notified when a fail dump is available.</p> <p>0_B Do not dump 1_B Dump each fault</p>
GP_BASE	8	rw	<p>Galpat Base</p> <p>This bit defines the value which is written into the base cell during the galpat mode. BITTOG and ROWTOG modify the actual value written.</p> <p>0_B Write a zero into the base cell prior to reading the background pattern from the adjacent cells 1_B Write a one into the base cell prior to reading the background pattern from the adjacent cells</p>
BITTOG	7	rw	<p>Bit toggling</p> <p>This field specifies whether to toggle the used bit pattern (non inverted/inverted) with each physical memory column. This is required when writing a checkerboard pattern or a column stripe pattern.</p> <p>0_B Do not toggle 1_B Do toggle with each column</p>
ROWTOG	6	rw	<p>Row toggling</p> <p>This field specifies whether to toggle the used bit pattern (non inverted/inverted) with each physical memory row. This is required when writing a checkerboard pattern or a row stripe pattern. Setting both BITTOG=1 and ROWTOG=1 would not make sense. The user must ensure that only one of these is set.</p> <p>0_B Do not toggle 1_B Do toggle with each row</p>

Memory Test Unit (MTU)

Field	Bits	Type	Description
RCADR	5	rw	Fast Row / Fast Column Addressing Scheme Select This bit selects between fast row and fast column addressing. It is compatible with the MBP 4.2 X/Y definitions. 0 _B Fast row 1 _B Fast column
DINIT	4	rw	Data Initialization Enable This bit enables a write of the RDBFL data to all locations defined by the range register. RDBFL can contain data that will produce an ECC error. Execution is started with MCONTROL.START. For this predefined action any information contained in CONFIG0/1 registers and the bits BITTOG, ROWTOG and DIR are ignored. 0 _B Disabled 1 _B Enabled
DIR	3	rw	Direction Select This field specifies the direction of a memory test operation. 0 _B Address direction is highest to lowest. 1 _B Address direction is lowest to highest.
ESTF	2	rw	Enable Sticky Fail Bit This bit enables the sticky fail bit MSTATUS.SFAIL. If set any fails will be collected in MSTATUS.SFAIL. Resetting this bit to 0 will also reset MSTATUS.SFAIL. 0 _B Do not collect fail events 1 _B Collect fail events
RESUME	1	rwh	Resume failed test This bit allows a test with fail that got suspended to be resumed after the dump of the fail bit map. A restart is possible only if MSTATUS.FDA was reset by hardware. It will be reset by hardware once the test is resumed. 0 _B Do not resume 1 _B Resume suspended MBIST run

Memory Test Unit (MTU)

Field	Bits	Type	Description
START	0	rw	<p>START</p> <p>If this bit is written to '1' by software the memory test will start. If it is reset by software, and the test has finished, MSTATUS.DONE will be set to 1.</p> <p>If MCONTROL.FAILDMP is set, a fail will stop the current execution. RESUME will continue a suspended test.</p> <p>1_B Start memory test 0_B No test started, finished or waiting for test end</p>

Fail Dump

Bit FAILDMP enables a dump of the failing address and a fail bit map after a fault has been detected. The memory test is suspended afterwards and resumed by RESUME. MSTATUS.FDA shows that a fail dump is available.

This bit has to be polled by the tester hardware as there is no signal that shows a stopped test.

Memory Test Unit (MTU)

MBIST Status Register

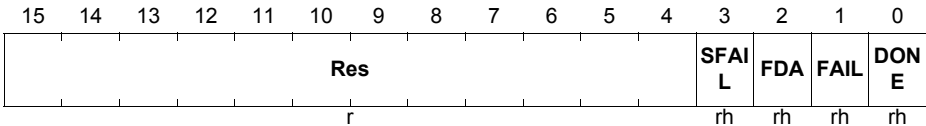
The bits in the status register show the status of the currently running and last test respectively.

MSTATUS

Status Register

(06_H)

Reset Value: 0001_H



Field	Bits	Type	Description
Res	15:4	r	Reserved Reads return 0
SFAIL	3	rh	Sticky Fail Bit This bit is set to 1 together with FAIL provided MCONTROL.ESTF is set. In contrast to FAIL it will not be reset when a new test is started. Therefore it will collect fail information over more than one MBIST run. It will be reset when MCONTROL.ESTF is reset or the 0 _B , No fail collected. 1 _B , A fail occurred during one of the test runs since MCONTROL.ESTF was set to 1.

Memory Test Unit (MTU)

Field	Bits	Type	Description
FDA	2	rh	<p>Fail Dump Available</p> <p>This bit shows that a fail has occurred if MCONTROL.FAILDMP is set. The test is suspended and fail dump information is available. The fail bit map is in RDBFL and the associated address is in ETRR(0). As long as no fail has occurred RDBFL contains the last read information and ETRR has no valid data . This bit will be set by hardware.</p> <p>It will be reset when MSTATUS was read with MSTATUS.FDA = 1 and the dump information was read from ETRR and RDBFL. Only the last read from the last word of RDBFL is checked by the hardware and taken as an indication for a complete read.</p> <p>A suspended test will be resumed by MCONTROL.RESUME if FDA was reset.</p> <p>This forms some sort of handshake to insure that a suspended test can only be resumed (by a broadcasted) MCONTROL.RESUME if the last fail information was actually collected.</p> <p>0_B , No fail dump data available. A suspended MBIST run can be resumed.</p> <p>1_B , Fail dump data is available and waiting for read.</p>
FAIL	1	rh	<p>FAIL</p> <p>This bit will be reset when a test is being started. It will be set to '1' by hardware under the following conditions: FAIL:='1' if the memory has at least one fault</p>
DONE	0	rh	<p>DONE</p> <p>Reset value is 1. This bit is reset at the start of a test and set when a test is completed and MCONTROL.START was reset by software. It is not set when a test is interrupted for fail dump.</p>

Memory Test Unit (MTU)

Range Register

The Range Register can be used to run a test only on a dedicated part of the RAM. This can be helpful to shorten the time which is needed to find a defect cell in a FAR where there's already some information available about a failing memory cell because it's not necessary to run the test using the full address range of the RAM under test.

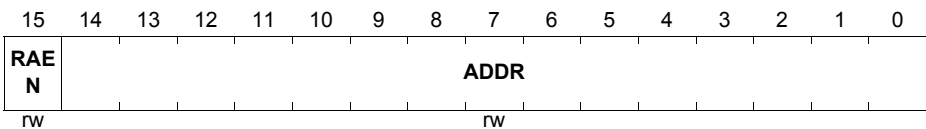
The range can be set in 64 word increments.

The range register can also be used to write to one specific logical address or read from it. In this case the range is disabled and the remaining part of the register is used as the logical address field.

RANGE

Range Register, single address mode (08_H)

Reset Value: xxxx_H



Field	Bits	Type	Description
RAEN	15	rw	<p>Range Enable</p> <p>0 disabled, single address mode. In this case a single word can be addressed for read or write. Config registers have to be set as follows CONFIG.NUMACCS:= "0001" (single access) CONFIG.AG_MOD := "0000" (linear) MCONTROL.DIR :=1 (up) For read just the value in this location will be delivered. No check against expected values is made; i.e. MSTATUS.FAIL will not be set.</p>
ADDR	14:0	rw	<p>Address</p> <p>This field specifies the logical address (before descrambling) of a single memory location. Reads and writes to this location are possible.</p>

Setting Address Ranges

If the RAEN field is set to '1' then range mode is enabled. In this case, the ADDR field of the RANGE register is interpreted as two separate fields:

Memory Test Unit (MTU)

ADDR[14:7] is interpreted as the Upper Range Limit.

This field specifies the upper logical block address limit in 64 word increments.

Upper end of the address range is UPLIMIT & 111111B

ADDR[6:0] is interpreted as the Lower Range Limit

This field specifies the upper logical block address limit in 64 word increments.

Lower end of the address range is LOLIMIT & 000000B

Note that the default reset value of the ADDR field will be set to the maximum range of the physical memory. The default behaviour is therefore that an initialization or test will operate over the whole memory.

Revision ID Register

The revision ID register contains a hard coded read only constant which describes the current status of the MBIST/ECC IP.

REVID

Revision ID Register (0C_H) **Reset Value: 0510_H**



Field	Bits	Type	Description
REV_ID	15:0	r	<p>Revision Identifier</p> <p>This field defines the currently implemented release, version and functionality of the used MBIST/ECC controller to track the MBIST/ECC version for easier handling at the tester.</p>

ECC Safety Register

This register controls the various error detection and notification modes.

Writes to this register are only permitted when `safe_endinit= 0` and no test is under way.

The hardware features that implement fault tolerance mechanisms for the SRAMs (e.g. single bit correction) shall be enabled per default after any reset. This is ensured by the reset value of the ECC safety register where `ECCS.ECE = 1` after a reset.

ECCS
ECC Safety Register
(0E_H)
Reset Value: 00XF_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res			TC_WAY_SEL	ECCMAP	SFLE	BFL E	TRE	ECE	AEN E	UEN E	CEN E				
r			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
Res	15:12	r	Reserved Reads return 0
TC_WAY_SEL	11:10	rw	TriCore Cache Way Select TriCore Cache only. These two bits select a cache way to run the non-destructive inversion test on. The bits represent the way number.
ECCMAP	9:8	rw	ECC Bit Mapping Mode ECCMAP sets three different test modes to allow access to data or ECC bits separately and independently. 00 _B Normal operation 01 _B Test mode. Only data bits mapped. All ECC functionality disabled. 10 _B Test mode. ECC check bits mapped to lower data bit positions. Other bits read as zero. All ECC functionality disabled. Data bits are not affected by write operations. 11 _B not allowed.

Memory Test Unit (MTU)

Field	Bits	Type	Description
SFLE	7:6	rw	Signature Bit Flip Enables If any SFLE bit is set no error tracking will take place. 00 _B Normal operation 01 _B Test mode only. Flips bitline address check signature bits. 10 _B Test mode only. Flips wordline address check signature bits. 11 _B Test mode only. Combination makes no sense.
BFLE	5	rw	Bit Flip Enable 0 _B Normal operation 1 _B Test mode only. Flips data and check bits according to RDBFL.
TRE	4	rw	Tracking Enable All errors will be tracked, if the associated notification enable bit is set. For TriCore memories this bit is set by default 0 _B Do not track address of detected error 1 _B Track address of detected error
ECE	3	rw	Error Correction Enable This bit shadows ECCD.ECE. Either bit can be written and both bits will read the last written value. 0 _B Do not correct correctable errors 1 _B Correct correctable errors
AENE	2	rw	Address Error Notification Enable This bit shadows ECCD.AENE. Either bit can be written and both bits will read the last written value. 0 _B Do not report address errors 1 _B Report detected address errors
UENE	1	rw	Uncorrectable Error Notification Enable This bit shadows ECCD.UENE. Either bit can be written and both bits will read the last written value. 0 _B Do not report uncorrectable data errors 1 _B Report detected uncorrectable errors to SMU
CENE	0	rw	Correctable Error Notification Enable This bit shadows ECCD.CENE. Either bit can be written and both bits will read the last written value. 0 _B Do not report correctable data errors 1 _B Report detected correctable errors to SMU

Memory Test Unit (MTU)
ECC Detection Register

The ECC detection register contains information on the errors detected and the tracking register clear.

Bits EOv, AERR, UERR and CERR are signals as well. While the bits are cleared by software write, the signals have a handshake associated to cater for different clock domains.

ECCD
Memory ECC Detection Register
(10_H)
Reset Value: 7800_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EOV	ECE	AEN E	UEN E	CEN E	Res	VAL					TRC	AER R	UER R	CER R	SER R
rh	rw	rw	rw	rw	rwh	rh					w	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
EOV	15	rh	Error Overflow 0 _B all errors detected since last clear were tracked. 1 _B more errors were detected since last clear than error tracking registers are available. Also, this bit is set if more than one memory block was in error at the same time. See ETRR.MBI for details. Enabled by ECCS.TRE and reset by ECCD.TRC
ECE	14	rw	Error Correction Enable Note that this bit is only writeable when Safety Endinit = 0. Other writes will not change this bit. 0 _B Do not correct correctable errors 1 _B Correct correctable errors

Memory Test Unit (MTU)

Field	Bits	Type	Description
AENE	13	rw	<p>Address Error Notification Enable This bit enables ECCD.AERR and the hardware flag to SMU. Note that this bit is only writeable when Safety Endinit = 0. Other writes will not change this bit. AENE shouldn't be set at the same time as MCONTROL.FAILDUMP is set because wrong address errors are notified when a fail dump is available.</p> <p>0_B Do not report address errors 1_B Report detected address errors</p>
UENE	12	rw	<p>Uncorrectable Error Notification Enable This bit enables ECCD.UERR and the hardware flag to SMU. Note that this bit is only writeable when Safety Endinit = 0. Other writes will not change this bit.</p> <p>0_B Do not report uncorrectable data errors 1_B Report detected uncorrectable errors to SMU</p>
CENE	11	rw	<p>Correctable Error Notification Enable This bit enables ECCD.CERR and the hardware flag to SMU. Note that this bit is only writeable when Safety Endinit = 0. Other writes will not change this bit.</p> <p>0_B Do not report correctable data errors 1_B Report detected correctable errors to SMU</p>
VAL	9:5	rh	<p>Valid Bits Every tracking register has a valid bit associated. Reset by ECCD.TRC. Up to five error tracking registers are available and five valid bits. Unused valid bits always read 0.</p>
TRC	4	w	<p>Tracking Clear Writing this bit with '1' clears the EOV, VAL bits plus the tracking registers. This bit will always read 0.</p> <p>0_B No effect 1_B Clear bits</p>

Memory Test Unit (MTU)

Field	Bits	Type	Description
AERR	3	rwh	Address Error Detected Write of '0' clears the sticky status. Write of '1' has no effect. A write does not clear the hardware flag. ¹⁾ Read as: 0 _B No address error detected 1 _B Address error detected (AERR)
UERR	2	rwh	Uncorrectable Error Detected Write of '0' clears the sticky status. Write of '1' has no effect. A write does not clear the associated hardware flag. Read as: 0 _B No uncorrectable error detected 1 _B Uncorrectable error detected (UERR)
CERR	1	rwh	Correctable Error Detected Write of '0' clears the sticky status. Write of '1' has no effect. A write does not clear the associated hardware flag. Read as: 0 _B No correctable error detected 1 _B Correctable error detected (CERR)
SERR	0	rwh	Error Detected Write of '0' clears the sticky status. Write of '1' has no effect. Read as: 0 _B No error detected 1 _B Error detected: any error detected :CERR, UERR or AERR

1) Note: A write to AERR, UERR and CERR does not clear the hardware flags.

Memory Test Unit (MTU)

Read Data and Bit Flip Register

This register is used for several purposes whenever a register with the size of the memory width is needed.

Normally this register contains the data which are directly read back from the RAM (without any data scrambling) during the last read access.

During logic test (LBIST) it contains the bit flip information. If ECCS.BFLE is set, this information is used to flip bits written to the SRAM.

After a failed test it contains the failed bit map if MCONTROL.FAILDMP is set. The address part is contained in the error tracking register.

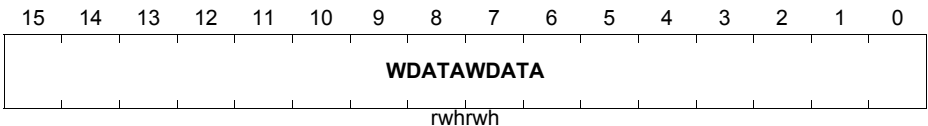
When MCONTROL.DINIT is set, this information is written to all locations within range.

Writes to this register are not permitted while a test is underway.

RDBFLy (y=0-39)

Read Data and Bit Flip Register y (A0_H + y*02_H)

Reset Value: 0000_H



Field	Bits	Type	Description
WDATA	15:0	rwh	Word Data This field contains the data of the last memory read operation. The width of WDATA does always match the memory data width. $m = nb_mems_g * (n + k)$

8.4.3 Safety Features

Several Safety features are implemented in conjunction with ECC and Safety SRAMs.

There are three signals to provide an alarm to the SMU:

correctable error detected

uncorrectable error detected

address error detected. Address errors are detected by the Safety SRAM feature. This discovers all wordline failures including broken wordlines and decoder malfunctions.

The codes used for ECC are of the SECDED (single error correction double error detection) type..

Memory Test Unit (MTU)

All codes are able to detect all zeroes and all ones, despite the fact that all physical faults resulting in this condition are covered by the Safety SRAM feature.

Whenever a new error is detected, not only a signal is set as an alarm, also an associated bit is set in ECCD. Three bits available are ECCD.CERR, ECCD.UERR and ECCD.AERR. They are cleared by writing a 0 to their location.

Bit ECE switches error correction on and off.

All errors are being tracked in error tracking registers (1 to 4). Correctable and uncorrectable errors are treated the same way except for the alarms and notification bits.

If an error is tracked then the associated register contains a valid value and a valid bit is set ECCD.VAL

The valid bits can be cleared by writing to ECCD.TRC

A valid bit is associated with each error entry. Bit ECCD.EOV is set when the error tracking register has recorded the maximum number of errors already and another error gets detected before the error tracking register is read by software and the event cleared by ECCD.TRC. All errors are treated the same way.

Uncorrectable errors are treated the same way as correctable errors and stored in the same registers.

Also, valid bits are used the same way.

ECCS.ECCMAP allows to select separately either the data bits of the stored information or the ECC check bits or data bits with ECC bits operational. In normal mode (ECCS.ECCMAP = 00) only access to data with correctly enabled ECC is possible and the ECC bits as such are neither visible nor accessible.

This mode (ECCS.ECCMAP ≠ "00") cannot be left without prior changing safe_endinit.

Bit will switch on special test modes to inject errors into the ECC encoder to ECC decoder path. This way the redundancy of this path can be broken and the logic tested.

Also, the ECC can be tested by firmware by writing illegal codes into the ECC data fields.

Then any bit combination can be written to the ECC bits.

There is a Safety SRAM feature implemented to detect errors that involve more errors that can be detected by ECC. Also, there are signature bit flip enables provided to test the logic of address error detection.

The Safety SRAM feature will also detect the failure mode where a valid SRAM access request present at the SRAM input is not processed by the SRAM, as long as the attempted access does not have the same address as the last functioning valid access to the SRAM

The Safety SRAM feature detects addressing faults for both read and write accesses. Atomic read-modify-write sequences are also considered.

Memory Test Unit (MTU)

When such an error occurs or when a bit flip enable is set, an address error is flagged, provided ECCS.AENE is set.

Normally SRAM, MBIST and ECC are delivered as block. If the ECC decoder and/or decoder are embedded in a pipeline, then the blocks can be separated.

8.4.4 Operation Modes

8.4.4.1 Starting a Memory Test Sequence (example)

It's not possible any longer to start simple standard tests such as Checkerboard, Row Stripe, March C+ and March U without writing special values to the CONFIG registers. These tests now also require to write dedicated values into the CONFIG registers as described in the following example:

1. Enter memory test mode
2. Initialize registers
Write the desired march sequence into the CONFIG registers
CONFIG1.ACCSPAT is the pattern to be executed, CONFIG0.ACCSTYPE the read/write sequence and CONFIG0.NUMACCS the number of bits in CONFIG1.ACCSPAT and CONFIG0.ACCSTYPE to used
AG_MOD contains the address mode to be used
RANGE := <memory range to be tested>
3. Start memory test by writing a 1 to register bit MCONTROL.START
4. Reset MCONTROL.START
5. Wait for the end of the test
 - poll MSTATUS.DONE
 - or poll primary output DONE
6. Check the result of the test
 - verify MSTATUS.FAIL
 - or verify primary output FAIL

Complex algorithm may require several march starts to get a full test.

8.4.4.2 Getting Detailed Memory Test Results

The FAIL and DONE bits provide a general pass/fail information.

If MCONTROL.FAILDMP = '1', the test stops after a failed test and the fail information is immediately available for dump. MSTATUS.FDA (fail dump available) is set and signal faildumpreq asserted in this case. RDBFL contains the fail bit map and ETRR the failed address. Reading MSTATUS with MSTATUS.FDA = '1' and RDBFL(n-1) will reset MSTATUS.FDA and faildumpreq. A consequent setting of MCONTROL.RESUME will restart the interrupted test sequence.

The RANGE register can be used to run consecutive tests on constantly shifted memory ranges so that in the end the complete memory has been analyzed.

8.4.4.3 Dumping Fail Bitmap

Any dump information has to be polled from registers RDBFL and ETRR(0).

8.4.4.4 Filling a Memory with Defined Contents

The MBIST/ECC can be used to fill a memory range or a complete memory with a defined pattern very fast, i.e. one write access per cycle with the full memory data width. There are three methods implemented, one using bit MCONTROL.DINIT, one using input mtu_auto_data_init and another one using input mtu_partial_erase_i.

MCONTROL.DINIT will write a range with any pattern in RDBFL no matter whether this is a valid ECC code or not. This is useful if memories have additional ECC or parity bits and must be initialized before they can be used. The necessary steps are:

1. Enter memory test mode
2. Initialize registers
RDBFL := <desired pattern>
RANGE := <memory range to be filled>
3. Start fill operation by setting MCONTROL.DINIT and MCONTROL.START of the MCONTROL register
4. Wait for MSTATUS.DONE to be reset and clear MCONTROL.START.
5. Wait for the end of fill operation by polling MSTATUS.DONE bit
6. Leave memory test mode.

Also, two hardware signals named mtu_auto_data_init_i and mtu_partial_erase_i are available to fill the whole memory with the data on d_i or erase parts of the memory. ECC codes are automatically correct. Range is ignored if mtu_auto_data_init_i is applied, in case mtu_partial_erase_i is used nb_partial_erase_size_g blocks of 1kB starting from the top of the memory are erased. Memory testmode does not need to be switched on but a clock must be available.

Both signals will have priority over MCONTROL.START and are not interruptable.

8.4.4.5 Reading a Single Memory Location

The MBIST/ECC can also be used to read the contents of a single word. The RDBFL register holds the contents of a complete memory word and thus it is possible to read all memory bits, including ECC or parity bits. The necessary steps are:

1. Enter memory test mode
2. Initialize registers
RANGE := RAEN = 0 (range disabled = single address) & address to be read
CONFIG0 := 1001_H (NUMACCS = 1_H, ACCSTYPE = 01_H (read))
CONFIG1 := 0000_H (linear mode, non inverted pattern)
3. MCONTROL := 4009_H (FAILDMP = 0, direction up, start): Start read operation
4. MCONTROL := 4008_H (clear START)
5. Wait for MSTATUS.DONE to be reset
6. Read RDBFL register
7. Leave memory test mode

8.4.4.6 Writing to a Single Memory Location

The MBIST/ECC can also be used to write the contents of RDBFL register to a single memory location. RDBFL holds the contents of a complete memory word and thus it is possible to write to all memory bits, including ECC or parity bits. The necessary steps are:

1. Enter memory test mode
2. Initialize registers
 - RDBFL := write data
 - RANGE := RAEN= 0 (range disabled = single address) & address to be written to
 - CONFIG0 := 1000_H (NUMACCS = 1_H, ACCSTYPE = 00_H (write))
 - CONFIG1 := 0000_H (linear mode, non inverted pattern)
3. MCONTROL := 4009_H (FAILDMP = 0, direction up, start): Start write operation
4. MCONTROL := 4008_H (clear START)
5. Wait for MSTATUS.DONE to be reset
6. Leave memory test mode

8.4.5 Memory Controller Register Addresses

There is one set of Memory Controller Registers for each logical memory module in the TC26x B-Step.

The sets of registers are located at offsets from the following base addresses.

Table 8-2 Registers Address Space - Memory Controller Registers

Module	Base Address	End Address	Note
MC0	F006 1000 _H	F006 10FF _H	-
MC1	F006 1100 _H	F006 11FF _H	-
MC2	F006 1200 _H	F006 12FF _H	-
MC3	F006 1300 _H	F006 13FF _H	-
MC4	F006 1400 _H	F006 14FF _H	-
MC5	F006 1500 _H	F006 15FF _H	-
MC6	F006 1600 _H	F006 16FF _H	-
MC7	F006 1700 _H	F006 17FF _H	-
MC8	F006 1800 _H	F006 18FF _H	-
MC9	F006 1900 _H	F006 19FF _H	-
MC10	F006 1A00 _H	F006 1AFF _H	-
MC11	F006 1B00 _H	F006 1BFF _H	-
MC12	F006 1C00 _H	F006 1CFF _H	-
MC13	F006 1D00 _H	F006 1DFF _H	-
MC14	F006 1E00 _H	F006 1EFF _H	-
MC15	F006 1F00 _H	F006 1FFF _H	-
MC16	F006 2000 _H	F006 20FF _H	-
MC17	F006 2100 _H	F006 21FF _H	-
MC18	F006 2200 _H	F006 22FF _H	-
MC19	F006 2300 _H	F006 23FF _H	-
MC20	F006 2400 _H	F006 24FF _H	-
MC21	F006 2500 _H	F006 25FF _H	-
MC22	F006 2600 _H	F006 26FF _H	-
MC23	F006 2700 _H	F006 27FF _H	-
MC24	F006 2800 _H	F006 28FF _H	-
MC25	F006 2900 _H	F006 29FF _H	-

Memory Test Unit (MTU)
Table 8-2 Registers Address Space - Memory Controller Registers (cont'd)

Module	Base Address	End Address	Note
MC26	F006 2A00 _H	F006 2AFF _H	-
MC27	F006 2B00 _H	F006 2BFF _H	-
MC28	F006 2C00 _H	F006 2CFF _H	-
MC29	F006 2D00 _H	F006 2DFF _H	-
MC30	F006 2E00 _H	F006 2EFF _H	-
MC31	F006 2F00 _H	F006 2FFF _H	-
MC32	F006 3000 _H	F006 30FF _H	-
MC33	F006 3100 _H	F006 31FF _H	-
MC34	F006 3200 _H	F006 32FF _H	-
MC35	F006 3300 _H	F006 33FF _H	-
MC36	F006 3400 _H	F006 34FF _H	-
MC37	F006 3500 _H	F006 35FF _H	-
MC38	F006 3600 _H	F006 36FF _H	-
MC39	F006 3700 _H	F006 37FF _H	-
MC40	F006 3800 _H	F006 38FF _H	-
MC41	F006 3900 _H	F006 39FF _H	-
MC42	F006 3A00 _H	F006 3AFF _H	-
MC43	F006 3B00 _H	F006 3BFF _H	-
MC44	F006 3C00 _H	F006 3CFF _H	-
MC45	F006 3D00 _H	F006 3DFF _H	-
MC46	F006 3E00 _H	F006 3EFF _H	-
MC47	F006 3F00 _H	F006 3FFF _H	-
MC48	F006 4000 _H	F006 40FF _H	-
MC49	F006 4100 _H	F006 41FF _H	-
MC50	F006 4200 _H	F006 42FF _H	-
MC51	F006 4300 _H	F006 43FF _H	-
MC52	F006 4400 _H	F006 44FF _H	-
MC53	F006 4500 _H	F006 45FF _H	-
MC54	F006 4600 _H	F006 46FF _H	-
MC55	F006 4700 _H	F006 47FF _H	-

Memory Test Unit (MTU)
Table 8-2 Registers Address Space - Memory Controller Registers (cont'd)

Module	Base Address	End Address	Note
MC56	F006 4800 _H	F006 48FF _H	-
MC57	F006 4900 _H	F006 49FF _H	-
MC58	F006 4A00 _H	F006 4AFF _H	-
MC59	F006 4B00 _H	F006 4BFF _H	-
MC60	F006 4C00 _H	F006 4CFF _H	-
MC61	F006 4D00 _H	F006 4DFF _H	-
MC62	F006 4E00 _H	F006 4EFF _H	-
MC63	F006 4F00 _H	F006 4FFF _H	-
MC64	F006 5000 _H	F006 50FF _H	-
MC65	F006 5100 _H	F006 51FF _H	-
MC66	F006 5200 _H	F006 52FF _H	-
MC67	F006 5300 _H	F006 53FF _H	-
MC68	F006 5400 _H	F006 54FF _H	-
MC69	F006 5500 _H	F006 55FF _H	-
MC70	F006 5600 _H	F006 56FF _H	-
MC71	F006 5700 _H	F006 57FF _H	-
MC72	F006 5800 _H	F006 58FF _H	-
MC73	F006 5900 _H	F006 59FF _H	-
MC74	F006 5A00 _H	F006 5AFF _H	-
MC75	F006 5B00 _H	F006 5BFF _H	-
MC76	F006 5C00 _H	F006 5CFF _H	-
MC77	F006 5D00 _H	F006 5DFF _H	-
MC78	F006 5E00 _H	F006 5EFF _H	-
MC79	F006 5F00 _H	F006 5FFF _H	-
MC80	F006 6000 _H	F006 60FF _H	-
MC81	F006 6100 _H	F006 61FF _H	-
MC82	F006 6200 _H	F006 62FF _H	-
MC83	F006 6300 _H	F006 63FF _H	-
MC84	F006 6400 _H	F006 64FF _H	-
MC85	F006 6500 _H	F006 65FF _H	-

Memory Test Unit (MTU)
Table 8-2 Registers Address Space - Memory Controller Registers (cont'd)

Module	Base Address	End Address	Note
MC86	F006 6600 _H	F006 66FF _H	-
MC87	F006 6700 _H	F006 67FF _H	-

8.4.6 Memory Controller Register Overview

This section describes registers in each memory controller module. Register names described in this section will be referenced in other parts of the TC26x B-Step User's Manual by the prefix "MCx_" where x is the index or name of the specific memory controller. The registers are replicated for each memory controller.

MTU Memory Control Kernel Register Overview
Table 8-3 Register Overview of each MTU Memory Control register block

Short Name	Long Name	Offset Addr. ¹⁾	Access Mode		Re set	Description See
			Read	Write		
CONFIG0	Memory Config Register 0	00 _H	U, SV, 16	U, SV, 16	-	Page 8-18
CONFIG1	Memory Config Register 1	02 _H	U, SV, 16	U, SV, 16	-	Page 8-19
MCONTROL	Memory Control Register	04 _H	U, SV, 16	SV, SE, 16	-	Page 8-21
MSTATUS	Memory Status Register	06 _H	U, SV, 16	BE	-	Page 8-25
RANGE	Memory Range Register	08 _H	U, SV, 16	U, SV, 16	-	Page 8-27
-	Reserved	0A _H	BE	BE	-	-
REVID	Revision Register	0C _H	U, SV, 16	BE	-	Page 8-28
ECCS	Range Register	0E _H	U, SV, 16	SV, SE, 16	-	Page 8-29
ECCD	ECC Detection Register	10 _H	U, SV, 16	SV, 16	-	Page 8-31
ETRRI	Error Tracking Register I (I= 0-4)	12 _H + (I * 02 _H)	U, SV, 16	BE	-	Page 8-34

Memory Test Unit (MTU)

Table 8-3 Register Overview of each MTU Memory Control register block

Short Name	Long Name	Offset Addr. ¹⁾	Access Mode		Re set	Description See
			Read	Write		
–	Reserved	$(14_H + (I * 02_H)) - 1E_H$	BE	BE	–	–
–	Reserved	$20_H - 9F_H$	BE	BE	–	–
RDBFLy	Memory Read Data Register y (y = 0-39)	$(y * 02_H) + A0_H$	U, SV, 16	U, SV, 16	–	Page 8-35
–	Reserved	$F0_H - FF_H$	BE	BE	–	–

1) The absolute register address is calculated as follows:
Memory Module Register Base Address + Offset Address (shown in this column)

8.5 ECC Implementation

8.5.1 ECC

8.5.1.1 ECC Codes

Most ECC codes except for the cache tag memories are of the SECDED (single error correction and double error detection) type. These are specially designed Hsiao codes optimized for triple error detection. They will detect all correctable errors and allow correction, detect all double errors and detect a relatively high percentage of triple errors. The cache tag memories will get DED codes (double error detection).

All codes will detect “all zeros” and “all ones”. However, with the safety SRAM concept of address error detection this feature is not required anymore as all physical faults that might produce an “all ones” or “all zeros” error are covered there.

SECDED Codes

Memory Test Unit (MTU)
14 bits total length, 8 bits data, 6 check bits

Matrix: [1, 2, 4, 8, 16, 32, 7, 11, 19, 35, 13, 22, 44, 52]

Check part of the matrix as bits:

00010011

00100101

01001010

10001111

11110100

11111000

No double columns, can be used as SEC code.

Type: Hsia-Code (all columns have odd weight)

Table 8-4 Code Characteristics 8/6

Number of codes with weight 3	0
Number of codes with weight 4	34
Can be used as SECDED code	
Undetectable double errors (correction on)	0
Undetectable triple errors (correction on)	136
Percentage of undetectable triple errors (correction on)	37.362637
Row weights	[6, 6, 6, 4, 4, 4]
Number of ones in the matrix	30

22 bits total length, 16 bits data, 6 check bits

Matrix: [1, 2, 4, 8, 16, 32, 7, 11, 19, 35, 13, 21, 37, 25, 41, 14, 22, 38, 26, 50, 44, 52]

Check part of the matrix as bits:

0001001010010111

0010010100101101

0100100111001010

1000111001110011

1111000001111100

1111111110000000

No double columns, can be used as SEC code.
 Type: Hsia-Code (all columns have odd weight)

Table 8-5 Code Characteristics 16/6

Number of codes with weight 3	0
Number of codes with weight 4	252
Undetectable double errors (correction on)	0
Undetectable triple errors (correction on)	1008
Percentage of undetectable triple errors (correction on)	64.454544
Row weights	[10, 10, 10, 8, 8, 8]
Number of ones in the matrix	54

28 bits total length, 21 bits data, 7 check bits

Matrix: [1, 2, 4, 8, 16, 32, 64, 76, 13, 35, 52, 69, 88, 28, 49, 38, 19, 73, 84, 11, 56, 50, 22, 97, 14, 67, 98, 104]

Check part of the matrix as bits:

```

100011000011000010111
001100011000011010011
000101110101011100000
110001100010110001001
110110101001000101000
001000001100101101110
011010010110100010100
  
```

No double columns, can be used as SEC code.
 Type: Hsia-Code (all columns have odd weight)

Table 8-6 Code Characteristics 21/7

Number of codes with weight 3	0
Number of codes with weight 4	1316
Undetectable double errors (correction on)	0
Undetectable triple errors (correction on)	1316

Memory Test Unit (MTU)
Table 8-6 Code Characteristics 21/7

Percentage of undetectable triple errors (correction on)	40.17094
Row weights	[10, 10, 10, 10, 10, 10, 10]
Number of ones in the matrix	70

31 bits total length, 24 bits data, 7 check bits

Maxtrix: [1, 2, 4, 8, 16, 32, 64, 15, 19, 97, 111, 41, 35, 98, 25, 104, 78, 37, 100, 112, 21, 49, 50, 84, 67, 69, 52, 73, 56, 82, 88]

Check part of the matrix as bits:

```
001100101101100011101011
001111101011101100010100
010000010000111110010111
10011001110000000001101
100100000111010010110000
110101100100000101000010
111111010010011001101000
```

No double columns, can be used as SEC code.

Type: No Hsia-Code (at least one row has even weight)

Table 8-7 Code Characteristics 24/7

Number of codes with weight 3	0
Number of codes with weight 4	442
Can be used as SECDED code	
Undetectable double errors (correction on)	0
Undetectable triple errors (correction on)	1768
Percentage of undetectable triple errors (correction on)	39.332592
Row weights	[14, 10, 10, 10, 10, 12, 14, 14]
Number of ones in the matrix	84

Memory Test Unit (MTU)
36 bits total length, 29 bits data, 7 check bits

Matrix: [1, 2, 4, 8, 16, 32, 64, 7, 11, 19, 35, 67, 13, 21, 37, 69, 25, 41, 73, 49, 14, 22, 38, 70, 26, 42, 82, 98, 28, 76, 52, 84, 56, 88, 104, 112]

Check part of the matrix as bits:

```
00001000100100001001101010111
00010001001010010010100101011
00100010010010100101010111101
01000100011101000110011001110
10000111100001111000011110000
11111000000001111111100000000
11111111111111000000000000000
```

No double columns, can be used as SEC code.

Type: Hsia-Code (all columns have odd weight)

Table 8-8 Code Characteristics 29/7

Number of codes with weight 3	0
Number of codes with weight 4	979
Undetectable double errors (correction on)	0
Undetectable triple errors (correction on)	3916
Percentage of undetectable triple errors (correction on)	54.84594
Row weights	[14, 14, 14, 14, 14, 12, 12]
Number of ones in the matrix	94

39 bits total length, 32 bits data, 7 check bits

NON-INVERTIBLE code!

Matrix: [1, 2, 4, 8, 16, 32, 64, 11, 35, 67, 13, 21, 37, 69, 25, 41, 73, 49, 81, 97, 14, 22, 38, 70, 26, 42, 74, 50, 82, 98, 28, 44, 76, 52, 84, 100, 56, 88, 112]

Check part of the matrix as bits:

```
00100010010110001001011001011011
01000100101010010010101010101101
00001001001100100100110100110111
10010001110001000111000111000110
```

Memory Test Unit (MTU)

```
00011110000001111000000111111000
11100000000001111111111000000000
11111111111110000000000000000000
```

No double columns, can be used as SEC code.
Type: Hsia-Code (all columns have odd weight)

Table 8-9 Code Characteristics 32/7

Number of codes with weight 3	0
Number of codes with weight 4	1363
Can be used as SECDED code	
Undetectable double errors (correction on)	0
Undetectable triple errors (correction on)	5452
Percentage of undetectable triple errors (correction on)	59.656418
Row weights	[14, 14, 15, 15, 15, 15, 15]
Number of ones in the matrix	103

40 bits total length, 32 bits data, 8 check bits

Matrix: [1, 2, 4, 8, 16, 32, 64, 128, 161, 98, 35, 194, 14, 11, 38, 41, 69, 134, 25, 52, 176, 138, 196, 88, 19, 145, 67, 74, 97, 26, 164, 73, 162, 152, 148, 13, 44, 21, 84, 76]

Check part of the matrix as bits:

```
10010000010011100100001011100000
01010000100000110011100100000011
11100011000110000000101010001000
00000000001110011100010001100110
00001101001001010001010101011001
00001010110100100000001000111111
01111110010001001011010010000000
10100101101000001110100100010100
```

No double columns, can be used as SEC code.
Type: Hsia-Code (all columns have odd weight)

Table 8-10 Code Characteristics 32/8

Number of codes with weight 3	0
Number of codes with weight 4	766
Undetectable double errors (correction on)	0
Undetectable triple errors (correction on)	3064
Percentage of undetectable triple errors (correction on)	31.012146
Row weights	[14, 14, 14, 14, 12, 12, 12, 12]
Number of ones in the matrix	104

42 bits total length, 35 bits data, 7 check bits

Matrix: [1, 2, 4, 8, 16, 32, 64, 7, 11, 19, 35, 67, 13, 21, 37, 69, 25, 41, 73, 49, 81, 97, 14, 22, 38, 70, 26, 42, 74, 50, 82, 98, 28, 44, 76, 52, 84, 100, 56, 88, 104, 112]

Check part of the matrix as bits:

```
00001000100101100010010110010110111
000100010010101001001010101011011
00100010010011001001001101001101101
01000100011100010001110001110001110
10000111100000011110000001111110000
11111000000000011111111110000000000
11111111111111100000000000000000000
```

No double columns, can be used as SEC code.

Type: Hsia-Code (all columns have odd weight)

Table 8-11 Code Characteristics 35/7

Number of codes with weight 3	0
Number of codes with weight 4	1855
Can be used as SECDED code	
Undetectable double errors (correction on)	0
Undetectable triple errors (correction on)	7420
Percentage of undetectable triple errors (correction on)	64.63415

Table 8-11 Code Characteristics 35/7

Row weights	[16, 16, 16, 16, 16, 16, 16]
Number of ones in the matrix	112

44 bits total length, 36 bits data, 8 check bits

Matrix: [1, 2, 4, 8, 16, 32, 64, 128, 21, 28, 74, 26, 7, 70, 140, 112, 138, 11, 196, 161, 22, 41, 193, 152, 100, 35, 56, 137, 200, 37, 131, 168, 50, 73, 82, 176, 81, 14, 52, 164, 84, 224, 42, 97]

Check part of the matrix as bits:

```
000000101011001100011011000100010100
001001010010001010001000011010001101
000000010001010011100101100100110111
110100010000100100100000101110101000
011100101100010100111001010001000010
110011100010100010000100000001111000
001111001100100001000010101001000010
100010000101011001010110010010000001
```

No double columns, can be used as SEC code.

Type: Hsiao-Code (all columns have odd weight)

Table 8-12 Code Characteristics 36/8

Number of codes with weight 3	0
Number of codes with weight 4	1143
Can be used as SECDED code	
Undetectable double errors (correction on)	0
Undetectable triple errors (correction on)	4572
Percentage of undetectable triple errors (correction on)	34.521294
Row weights	[14, 14, 14, 16, 14, 16, 14, 14]
Number of ones in the matrix	116

Memory Test Unit (MTU)
72 bits total length, 64 bits data, 8 check bits

Matrix: [1, 2, 4, 8, 16, 32, 64, 128, 7, 11, 19, 35, 67, 131, 13, 21, 37, 69, 133, 25, 41, 73, 137, 49, 81, 145, 97, 161, 193, 14, 22, 38, 70, 134, 26, 42, 74, 138, 50, 82, 146, 98, 162, 194, 28, 44, 76, 140, 52, 84, 148, 100, 164, 196, 56, 88, 152, 104, 168, 200, 112, 176, 208, 224, 227, 233, 143, 62, 93, 47, 181, 124]

Check part of the matrix as bits:

```
00000100001000100101011000010001001011000100101100101101111100010
000010000100010010101010001000100101010010010101010101101111001001
0001000010001001001100010001001001100100100110100110110111010111
0010000100010001110000100010001110001000111000111000111000011011
0100001000011110000001000011110000001111000000111111000001111101
100000111110000000000111110000000000111111111000000000000111111
111111000000000000000111111111111110000000000000000000010110100
11111111111111111111000000000000000000000000000000000000011101110
```

No double columns, can be used as SEC code.

Type: Hsiao-Code (all columns have odd weight)

Table 8-13 Code Characteristics 64/8

Number of codes with weight 3	0
Number of codes with weight 4	8414
Undetectable double errors (correction on)	0
Undetectable triple errors (correction on)	33656
Percentage of undetectable triple errors (correction on)	56.431927
Row weights	[28, 26, 28, 28, 26, 28, 26, 26]
Number of ones in the matrix	216

137 bits total length, 128 bits data, 9 check bits

Matrix: [1, 2, 4, 8, 16, 32, 64, 128, 256, 19, 21, 25, 37, 38, 44, 49, 52, 67, 69, 73, 81, 82, 84, 88, 97, 100, 112, 131, 133, 137, 145, 146, 148, 152, 161, 164, 176, 193, 194, 196, 200, 208, 224, 448, 385, 265, 266, 400, 321, 268, 273, 296, 292, 280, 328, 336, 392, 15, 30, 43, 58, 78, 106, 142, 170, 306, 354, 263, 278, 326, 291, 418, 390, 55, 61, 103, 117, 118, 124, 167, 173, 181, 182, 188, 211, 213, 217, 230, 236, 109, 345, 229, 346, 241, 244, 409, 425, 410, 484, 457, 364, 428, 412, 458, 361, 460, 421, 357, 301, 283, 285,

DED Codes

2-Bit-error detection, but no correction possible!

25 bits total length, 20 bits data, 5 check bits

Matrix: [1, 2, 4, 8, 16, 3, 5, 9, 17, 6, 10, 18, 12, 20, 24, 7, 11, 19, 13, 21, 14, 26, 28, 15, 31]

Check part of the matrix as bits:

```
00010010110010101101
0010010101010101011111
01001001101001110111
10001110001110011011
11110000001111100011
```

No double columns, can be used as SEC code.

Type: No Hsiao code (at least one row has even weight)

Table 8-15 Code Characteristics 20/5

Number of codes with weight 3	80
Number of codes with weight 4	435
Can be used as DED code only	
Percentage of undetectable triple errors	3.4782608
Row weights	[12, 12, 12, 12, 10]
Number of ones in the matrix	58

26 bits total length, 21 bits data, 5 check bits

Matrix: [1, 2, 4, 8, 16, 3, 5, 9, 17, 6, 10, 18, 12, 20, 24, 7, 11, 19, 13, 21, 25, 14, 22, 26, 28, 31]

Check part of the matrix as bits:

```
000100101100101101111
001001010101010101111
010010011010011011011
100011100011100011101
111100000011111100001
```

Memory Test Unit (MTU)

No double columns, can be used as SEC code.

Type: No Hsiao code (at least one row has even weight)

Table 8-16 Code Characteristics 21/5

Number of codes with weight 3	90
Number of codes with weight 4	515
Can be used as DED code only	
Percentage of undetectable triple errors	3.4615386
Row weights	[12, 12, 12, 12, 12]
Number of ones in the matrix	60

27 bits total length, 22 bits data, 5 check bits

Matrix: [1, 2, 4, 8, 16, 3, 5, 9, 17, 6, 10, 18, 12, 20, 24, 7, 11, 19, 13, 21, 25, 14, 22, 26, 15, 23, 27]

Check part of the matrix as bits:

```
0001001011001011011011
00100101010101010110101
0100100110100110110110
1000111000111000111111
1111000000111111000111
```

No double columns, can be used as SEC code.

Type: No Hsiao code (at least one row has even weight)

Table 8-17 Code Characteristics 22/5

Number of codes with weight 3	101
Number of codes with weight 4	606
Can be used as DED code only	
Percentage of undetectable triple errors	3.4529915
Row weights	[14, 14, 12, 12, 12]
Number of ones in the matrix	64

Memory Test Unit (MTU)
30 bits total length, 24 bits data, 6 check bits

Matrix: [1, 2, 4, 8, 16, 32, 3, 5, 9, 17, 33, 6, 10, 18, 34, 12, 20, 36, 24, 40, 7, 11, 19, 35, 13, 49, 22, 42, 52, 56]

Check part of the matrix as bits:

```
000010001001010001010111
000100010010100010011011
001000100100110100100101
010001000111001000101010
100001111000001111001100
111110000000001111110000
```

No double columns, can be used as SEC code.

Type: No Hsiao code (at least one row has even weight)

Table 8-18 Code Characteristics 24/6

Number of codes with weight 3	75
Number of codes with weight 4	458
Can be used as DED code only	
Percentage of undetectable triple errors	1.8472906
Row weights	[12, 12, 10, 10, 10, 10]
Number of ones in the matrix	64

71 bits total length, 64 bits data, 7 check bits

Matrix: [1, 2, 4, 8, 16, 32, 64, 3, 5, 9, 17, 33, 65, 6, 10, 18, 34, 66, 12, 20, 36, 68, 24, 40, 72, 48, 80, 96, 7, 11, 19, 35, 67, 13, 21, 37, 69, 25, 41, 73, 49, 81, 97, 14, 22, 38, 70, 26, 42, 74, 50, 82, 98, 28, 44, 76, 52, 84, 100, 56, 88, 104, 112, 15, 23, 43, 51, 101, 89, 102, 90]

Check part of the matrix as bits:

```
0000010000100010010110000100010010110001001011001011011100001111
0000100001000100101010001000100101010010010101010101101100111010
0001000010001001001100010001001001100100100110100110110101010101
0010000100010001110000100010001110001000111000111000111010100101
0100001000011110000001000011110000001111000000111111000011001010
```

Memory Test Unit (MTU)

100001111100000000001111100000000001111111111000000000011110011
 1111110000000000000001111111111111100000000000000000000011111100

No double columns, can be used as SEC code.

Type: No Hsiao code (at least one row has even weight)

Table 8-19 Code Characteristics 64/7

Number of codes with weight 3	505
Number of codes with weight 4	7933
Can be used as DED code only	
Percentage of undetectable triple errors	0.88356227
Row weights	[28, 28, 26, 26, 26, 26, 26]
Number of ones in the matrix	186

136 bits total length, 128 bits data, 8 check bits

Matrix: [1, 2, 4, 8, 16, 32, 64, 128, 3, 5, 9, 17, 33, 65, 129, 6, 10, 18, 34, 66, 130, 12, 20, 36, 68, 132, 24, 40, 72, 136, 48, 80, 144, 96, 160, 192, 7, 11, 19, 35, 67, 131, 13, 21, 37, 69, 133, 25, 41, 73, 137, 49, 81, 145, 97, 161, 193, 14, 22, 38, 70, 134, 26, 42, 74, 138, 50, 82, 146, 98, 162, 194, 28, 44, 76, 140, 52, 84, 148, 100, 164, 196, 56, 88, 152, 104, 168, 200, 112, 176, 208, 224, 15, 216, 202, 75, 150, 92, 89, 149, 204, 60, 228, 153, 108, 99, 166, 172, 135, 169, 46, 51, 198, 116, 106, 86, 240, 78, 209, 177, 27, 141, 225, 232, 147, 43, 212, 163, 39, 45, 90, 210, 57, 23, 29, 165]

Check part of the matrix as bits:

```
000000100000100001000100101100000100001000100101100001000100101100010
01011001011011101101001101100111100100010110111101100010001
000001000001000010001001010100001000010001001010100010001001010100100
1010101010110110110101010100000011111100011001000110000
000010000010000100010010011000010000100010010011000100010010011001001
00110100110110100000000011011110111011010010011010111001001
000100000100001000100011100000100001000100011100001000100011100010001
11000111000111001001111010100000001010110111000101000111110
001000001000010000111100000001000010000111100000010000111100000011110
00000111111000011110110110110010110001001001101010001101010
010000010000011111000000000010000011111000000000011111000000000011111
111110000000000010001101111010111010110101000100001011000111
```


8.5.2 Address Error Detection

Address error detection has to capture several memory faults that cause multiple or all bits of a word to fail. Such faults could be address decoder stuck-at-faults, broken word-lines, bridged word-lines etc. Because of this origin an ECC detection would not cover these cases because the words read back from memory would have perfectly matched check bits despite corrupted word selects.

In this version of MBIST/ECC the SRAMs will detect any of the faulty cases.

8.6 Implementation Section

All Memory Controller register regions which are not allocated in the table below are not implemented. Attempted accesses to regions which are not implemented will return a bus error.

8.6.1 Memory Control Register Implementation

8.6.1.1 MEMTEST Implementation

The memory test register MEMTEST contains enable bits for the various Memory Controllers.

Memory control registers described in this chapter are accessible only when the Memory Controller is enabled (MTU_MEMTEST.MEMxEN = 1). The only exceptions to this are registers ECCD and ETRR (for each memory) which are available at all times to permit easy runtime access to ECC features.

Note:

2. *When a Memory Controller is enabled, functional access to the memory is temporarily unavailable. When a memory is being tested it is essential to prevent an attempted functional access to that memory (e.g. While testing a CPU's local memories, the CPU should be in an idle state, or executing from other memory). Attempted functional accesses to memories when MEMxEN=1 may result in unexpected behaviour.*
3. *Correct execution of a Memory Controller operation (e.g. MBIST or initialization sequence) requires that both the MTU and the module containing the memory-under-test are both operational for the duration of the sequence. A module reset of the MTU or the module containing the memory-under-test, during execution of such an MBIST sequence may result in unexpected behaviour.*

MTU_MEMTEST0

Memory MBISTEnable Register 0 (F006 0010_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GTM 1AEN	GTM M1EN	GTM M0EN	GTM FEN	CPU ODS 2EN	FSIO EN	Res	Res	Res	ETH EN	CPU 2DS 2EN	CPU 1DS 2EN	CPU 0DT EN	Res	CPU 0PT EN	CPU 0PS EN
rwh	rwh	rwh	rwh	rwh	rwh	r	r	r	rwh	rwh	rwh	rwh	r	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	CPU ODS EN	MMC DSE N	LM- UEN	CPU 1PT EN	Res	CPU 1PS EN	CPU 1DT EN	Res	CPU 1DS EN	CPU 2PT EN	Res	CPU 2PS EN	CPU 2DT EN	Res	CPU 2DS EN
r	rwh	rwh	rwh	rwh	r	rwh	rwh	r	rwh	rwh	r	rwh	rwh	r	rwh

Memory Test Unit (MTU)

Field	Bits	Type	Description
CPU2XEN	[5:0]	r	Reserved in this product
CPU1DSEN	6	rwh	CPU1 TC1.6P DSPR MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
Res	7	r	Reserved in this product
CPU1DTEN	8	rwh	CPU1 TC1.6P DTAG MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
CPU1PSEN	9	rwh	CPU1 TC1.6P PSPR MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
Res	10	r	Reserved
CPU1PTEN	11	rwh	CPU1 TC1.6P PTAG MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
LMUEN	12	r	Reserved in this product
MMCDSEN	13	r	Reserved in this product
CPU0DSEN	14	rwh	CPU0 DSPR MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
Res	15	r	Reserved in this product
CPU0PSEN	16	rwh	CPU0 PSPR MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
CPU0PTEN	17	rwh	CPU0 PTAG MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled (subject to restrictions)
Res	18	r	Reserved
CPU0DTEN	19	r	Reserved in this product
CPUXDS2EN	[21:20]	r	Reserved in this product

Memory Test Unit (MTU)

Field	Bits	Type	Description
ETHEN	22	rwh	ETHERMAC MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
Res	[25:23]	r	Reserved
FSI0EN	26	rwh	FSI0 MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
CPU0DS2EN	27	r	Reserved in this product
GTMFEN	28	rwh	GTM FIFO0 MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
GTMM0EN	29	rwh	GTM MCS0 MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
GTMM1EN	30	rwh	GTM RAM1 MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
GTM1AEN	31	rwh	GTM RAM1A MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled

MTU_MEMTEST1
Memory MBISTEnable Register 1 (F006 0014_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EME MU1 EN	EME MU0 EN	EME ML1 5EN	EME ML1 4EN	EME ML1 3EN	EME ML1 2EN	EME ML1 1EN	EME ML1 0EN	EME ML9 EN	EME ML8 EN	EME ML7 EN	EME ML6 EN	EME ML5 EN	EME ML4 EN	EME ML3 EN	EME ML2 EN
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EME ML1 EN	EME ML0 EN	MCD SEN	STB Y0E N	ERA Y1M EN	ERA Y1T EN	ERA Y1O EN	ERA Y0M EN	ERA Y0T EN	ERA Y0O EN	MCA N1E N	MCA N0E N	Res	PSI5 EN	GTM 2EN	GTM 1BE N
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	r	rwh	rwh	rwh

Memory Test Unit (MTU)

Field	Bits	Type	Description
GTM1BEN	0	rwh	GTM RAM1B Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
GTM2EN	1	rwh	GTM RAM2 Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
PSI5EN	2	rwh	PSI5 Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
Res	3	r	Reserved
MCAN0EN	4	rwh	MultiCAN0 Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
MCAN1EN	5	r	Reserved in this product
ERAY0OEN	6	rwh	ERAY0 OBF Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
ERAY0TEN	7	rwh	ERAY0 TBF Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
ERAY0MEN	8	rwh	ERAY0 MBF Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
ERAY1XEN	[11:9]	r	Reserved in this product
STBY1EN	12	rwh	8 Bit Standby RAM Controller Memory 1 Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
MCDSEN	13	rwh	MCDS Controller Memory Enable (ED only) 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
EMEMLxEN (x = 0-3)	x+14	rwh	EMEM Lower x (TCM) MBIST Controller Memory Test Enable (ED only) 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled

Memory Test Unit (MTU)

Field	Bits	Type	Description
EMEMLxEN (x = 4-7)	x+14	rwh	EMEM Lower x (TCM) MBIST Controller Memory Test Enable (ED only) 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
EMEMLXEN	[29:22]	r	Reserved in this product
EMEMUXEN	[31:30]	r	Reserved in this product

MTU_MEMTEST2
Memory MBISTEnable Register 2 (F006 0018_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								FFT1 EN	FFT0 EN	XTM 1EN	XTM 0EN	DMA EN	STB Y2E N	CIF2 EN	CIF1 EN
r								rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DAM EN	CIF0 EN	EME MU1 5EN	EME MU1 4EN	EME MU1 3EN	EME MU1 2EN	EME MU1 1EN	EME MU1 0EN	EME MU9 EN	EME MU8 EN	EME MU7 EN	EME MU6 EN	EME MU5 EN	EME MU4 EN	EME MU3 EN	EME MU2 EN
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
EMEMUxEN	[13:0]	r	Reserved in this product
CIF0EN	14	rwh	CIF JPEG1_4 Memory Enable (ED only) 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
DAMEN	15	r	Reserved in this product
CIF1EN	16	rwh	CIF JPEG3 Memory Enable (ADAS Product only) 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
CIF2EN	17	rwh	CIF Memory2 Enable (ADAS Product only) 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled

Memory Test Unit (MTU)

Field	Bits	Type	Description
STBY2EN	18	rwh	8-bit Standby Controller Memory2 Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled
DMAEN	19	rwh	DMA MBIST Controller Memory Enable 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled (subject to restrictions)
XTM0EN	20	rwh	EMEM XTM0 Controller Memory Enable (ED only) 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled (subject to restrictions)
XTM1EN	21	rwh	EMEM XTM1 Controller Memory Enable (ED only) 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled (subject to restrictions)
FFT0EN	22	rwh	FFT0 Memory Controller Memory Enable (ADAS Product only) 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled (subject to restrictions)
FFT1EN	23	rwh	FFT1 Memory Controller Memory Enable (ADAS Product only) 0 _B MBIST controller is disabled 1 _B MBIST controller is enabled (subject to restrictions)
Res	[31:24]	r	Reserved

8.6.1.2 MEMMAP Implementation

The Memory Mapping Enable register MEMMAP has configurable control bits to select memory-mapped test mode for each CPU memory.

Cache and Scratchpad memories are physically implemented as a single RAM, but this register function assumes two separate logical RAM partitions. In this register additional bits CPUxDCMAP and CPUxPCMAP are defined. These control the Cache partitions of the RAMs for Data Side and Program side respectively. Since cache content and tags of a cache must be simultaneously switched from memory mapped to cache functional mode, the control bits are mirrored and only one bit is writeable for each cache. The bits corresponding to the tag memories of the same cache will always take the same value as that written to the main Cache Memory control bit. This linkage is product specific.

MTU_MEMMAP

Memory Mapping Enable Register (F006 001C_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Res												CPU 0DT MAP	CPU 0DC MAP	CPU 0PT MAP	Res	
												r	rh	rwh	rh	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
CPU 0PC MAP	Res			CPU 1PT MAP	CPU 1PC MAP	Res	CPU 1DT MAP	CPU 1DC MAP	Res	CPU 2PT MAP	CPU 2PC MAP	Res	CPU 2DT MAP	CPU 2DC MAP	Res	
rwh	r	r	r	rh	rwh	r	rh	rwh	r	rh	rwh	r	rh	rwh	r	

Field	Bits	Type	Description
Res	0	r	Reserved Not used
CPU2DxMAP	[2:1]	r	Reserved in this product Not used
Res	3	r	Reserved Not used
CPU2PxMAP	[5:4]	r	Reserved in this product Not used

Memory Test Unit (MTU)

Field	Bits	Type	Description
Res	6	r	Reserved Not used
CPU1DCMAP	7	rwh	CPU1 DCACHE Mapping 0 _B Normal cache function 1 _B Memory-mapped
CPU1DTMAP	8	rh	CPU1 DTAG Mapping 0 _B Normal cache function 1 _B Memory-mapped Read only. Mirrors the state of CPU1DCEN CPU1 cache memories may only be mapped simultaneously.
Res	9	r	Reserved Not used
CPU1PCMAP	10	rwh	CPU1 PCACHE Mapping 0 _B Normal cache function 1 _B Memory-mapped
CPU1PTMAP	11	rh	CPU1 PTAG Mapping 0 _B Normal cache function 1 _B Memory-mapped Read only. Mirrors the state of CPU1PCEN (bit 10) CPU1 cache memories may only be mapped simultaneously.
Res	[14:12]	r	Reserved Not used
CPU0PCMAP	15	rwh	CPU0 PCACHE Mapping 0 _B Normal cache function 1 _B Memory-mapped
Res	16	r	Reserved Not used
CPU0PTMAP	17	rh	CPU0 PTAG Mapping 0 _B Normal cache function 1 _B Memory-mapped Read only. Mirrors the state of CPU0PCEN. CPU0 cache memories may only be mapped simultaneously.
CPU0DxMAP	[19:18]	r	Reserved in this product Not used in this product.

Memory Test Unit (MTU)

Field	Bits	Type	Description
Res	[31:20]	r	Reserved Not used

8.6.1.3 MEMSTAT Implementation

The Memory Status registers MEMSTATx have an implemented bit for each security-relevant RAM.

Cache and Scratchpad memories are physically implemented as a single RAM with a single MBIST, so bits CPUxDSAIU and CPUxPSAIU give the status of the partial initialization of the cache partitions for Data Side and Program side respectively. Since cache content and tags of a cache may take different lengths of time to erase, both are implemented as separate status bits.

MTU_MEMSTAT0

Memory Status Register 0
(F006 0038_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res			CPU 0DS 2AIU	FSIO AIU	HSM RAIU	HSM TAIU	HSM CAIU	Res	CPU 2DS 2AIU	CPU 1DS 2AIU	CPU 0DT AIU	Res	CPU 0PT AIU	CPU 0PS AIU	
r			rh	rh	rh	rh	rh	r	rh	rh	rh	r	r	rh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	CPU 0DS AIU	Res		CPU 1PT AIU	Res	CPU 1PS AIU	CPU 1DT AIU	Res	CPU 1DS AIU	CPU 2PT AIU	Res	CPU 2PS AIU	CPU 2DT AIU	Res	CPU 2DS AIU
r	rh	r	r	rh	r	rh	rh	r	rh	rh	r	rh	rh	r	rh

Field	Bits	Type	Description
CPU2DSAIU	0	r	Reserved in this product
CPU2DTAIU	2	r	Reserved in this product
CPU2PSAIU	3	r	Reserved in this product
CPU2PTAIU	5	r	Reserved in this product
CPU1DSAIU	6	rh	CPU1 DCACHE Partial AutoInitialize Underway 0 _B MBIST not running autoinitialize 1 _B MBIST running autoinitialize This bit indicates whether an automatic data initialization has been triggered by a change of state of MEMTEST.MEMxEN or MEMxMAP but that the initialization sequence has not yet completed.

Memory Test Unit (MTU)

Field	Bits	Type	Description
CPU1DTAIU	8	rh	CPU1 DTAG MBIST AutoInitialize Underway 0 _B MBIST not running autoinitialize 1 _B MBIST running autoinitialize This bit indicates whether an automatic data initialization has been triggered by a change of state of MEMTEST.MEMxEN or MEMxMAP but that the initialization sequence has not yet completed.
CPU1PSAIU	9	rh	CPU1 PCACHE Partial AutoInitialize Underway 0 _B MBIST not running autoinitialize 1 _B MBIST running autoinitialize This bit indicates whether an automatic data initialization has been triggered by a change of state of MEMTEST.MEMxEN or MEMxMAP but that the initialization sequence has not yet completed.
CPU1PTAIU	11	rh	CPU1 PTAG MBIST AutoInitialize Underway 0 _B MBIST not running autoinitialize 1 _B MBIST running autoinitialize This bit indicates whether an automatic data initialization has been triggered by a change of state of MEMTEST.MEMxEN or MEMxMAP but that the initialization sequence has not yet completed.
CPU0DSAIU	14	r	Reserved in this product
CPU0PSAIU	16	rh	CPU0 PCACHE Partial AutoInitialize Underway 0 _B MBIST not running autoinitialize 1 _B MBIST running autoinitialize This bit indicates whether an automatic data initialization has been triggered by a change of state of MEMTEST.MEMxEN or MEMxMAP but that the initialization sequence has not yet completed.
CPU0PTAIU	17	rh	CPU0 PTAG MBIST AutoInitialize Underway 0 _B MBIST not running autoinitialize 1 _B MBIST running autoinitialize This bit indicates whether an automatic data initialization has been triggered by a change of state of MEMTEST.MEMxEN or MEMxMAP but that the initialization sequence has not yet completed.
CPU0DxAIU	19	r	Reserved in this product
CPU1DS2AIU	20	r	Reserved in this product

Memory Test Unit (MTU)

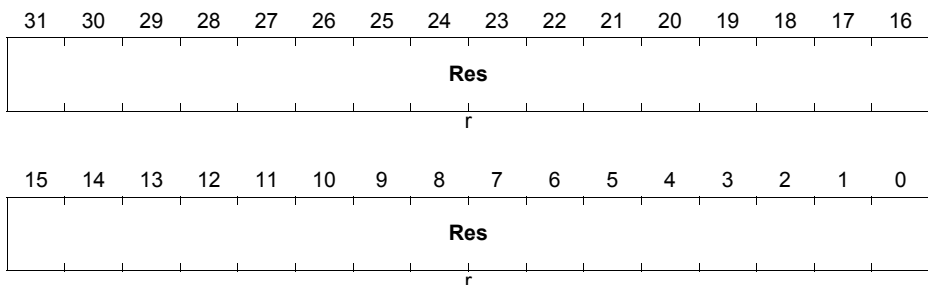
Field	Bits	Type	Description
CPU2DS2AIU	21	r	Reserved in this product
HSMCAIU	23	r	Reserved in this product Not used in this product.
HSMTAIU	24	r	Reserved in this product Not used in this product.
HSMRAIU	25	r	Reserved in this product Not used in this product.
FSI0AIU	26	rh	FSI0 MBIST AutoInitialize Underway 0 _B MBIST not running autoinitialize 1 _B MBIST running autoinitialize This bit indicates whether an automatic data initialization has been triggered by a change of state of MEMTEST.MEMxEN or MEMxMAP but that the initialization sequence has not yet completed.
CPU0DS2AIU	27	r	Reserved in this product
0	31,30,29,28,22,18,15,13,12,10,7,4,1	r	Reserved Not used

MTU_MEMSTAT1

Memory Status Register 1

(F006 003C_H)

Reset Value: 0000 0000_H



Memory Test Unit (MTU)

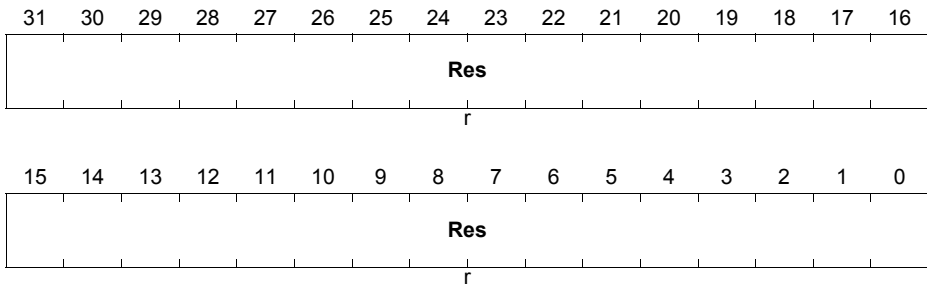
Field	Bits	Type	Description
Res	[31:0]	r	Reserved Not used

MTU_MEMSTAT2

Memory Status Register 2

(F006 0040_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
Res	[31:0]	r	Reserved Not used

8.6.1.4 Memory Controller Instances

Table 8-21 Address Regions - Memory Controller Register

No	Module	Base Address	End Address	Note
.				
0	Reserved in this product	F006 1000 _H	F006 10FF _H	-
1	Reserved	F006 1100 _H	F006 11FF _H	-
2	Reserved in this product	F006 1200 _H	F006 12FF _H	-
3	Reserved in this product	F006 1300 _H	F006 13FF _H	-
4	Reserved	F006 1400 _H	F006 14FF _H	-
5	Reserved in this product	F006 1500 _H	F006 15FF _H	-
6	MC_CPU1 TC16P_DSPP	F006 1600 _H	F006 16FF _H	-

Memory Test Unit (MTU)
Table 8-21 Address Regions - Memory Controller Register

No	Module	Base Address	End Address	Note
7	Reserved	F006 1700 _H	F006 17FF _H	-
8	MC_CPU1 TC16P_DTAG	F006 1800 _H	F006 18FF _H	-
9	MC_CPU1 TC16P_PSPR0	F006 1900 _H	F006 19FF _H	-
10	Reserved	F006 1A00 _H	F006 1AFF _H	-
11	MC_CPU1 TC16P_PTAG	F006 1B00 _H	F006 1BFF _H	-
12	Reserved	F006 1C00 _H	F006 1CFF _H	-
13	Reserved in this product	F006 1D00 _H	F006 1DFF _H	-
14	MC_CPU0_DSPR	F006 1E00 _H	F006 1EFF _H	-
15	Reserved	F006 1F00 _H	F006 1FFF _H	-
16	MC_CPU0_PSPR	F006 2000 _H	F006 20FF _H	-
17	MC_CPU0_PTAG	F006 2100 _H	F006 21FF _H	-
18	Reserved	F006 2200 _H	F006 22FF _H	-
19	Reserved in this product	F006 2300 _H	F006 23FF _H	-
20	Reserved in this product	F006 2400 _H	F006 24FF _H	-
21	Reserved in this product	F006 2500 _H	F006 25FF _H	-
22	MC_ETHERMAC	F006 2600 _H	F006 26FF _H	-
23	Reserved in this product	F006 2700 _H	F006 27FF _H	-
24	Reserved in this product	F006 2800 _H	F006 28FF _H	-
25	Reserved in this product	F006 2900 _H	F006 29FF _H	-
26	MC_FSI0	F006 2A00 _H	F006 2AFF _H	-
27	Reserved	F006 2B00 _H	F006 2BFF _H	-
28	MC_GTM FIFO0	F006 2C00 _H	F006 2CFF _H	-
29	MC_GTM MCS0	F006 2D00 _H	F006 2DFF _H	-
30	MC_GTM MCS1	F006 2E00 _H	F006 2EFF _H	-
31	MC_GTM DPLL RAM1A	F006 2F00 _H	F006 2FFF _H	-
32	MC_GTM DPLL RAM1B	F006 3000 _H	F006 30FF _H	-
33	MC_GTM DPLL RAM2	F006 3100 _H	F006 31FF _H	-
34	MC_PSI5	F006 3200 _H	F006 32FF _H	-
35	Reserved	F006 3300 _H	F006 33FF _H	-
36	MC_MCAN0	F006 3400 _H	F006 34FF _H	-

Memory Test Unit (MTU)
Table 8-21 Address Regions - Memory Controller Register

No	Module	Base Address	End Address	Note
37	Reserved in this product	F006 3500 _H	F006 35FF _H	-
38	MC_ERAY0 OBF	F006 3600 _H	F006 36FF _H	-
39	MC_ERAY0 IBF_TBF	F006 3700 _H	F006 37FF _H	-
40	MC_ERAY0 MBF	F006 3800 _H	F006 38FF _H	-
41	Reserved in this product	F006 3900 _H	F006 39FF _H	-
42	Reserved in this product	F006 3A00 _H	F006 3AFF _H	-
43	Reserved in this product	F006 3B00 _H	F006 3BFF _H	-
44	MC_STBY1	F006 3C00 _H	F006 3CFF _H	-
45	MC_MCDS (ED only)	F006 3D00 _H	F006 3DFF _H	-
46	MC_EMEM LOWER 0 (ED only)	F006 3E00 _H	F006 3EFF _H	-
47	MC_EMEM LOWER 1 (ED only)	F006 3F00 _H	F006 3FFF _H	-
48	MC_EMEM LOWER 2 (ED only)	F006 4000 _H	F006 40FF _H	-
49	MC_EMEM LOWER 3 (ED only)	F006 4100 _H	F006 41FF _H	-
50	MC_EMEM LOWER 4 (ED only)	F006 4200 _H	F006 42FF _H	-
51	MC_EMEM LOWER 5 (ED only)	F006 4300 _H	F006 43FF _H	-
52	MC_EMEM LOWER 6 (ED only)	F006 4400 _H	F006 44FF _H	-
53	MC_EMEM LOWER 7 (ED only)	F006 4500 _H	F006 45FF _H	-
-	Reserved in this product	F006 4600 _H	F006 5DFF _H	-
78	MC_CIF0 (ADAS Product only)	F006 5E00 _H	F006 5EFF _H	-
79	Reserved in this product	F006 5F00 _H	F006 5FFF _H	-
80	MC_CIF1 (ADAS Product only)	F006 6000 _H	F006 60FF _H	-
81	MC_CIF2 (ADAS Product only)	F006 6100 _H	F006 61FF _H	-
82	MC_STBY2	F006 6200 _H	F006 62FF _H	-
83	MC_DMA	F006 6300 _H	F006 63FF _H	-
84	MC_EMEM XTM0 (ED only)	F006 6400 _H	F006 64FF _H	-
85	MC_EMEM XTM1 (ED only)	F006 6500 _H	F006 65FF _H	-
86	MC_FFT0 (ADAS Product only)	F006 6600 _H	F006 66FF _H	-

Memory Test Unit (MTU)
Table 8-21 Address Regions - Memory Controller Register

No	Module	Base Address	End Address	Note
87	MC_FFT1 (ADAS Product only)	F006 6700 _H	F006 67FF _H	-
	Reserved.			All other unassigned addresses up to F006 FFFF _H

The system SRAMs do not all have the same configuration. [Chapter 8-22](#) shows the instance-specific configurations of the MBIST modules.

The ECC values for all SRAMs are computed only out of the data information*.

Table 8-22 Configurations - Memory Controllers

No	Module	Error Addr Buffer Depth	ECC type	ECC granularity	Mux Factor
6	MC_CPU1 TC16P_DSPR	5	SECDED	2	16
8	MC_CPU1 TC16P_DTAG	2	SECDED	2	4
9	MC_CPU1 TC16P_PSPR	5	SECDED	1	8
11	MC_CPU1 TC16P_PTAG	2	DED	2	4
14	MC_CPU0_TC16E_DSPR	5	SECDED	2	16
16	MC_CPU0_TC16E_PSPR	3	SECDED	1	8
17	MC_CPU0_TC16E_PTAG	2	DED	2	4
20	MC_FFT0	5	SECDED	1	4
21	MC_FFT1	1	SECDED	1	4
22	MC_ETHERMAC	1	SECDED	1	4
26	MC_FSI0	1	SECDED	1	8
28	MC_GTM FIFO0	1	SECDED	1	4
29	MC_GTM MCS0	1	SECDED	1	8
30	MC_GTM MCS1	1	SECDED	1	8
31	MC_GTM DPLL RAM1A	1	SECDED	1	4
32	MC_GTM DPLL RAM1B	1	SECDED	1	4

Memory Test Unit (MTU)
Table 8-22 Configurations - Memory Controllers

No	Module	Error Addr Buffer Depth	ECC type	ECC granularity	Mux Factor
33	MC_GTM DPLL RAM2	1	SECDED	1	8
34	MC_PSI5	1	SECDED	1	4
36	MC_MCAN0	1	SECDED	1	4
38	MC_ERAY0 OBF	1	SECDED	1	4
39	MC_ERAY0 IBF_TBF	1	SECDED	1	4
40	MC_ERAY0 MBF	1	SECDED	1	4
44	MC_STBY1	1	SECDED	1	8
45	MC_MCDS (ED only)	1	DED	1	4
46	MC_EMEM LOWER 0 (ED only)	1	SECDED	1	4
47	MC_EMEM LOWER 1 (ED only)	1	SECDED	1	4
48	MC_EMEM LOWER 2 (ED only)	1	SECDED	1	4
49	MC_EMEM LOWER 3 (ED only)	1	SECDED	1	4
50	MC_EMEM LOWER 4 (ED only)	1	SECDED	1	4
51	MC_EMEM LOWER 5 (ED only)	1	SECDED	1	4
52	MC_EMEM LOWER 6 (ED only)	1	SECDED	1	4
53	MC_EMEM LOWER 7 (ED only)	1	SECDED	1	4
78	MC_CIF0 (ADAS Product only)	1	SECDED	1	16
80	MC_CIF1 (ADAS Product only)	1	SECDED	1	4
81	MC_CIF2 (ADAS Product only)	1	SECDED	1	4
82	MC_STBY2	1	SECDED	1	4
83	MC_DMA	1	SECDED	1	4
84	MC_EMEM XTM0 (ED only)	1	SECDED	1	4
85	MC_EMEM XTM1 (ED only)	1	SECDED	1	4
86	MC_FFT0 (ADAS Product only)	1	SECDED	1	-
87	MC_FFT1 (ADAS Product only)	1	SECDED	1	-

9 Safety Management Unit (SMU)

9.1 Introduction

The SMU is a central and modular component of the safety architecture providing a generic interface to manage the behavior of the microcontroller under the presence of faults. The SMU centralizes all the alarm signals related to the different hardware and software-based safety mechanisms. Each alarm can be individually configured to trigger internal actions and/or notify externally the presence of faults via a fault signaling protocol. The default fault signaling protocol is a bi-stable Error Pin. The severity of each alarm shall be configured according to the needs of the safety application(s): per default every alarm is disabled with the exception of the watchdog timeout alarms. For debug and diagnosis purposes the alarm events are logged, the logged information is resilient to any system or application reset. The SMU also implements some housekeeping functions related to the management and test of dedicated safety mechanisms. A special test mode is available to test the SMU itself enabling to detect latent faults as required by the ISO 26262 safety standard. In addition to the generic register access protection, the SMU implements an independent configuration locking mechanism.

The SMU in combination with the embedded safety mechanisms enable to detect and report more than 99% of the critical failure modes of the microcontroller within the fault tolerance time interval. The timing characteristics of the fault tolerance time interval can be configured in the SMU.

Figure 9-1 “SMU Interfaces” on Page 9-2 gives an overview of the SMU interfaces.

Safety Management Unit (SMU)

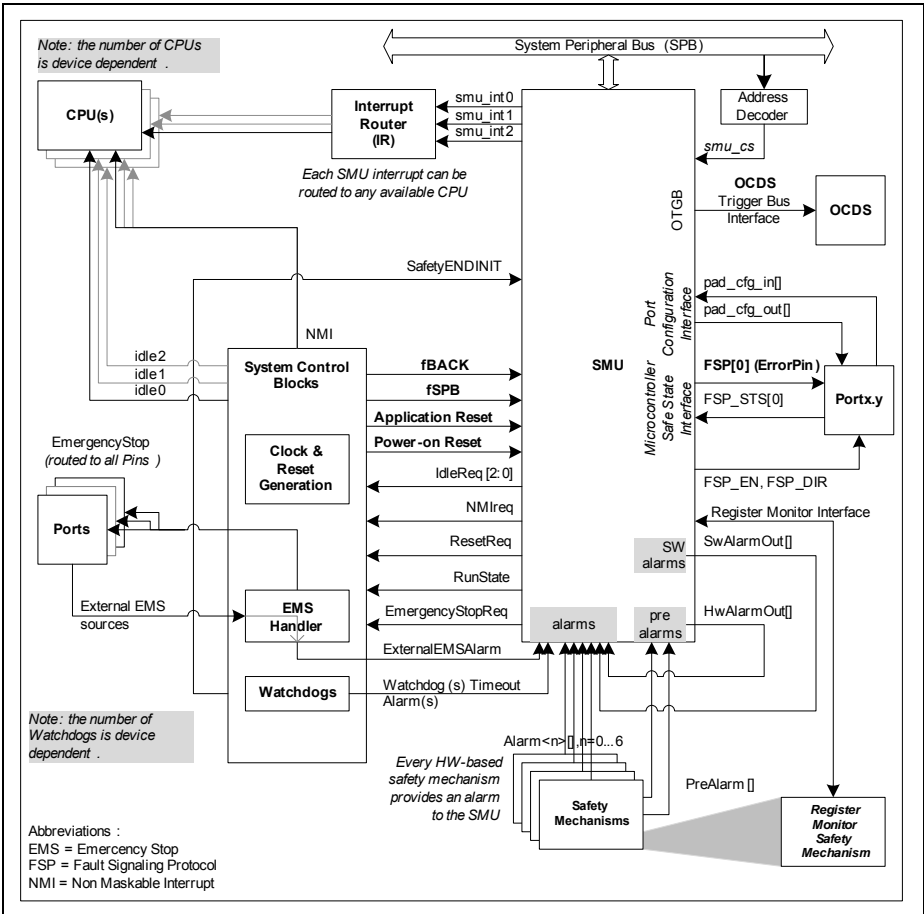


Figure 9-1 SMU Interfaces

9.2 Features

The SMU implements the following features:

- Collects every alarm (error notification) information from every safety mechanism:
 - CPU Lockstep
 - SRAM single bit error correction
 - SRAM uncorrectable error
 - SRAM Monitor: Address Buffer overflow
 - SRAM addressing fault
 - Program FLASH single bit error correction
 - Program FLASH double bit error correction
 - Program FLASH arbitrary multiple-bit uncorrectable error
 - Program FLASH Monitor: address buffer full
 - Program FLASH addressing fault
 - Processor interconnect (SRI) safety mechanisms
 - Safety Flip-Flop
 - Clock monitors
 - System PLL & Flexray-PLL loss of lock
 - Input/Output Monitoring Module
 - Interrupt Router and Interrupt Control Units hardware monitor
 - Voltage Monitor for internal voltage regulators
 - Error reporting from Shared Resource Interconnect and System Peripheral Bus
 - Die Temperature Sensor
- Alarm flags are stored in a debug register that is only reset by the Power-on reset, to enable fault diagnosis and possible recovery.
- An alarm emulation facility is provided to enable software-based diagnostics to post an alarm condition with the same properties as the hardware alarms.
- Implements the access protection and SafetyEndinit modes to protect configuration registers.
- Implements a Fault Signaling Protocol (FSP) reporting internal faults to the external environment. The FSP can be configured using the following modes:
 - Bi-stable single pin output (push-pull active low configuration using FSP[0]), also called ErrorPin.
 - Single-bit timed protocol using FSP[0]
- The FSP value driven by the microcontroller can be observed via an internal status flag
 - Additionally a monitor is available to check the timing and state properties of the FSP protocol when a fault is reported.
- After power-on reset the FSP is disabled, software needs to connect the FSP to the assigned port. The port is configured as GPIO.
- Each individual alarm can be configured to activate the fault signaling protocol and/or one of the following internal actions:

Safety Management Unit (SMU)

- generate an Interrupt request to any of the CPUs, concurrent interrupts to several CPUs can be configured
- generate a NMI request to the System Control Unit
- generate a reset request to the System Control Unit leading to a system or application reset
- activate the pad emergency stop signal controlling the safe state of output pads
- generate a CPU idle request
- After reset every alarm (except watchdog timeout) is disabled.
- A lock mechanism is available to protect the SMU configuration in addition to the standard access protection
- Implements internal watchdog(s) timeout pre-warning function.
- Implements an internal watchdog called recovery timer to monitor the execution of critical software error handlers. The watchdog is started automatically by hardware according to configurable alarm events.

9.3 Functional Overview

The SMU collects all the error flags (called alarms in this document) from all the hardware monitors (safety mechanisms) defined by the safety concept. The section **“Alarm Mapping” on Page 9-14** specifies the alarm interface and classifies them into alarm groups. The alarm groups do not define any hierarchy but only a logical mapping used to map input alarm signals to internal configuration registers. The section **“Alarm Handling” on Page 9-41** describes the configuration options that can be specified. The configuration options specify the behavior of the SMU when an alarm event is detected. The alarm event can trigger an internal action and/or the activation of the Error Pin that indicates the presence of a dangerous fault to the external environment. The section **“SMU Control Interface” on Page 9-48** specifies how the SMU can be controlled by software and the dependencies with the hardware operation. The section **“Fault Signaling Protocol (FSP)” on Page 9-52** describes the properties of the external fault signaling protocol defining the timing and logical properties of the Error Pin(s).

The SMU implements the technical safety requirements

9.4 Functional Description

9.4.1 Reset Types

The SMU requires multiple reset types for its operation. The reset types are fully specified in the System Control Units. The reset types that are required by the SMU are:

- Power-on Reset.
- Debug Reset.
- Application Reset.

Additionally the SMU supports the **module reset** feature. The module reset acts as an application reset, it is fully controlled by software using the [SMU_KRST0](#), [SMU_KRST1](#), [SMU_KRSTCLR](#) registers. The module reset only affects the SMU kernel and not the System Peripheral Bus (SPB) bus interface logic.

[Table 9-1 “Effect of Reset Types to SMU functionality” on Page 9-6](#) specifies the scope of each reset type to the SMU functions (a function includes the control and configuration registers and the related logic).

Table 9-1 Effect of Reset Types to SMU functionality

SMU Function	Module Reset	Application Reset	Debug Reset	System Reset	Power-on Reset
SMU FSP Function Chapter 9.4.8	Not Affected	Not Affected	Not Affected	Not Affected	Reset
SMU SSM Function Chapter 9.4.7	Not Affected	Not Affected	Not Affected	Not Affected	Reset
SMU Debug Function Chapter 9.4.9	Not Affected	Not Affected	Reset	Not Affected	Reset
SMU Alarm Debug Registers Chapter 9.5.6	Not Affected	Not Affected	Not Affected	Not Affected	Reset
SMU_PCTL.PCS Register Field SMU_PCTL	Not Affected	Not Affected	Not Affected	Not Affected	Reset

Safety Management Unit (SMU)

Table 9-1 Effect of Reset Types to SMU functionality (cont'd)

SMU Function	Module Reset	Application Reset	Debug Reset	System Reset	Power-on Reset
SMU SPB BPI	Not Affected	Reset	Not Affected	Reset	Reset
SMU Other Functions	Reset	Reset	Not Affected	Reset	Reset

9.4.2 Interfaces Overview

This section describes the main interface signals between the SMU and the other modules.

9.4.2.1 Interfaces to SCU

Internal actions resulting from an alarm event that interface to the System Control Unit. The interface signals are:

- Emergency Stop Request
- Reset Request
- Idle Request
- NMI Request

9.4.2.2 Interfaces to the Interrupt Router

Internal actions resulting from an alarm event that interface to the Interrupt Router. The interface signals are:

- SMU Interrupt Request 0
- SMU Interrupt Request 1
- SMU Interrupt Request 2

The mapping of SMU Interrupt Requests to the Interrupt Router (IR) interrupt nodes can be found in the IR chapter.

The **SMU_AGC.IGCS<i>**, $i=\{0,1,2\}$ register fields provide the software interface to control how the SMU triggers interrupt requests to the interrupt router.

Each **SMU_AGC.IGCS<i>** is a 3-bits bit-field: **SMU_AGC.IGCS<i>[2:0]** that defines a direct relationship to the 3 SMU interrupt requests:

- **SMU_AGC.IGCS<i>[0]** shall be set to '1' to trigger SMU Interrupt Request 0
- **SMU_AGC.IGCS<i>[1]** shall be set to '1' to trigger SMU Interrupt Request 1
- **SMU_AGC.IGCS<i>[2]** shall be set to '1' to trigger SMU Interrupt Request 2

The usage of the three **SMU_AGC.IGCS<i>** bit-fields is defined in **“Alarm Configuration” on Page 9-41**; where each **SMU_AGC.IGCS<i>** can be selected as an internal action.

9.4.2.3 Interface to the Ports (Error Pin)

The generic port structure is presented in presented in **Figure 9-2 “Generic Port Structure” on Page 9-9**.

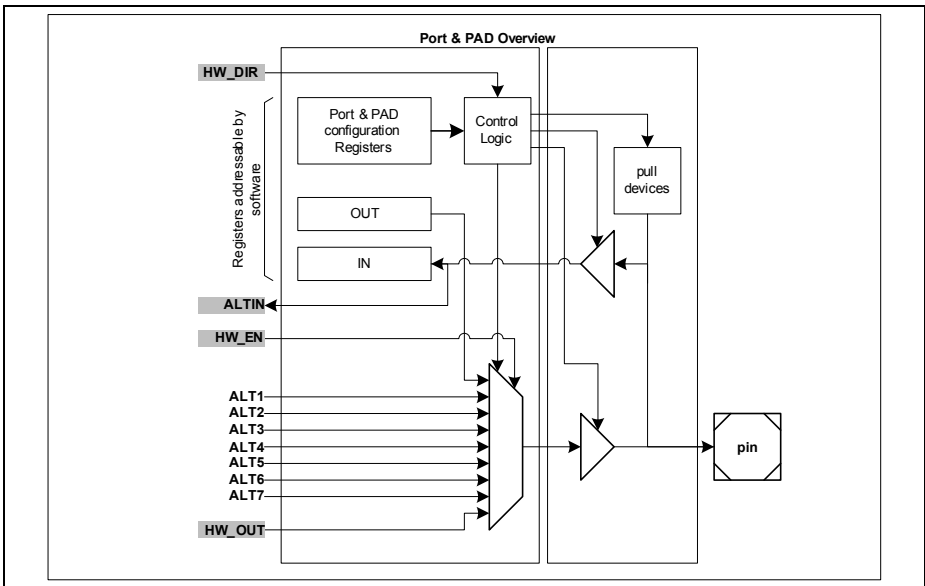


Figure 9-2 Generic Port Structure

The port can be connected to peripheral functions via the ALT_x inputs. This is the default state of the port after power-on reset. The SMU connects to the port using the HW_DIR, ALTIN, HW_EN, HW_OUT signals. When the HW_EN port input is driven active by the SMU, SMU gets full control over the port data path, bypassing any other software configuration related to the usage of the ALT_x inputs.

Figure 9-3 “SMU Data Path Interfaces with the Ports” on Page 9-10 fully specifies the data path connectivity of the SMU with the Ports. The mapping to the port for the product can be found in the port specification chapter of the TC26x B-Step microcontroller.

Safety Management Unit (SMU)

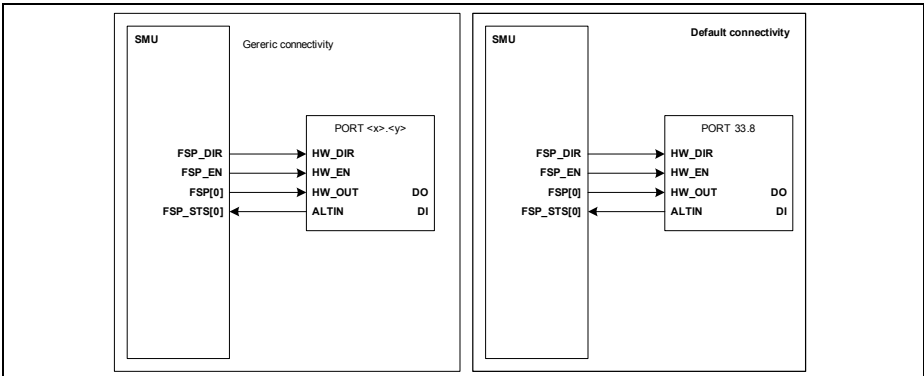


Figure 9-3 SMU Data Path Interfaces with the Ports

Figure 9-4 “SMU/PAD Control Interface to the PADs” on Page 9-10 provides a more detailed overview of the PORT structure and highlights the signals that play a role in the SMU connectivity.

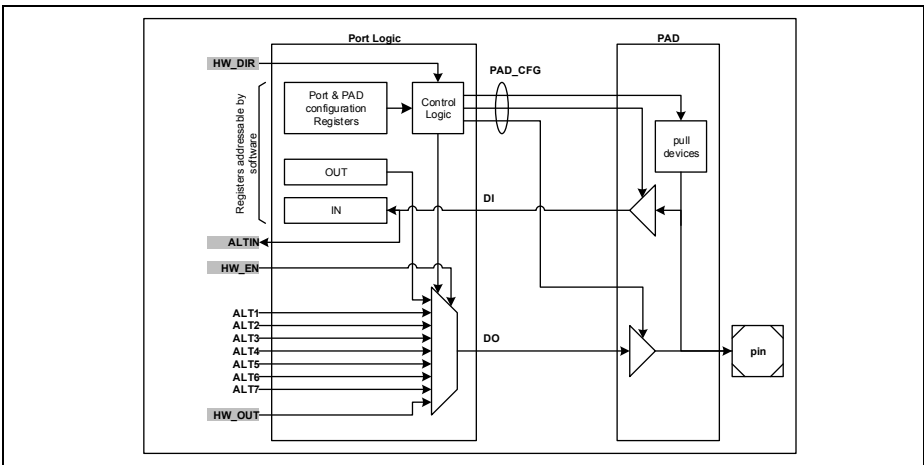


Figure 9-4 SMU/PAD Control Interface to the PADs

While FSP[0] is controlled by hardware, the FSP_DIR and FSP_EN are controlled by software as follows:

- FSP_DIR output is directly driven by the **SMU_PCTL** register HWDIR field
- FSP_EN output is directly driven by the **SMU_PCTL** register HWEN field

Safety Management Unit (SMU)

SMU_PCTL provides a field PCS that enables software to change the PAD control. With **SMU_PCTL** HWDIR, HWEN and PCS fields, software can control the PAD ownership transition from GPIO to full SMU hardware control following the guidelines provided by the **SMU_PCTL** register description.

To satisfy safety requirements it shall be ensured that the PAD operation is not affected by an application or system reset: the application can configure a system reset and the SMU Error Pin fault state activation upon detection of a critical alarm. This is achieved by safeguarding all the PAD configuration signals (see PAD_CFG signals [Figure 9-4](#)) into a special register. The contents of this register are exposed to software using the **SMU_PCTL**.PCFG field. They are only available for debug purposes.

The contents of the **SMU_PCTL** register are locked by the **SMU_KEYS** register and are only reset by a power-on reset, therefore the PAD configuration remains stable even in the presence of an application or system reset. Furthermore the **SMU_PCTL** register is implemented using dedicated Safety Flip-Flops safety mechanism that detects in run-time any bit change caused by a random hardware fault.

Refer to [“SMU Integration Guidelines” on Page 9-13](#) for the SMU and PORT configuration steps that are required to use the Error Pin.

9.4.2.4 Interface to the Safety Flip-Flop Safety Mechanism

The SMU provides a register interface that enables to test the Safety Flip-Flop safety mechanism. The register interface is realized by the **SMU_RMCTL**, **SMU_RMEF**, **SMU_RMSTS** registers.

9.4.3 SMU Integration Guidelines

This chapter extends the “[Interfaces Overview](#)” on [Page 9-8](#) section by providing additional information for the usage of the Error Pin (“[Fault Signaling Protocol \(FSP\)](#)” on [Page 9-52](#)) in combination with other input/output (GPIO) functions of the microcontroller.

Note: The PAD properties (push-pull, open-drain, drive strength,...) are configured in the registers of the PORT to which the SMU connects to. These registers are described in the PORT chapter of the microcontroller.

- During power-on-reset, the Error Pin is in high impedance: the pull devices are disabled under hardware control.
- After power-on-reset the default mode of the PORT to which the Error Pin is connected is GPIO. In that mode the pull devices can be used.
- Before changing the ownership of the PAD to SMU, software shall configure the PORT registers including the following:
 - Software shall disable the pull devices if GPIO is not used
 - Software shall program the GPIO registers of the error pin to strong driver output constant low
- To enable SMU to control the Error Pin PAD, software shall activate the PAD configuration safeguarding process (see “[Interface to the Ports \(Error Pin\)](#)” on [Page 9-8](#)).
 - The safeguarding process requires a software action that consists on writing a 1 into the `SMU_PCTL.PCS` field. **Only with the first transition from 0 to 1 leads to the safeguarding process. A new PORT configuration followed by a new transition from 0 to 1 of the `SMU_PCTL.PCS` has no effect to the hardware.**

9.4.4 Alarm Mapping

This section defines the mapping between the alarm signals at the input of the SMU and the alarm configuration registers. For that purpose alarm groups are defined. There is a one to one relationship between an alarm group index ALM<n>[index] signal and the alarm configuration and status registers (AG<n>[index]). A group is made of up to 32 input alarms; for convenience some entries may be reserved.

9.4.4.1 Pre-alarm Definition

There are situations where it is not necessary to implement configuration and status registers for every alarm; a typical case is a module with multiple SRAM instances. For that purpose a so-called PreAlarm[] bus is available to which the alarms that can be combined together are connected to. The result of the combination is sent to a HwAlarmOut[] output so that HwAlarmOut[] = Combine(PreAlarm[n], PreAlarm[m],...). The tables in **“Pre-alarm Signals” on Page 9-14** fully specify the arguments for the Combine() functions. The Combine() function implements a logical OR between all arguments.

Note: The unspecified (or unused) PreAlarm[] bits shall be tied to the logic level ‘0’ at the SMU input.

Note: The unspecified (or unused) HwAlarmOut[] bits shall be driven to the logic level ‘0’ by the SMU.

9.4.4.2 Pre-alarm Signals

In the next tables the PreAlarm[] bits are specified in the form index=MODULE.(type of error message), where MODULE.(type of error message) enables to identify the module instance and the nature of the error message. The error message is specified in a functional way and does not refer to a real signal name in the design. The modules connecting to the SMU shall use the same functional name to enable a cross-reference through the overall specification.

Table 9-2 HwAlarmOut[3:0]: CPU0 DCACHE/DSPR SRAM

Function
HwAlarmOut[0]= PreAlarm[0=CPU0.DMI.DSPR.(ECC single bit correction)]
HwAlarmOut[1]= PreAlarm[2=CPU0.DMI.DSPR.(ECC uncorrectable error)]

Table 9-2 HwAlarmOut[3:0]: CPU0 DCACHE/DSPR SRAM (cont'd)

Function
HwAlarmOut[2]= PreAlarm[4=CPU0.DMI.DSPR.(Address error)]
HwAlarmOut[3]= PreAlarm[6=CPU0.DMI.DSPR.(Address buffer overflow)]

Table 9-3 HwAlarmOut[7:4]: CPU1 DCACHE/DSPR SRAM

Function
HwAlarmOut[4]= PreAlarm[8=CPU1.DMI.DSPR.(ECC single bit correction)]
HwAlarmOut[5]= PreAlarm[10=CPU1.DMI.DSPR.(ECC uncorrectable error)]
HwAlarmOut[6]= PreAlarm[12=CPU1.DMI.DSPR.(Address error)]
HwAlarmOut[7]= PreAlarm[14=CPU1.DMI.DSPR.(Address buffer overflow)]

Table 9-4 HwAlarmOut[15:8]: CPU2 SRAMs

Function
HwAlarmOut[8]= PreAlarm[16=CPU2.DMI.DSPR.(ECC single bit correction)]
HwAlarmOut[9]= PreAlarm[18=CPU2.DMI.DSPR.(ECC uncorrectable error)]
HwAlarmOut[10]= PreAlarm[20=CPU2.DMI.DSPR.(Address error)]
HwAlarmOut[11]= PreAlarm[22=CPU2.DMI.DSPR.(Address buffer overflow)]

Table 9-5 HwAlarmOut[19:16]: GTM SRAMs

Function
HwAlarmOut[16]= PreAlarm[32=GTM.SRAM.FIFO.(ECC single bit correction)] or PreAlarm[33=GTM.SRAM.RAM0.(ECC single bit correction)] or PreAlarm[34=GTM.SRAM.RAM1.(ECC single bit correction)] or PreAlarm[35=GTM.SRAM.RAM1a.(ECC single bit correction)] or PreAlarm[36=GTM.SRAM.RAM1b.(ECC single bit correction)] or PreAlarm[37=GTM.SRAM.RAM2.(ECC single bit correction)]
HwAlarmOut[17=GTM.SRAM.(ECC uncorrectable error)]= PreAlarm[38=GTM.SRAM.FIFO.(ECC uncorrectable error)] or PreAlarm[39=GTM.SRAM.RAM0.(ECC uncorrectable error)] or PreAlarm[40=GTM.SRAM.RAM1.(ECC uncorrectable error)] or PreAlarm[41=GTM.SRAM.RAM1a.(ECC uncorrectable error)] or PreAlarm[42=GTM.SRAM.RAM1b.(ECC uncorrectable error)] or PreAlarm[43=GTM.SRAM.RAM2.(ECC uncorrectable error)]
HwAlarmOut[18]= PreAlarm[44=GTM.SRAM.FIFO.(Address error)] or PreAlarm[45=GTM.SRAM.RAM0.(Address error)] or PreAlarm[46=GTM.SRAM.RAM1.(Address error)] or PreAlarm[47=GTM.SRAM.RAM1a.(Address error)] or PreAlarm[48=GTM.SRAM.RAM1b.(Address error)] or PreAlarm[49=GTM.SRAM.RAM2.(Address error)]
HwAlarmOut[19]= PreAlarm[50=GTM.SRAM.FIFO.(Address Buffer overflow)] or PreAlarm[51=GTM.SRAM.RAM0.(Address Buffer overflow)] or PreAlarm[52=GTM.SRAM.RAM1.(Address Buffer overflow)] or PreAlarm[53=GTM.SRAM.RAM1a.(Address Buffer overflow)] or PreAlarm[54=GTM.SRAM.RAM1b.(Address Buffer overflow)] or PreAlarm[55=GTM.SRAM.RAM2.(Address Buffer overflow)]

Table 9-6 HwAlarmOut[23:20]: ERAY (FLEXRAY) SRAMs

Function
HwAlarmOut[20]= PreAlarm[56=ERAY.SRAM.OBF0.(ECC single bit correction)] or PreAlarm[58=ERAY.SRAM.TBF_IBF0.(ECC single bit correction)] or PreAlarm[60=ERAY.SRAM.MBF0.(ECC single bit correction)]
HwAlarmOut[21]= PreAlarm[62=ERAY.SRAM.OBF0.(ECC uncorrectable error)] or PreAlarm[64=ERAY.SRAM.TBF_IBF0.(ECC uncorrectable error)] or PreAlarm[66=ERAY.SRAM.MBF0.(ECC uncorrectable error)]
HwAlarmOut[22]= PreAlarm[68=ERAY.SRAM.OBF0.(Address error)] or PreAlarm[70=ERAY.SRAM.TBF_IBF0.(Address error)] or PreAlarm[72=ERAY.SRAM.MBF0.(Address error)]
HwAlarmOut[23]= PreAlarm[74=ERAY.SRAM.OBF0.(Address Buffer overflow)] or PreAlarm[76=ERAY.SRAM.TBF_IBF0.(Address Buffer overflow)] or PreAlarm[78=ERAY.SRAM.MBF0.(Address Buffer overflow)]

Table 9-7 HwAlarmOut[27:24]: CAN SRAM

Function
HwAlarmOut[24]= PreAlarm[80=CAN.SRAM.MCAN0.(ECC single bit correction)]
HwAlarmOut[25]= PreAlarm[82=CAN.SRAM.MCAN0.(ECC uncorrectable error)]
HwAlarmOut[26]= PreAlarm[84=CAN.SRAM.MCAN0.(Address error)]
HwAlarmOut[27]= PreAlarm[86=CAN.SRAM.MCAN0.(Address Buffer overflow)]

Table 9-8 HwAlarmOut[31:28]: LMU sub-system SRAMs

Function
HwAlarmOut[28]= PreAlarm[88=LMU.DAM.SRAM(ECC single bit correction)]
HwAlarmOut[29]= PreAlarm[90=LMU.DAM.SRAM(ECC uncorrectable error)]

Safety Management Unit (SMU)
Table 9-8 HwAlarmOut[31:28]: LMU sub-system SRAMs

Function
HwAlarmOut[30]= PreAlarm[92=LMU.DAM.SRAM(Address error)]
HwAlarmOut[31]= PreAlarm[94=LMU.DAM.SRAM(Address buffer overflow)]

Table 9-9 HwAlarmOut[34:32]: SRI Agents

Function
HwAlarmOut[32]= PreAlarm[101=PMU.SRI_SLAVE(SRI Address Phase Error)] PreAlarm[103=LMU.SRI_SLAVE(SRI Address Phase Error)] or PreAlarm[104=XBAR_SRI.SRI_SLAVE(SRI Address Phase Error)]
HwAlarmOut[33]= PreAlarm[110=PMU.SRI_SLAVE(SRI Write Data Phase Error)] or PreAlarm[112=LMU.SRI_SLAVE(SRI Write Data Phase Error)] or PreAlarm[113=XBAR_SRI.SRI_SLAVE(SRI Write Data Phase Error)]
HwAlarmOut[34]= PreAlarm[115=DMA.SRI_MASTER(SRI Read Data Phase Error)] or PreAlarm[116=HSSL.SRI_MASTER(SRI Read Data Phase Error)] or PreAlarm[117=SFI.SRI_MASTER(SRI Read Data Phase Error)] or PreAlarm[124=LMU.DAM.SRI_MASTER(SRI Read Data Phase Error)]
PreAlarm[109:105], PreAlarm[111], PreAlarm[114], PreAlarm[123:118] are reserved

Table 9-10 HwAlarmOut[35]: Safety Flip-Flop

Function
HwAlarmOut[35]= PreAlarm[125=SMU.(Safety FF Error Detection)] or PreAlarm[126=SCU.(Safety FF Error Detection)] or PreAlarm[127=MTU.(Safety FF Error Detection)]
Note: if a listed module does not implement a Safety FF, the corresponding input shall be tied to '0'.

Table 9-11 HwAlarmOut[44:41]: Misc. SRAMs

Function
<p>HwAlarmOut[41]= PreAlarm[142=PMU.FSI.SRAM(ECC single bit correction)] or PreAlarm[143=DMA.SRAM(ECC single bit correction)] or PreAlarm[144=Ethernet.SRAM(ECC single bit correction)] or PreAlarm[145=PSI5.SRAM(ECC single bit correction)] or PreAlarm[146=MCDS.SRAM(ECC single bit correction)] or PreAlarm[147=CIF.SRAM(ECC single bit correction)] or PreAlarm[148=CIF.JPEG1_4.SRAM(ECC single bit correction)] or PreAlarm[149=CIF.JPEG3.SRAM(ECC single bit correction)]</p>
<p>HwAlarmOut[42]= PreAlarm[150=PMU.FSI.SRAM(ECC uncorrectable error)] or PreAlarm[151=DMA.SRAM(ECC uncorrectable error)] or PreAlarm[152=Ethernet.SRAM(ECC uncorrectable error)] or PreAlarm[153=PSI5.SRAM(ECC uncorrectable error)] or PreAlarm[154=MCDS.SRAM(ECC uncorrectable error)] or PreAlarm[155=CIF.SRAM(ECC uncorrectable error)] or PreAlarm[156=CIF.JPEG1_4.SRAM(ECC uncorrectable error)] or PreAlarm[157=CIF.JPEG3.SRAM(ECC uncorrectable error)]</p>
<p>HwAlarmOut[43]= PreAlarm[158=PMU.FSI.SRAM(Address error)] or PreAlarm[159=DMA.SRAM(Address error)] or PreAlarm[160=Ethernet.SRAM(Address error)] or PreAlarm[161=PSI5.SRAM(Address error)] or PreAlarm[162=MCDS.SRAM(Address error)] or PreAlarm[163=CIF.SRAM(Address error)] or PreAlarm[164=CIF.JPEG1_4.SRAM(Address error)] or PreAlarm[165=CIF.JPEG3.SRAM(Address error)]</p>
<p>HwAlarmOut[44]= PreAlarm[166=PMU.FSI.SRAM(Address Buffer overflow)] or PreAlarm[167=DMA.SRAM(Address Buffer overflow)] or PreAlarm[168=Ethernet.SRAM(Address Buffer overflow)] or PreAlarm[169=PSI5.SRAM(Address Buffer overflow)] or PreAlarm[170=MCDS.SRAM(Address Buffer overflow)] or PreAlarm[171=CIF.SRAM(Address Buffer overflow)] or PreAlarm[172=CIF.JPEG1_4.SRAM(Address Buffer overflow)] or PreAlarm[173=CIF.JPEG3.SRAM(Address Buffer overflow)]</p>

Table 9-11 HwAlarmOut[44:41]: Misc. SRAMs (cont'd)

Function
HwAlarmOut[41]= PreAlarm[142=PMU.FSI.SRAM(ECC single bit correction)] or PreAlarm[143=DMA.SRAM(ECC single bit correction)] or PreAlarm[144=Ethernet.SRAM(ECC single bit correction)] or PreAlarm[146=MCDS.SRAM(ECC single bit correction)] or PreAlarm[147=CIF.SRAM(ECC single bit correction)] or PreAlarm[148=CIF.JPEG1_4.SRAM(ECC single bit correction)] or PreAlarm[149=CIF.JPEG3.SRAM(ECC single bit correction)]
Note: HwAlarmOut[44:41] connects respectively to ALM4[19:16]

Table 9-12 HwAlarmOut[45]: Watchdogs timeout

Function
HwAlarmOut[45]= PreAlarm[174=WDTCPU0.(timeout)] or PreAlarm[175=WDTCPU1.(timeout)] or PreAlarm[176=WDTCPU2.(timeout)] or PreAlarm[177=WDTS.(timeout)]

Table 9-13 HwAlarmOut[50:46]: PMU Alarms

Function
HwAlarmOut[46]= PreAlarm[178=PMU.PFLASH0.(ECC single bit correction notification)] or PreAlarm[179=PMU.PFLASH1.(ECC single bit correction notification)]
HwAlarmOut[47]= PreAlarm[186=PMU.PFLASH0.(ECC double bit correction notification)] or PreAlarm[187=PMU.PFLASH1.(ECC double bit correction notification)]
HwAlarmOut[48]= PreAlarm[194=PMU.PFLASH0.(ECC non correctable multiple bit error)] or PreAlarm[195=PMU.PFLASH1.(ECC non correctable multiple bit error)]
HwAlarmOut[49]= PreAlarm[202=PMU.PFLASH0.(Addressing error)] or PreAlarm[203=PMU.PFLASH1.(Addressing error)]
HwAlarmOut[50]= PreAlarm[210=PMU.PFLASH0.(Address buffer full)] or PreAlarm[211=PMU.PFLASH1.(Address buffer full)]

Table 9-14 HwAlarmOut[51]: Access Enable Alarms

Function
HwAlarmOut[51]= PreAlarm[218=IR.(Access Enable Error)] ¹⁾ or PreAlarm[219=SCU.DTS.(Access Enable Error)]
PreAlarm[229:220] bits are reserved for extensions of the Access Enable Alarms

- 1) This alarm is related to the event where a master not configured in the module ACCEN register tries to access a register protected by the ACCEN scheme. Depending on the module all or only a subset of the module registers are protected by the ACCEN. Most of the modules only generate a bus error condition and no dedicated ACCEN alarm when the aforementioned event happens.

9.4.4.3 Non-compliant Alarms

This section describes the alarms that are not compliant with the SMU alarm protocol and for which a special processing is required. They are first connected to a pre-alarm to be processed by the SMU in order to deliver a SMU compatible input alarm signal.

Table 9-15 HwAlarmOut[63] specification: SCR Alarm

Function
HwAlarmOut[63]= PreAlarm[255=SCR(Standby Controller Alarm)]
PreAlarm[255] is a signal clocked by the fBACK clock, it shall be synchronized to the fSPB clock. HwAlarmOut[63] shall be connected to ALM3[24].

Note: The Standby Controller is not available in all devices. When not available PreAlarm[255] shall be connected to '0' and ALM3[24] shall be reserved.

9.4.4.4 Internal SMU Alarms

The SMU has also the capability to generate its own alarm signals. They are specified in [Table 9-16 “Internal Alarms” on Page 9-23](#).

Table 9-16 Internal Alarms

HwAlarmOut	Function
HwAlarmOut[60]	Recovery Timer 0 Timeout
HwAlarmOut[61]	Recovery Timer 1 Timeout
HwAlarmOut[62]	FSP Fault State Activation
HwAlarmOut[63]	Reserved

9.4.4.5 Alarm Signals

The following tables fully specify the mapping between the alarms provided by the safety mechanisms implemented by the microcontroller and the alarm groups. Cells in **bold** refer to alarms coming from multiple instances of the same safety mechanism that are logically combined to a single alarm definition at the SMU level.

In the following tables the column “Safety Mechanism & Error Indication” indicates to which safety mechanism the alarm is related to. If multiple safety mechanisms are indicated, the alarm corresponds to the detection of an error by one of the listed safety mechanisms.

For some safety mechanisms different terms are used in the microcontroller documents; the following list provides a guideline between the term used in the alarm tables and the other definitions. In bold the definition used in the alarm tables.

- **Register Access Protection** or alternatively called Safety Register Protection
 - Purpose: Monitors the master identifier of a given bus-master during a write access to a configuration register. The master identifier is a hard-coded information that is provided during any bus access. If the master identifier is not enabled by the Register Access Protection configuration registers (ACCEN0) the write is aborted. Most of the modules do not provide a dedicated alarm for this event and instead will generate a bus error. Therefore the Register Access Protection is only documented where a dedicated alarm is available.
 - Note: for peripherals that implement memory-mapped SRAMs, the write accesses to the memories are monitored as well.
- **Bus-level Memory Protection Unit (MPU)** or alternatively called Safety Memory Protection
 - Purpose: Monitors the master identifier and the address of a given bus-master during a write access to a local SRAM. The master identifier is a hard-coded information that is provided during any bus access. If the master identifier is enabled by the Bus-level MPU configuration registers and the address is within the valid address range the write is accepted, otherwise the write is aborted and a Bus-level MPU alarm is issued.
 - The SRAMs monitored are the {PSPR, DSPR} SRAMs of each CPU and the LMU SRAMs when available in the product.

Safety Management Unit (SMU)
Table 9-17 TC26x B-Step Alarm Mapping related to Alarm 0 group

Alarm Index	Module	Safety Mechanism & Alarm Indication
ALM0[0]	CPU0	Safety Mechanism: Lockstep CPU Alarm: Lockstep Comparator Error
ALM0[1]	CPU0	Safety Mechanism: Bus-level Memory Protection Unit Alarm: Bus-level MPU violation. Safety Mechanism: Register Access Protection Alarm: Access Protection violation.
ALM0[2]	Reserved	Reserved
ALM0[3]	CPU0	Safety Mechanism: SRAM Error Detection Code (EDC) Alarm: PCACHE TAG uncorrectable error
ALM0[4]	CPU0	Safety Mechanism: SRAM Address Monitor Alarm: PCACHE TAG address error
ALM0[5]	CPU0	Safety Mechanism: SRAM Monitor Alarm: PCACHE TAG Address Buffer overflow
ALM0[6]	CPU0	Safety Mechanism: SRAM ECC Alarm: Unified PCACHE/PSPR single-bit correction
ALM0[7]	CPU0	Safety Mechanism: SRAM ECC Alarm: Unified PCACHE/PSPR uncorrectable error
ALM0[8]	CPU0	Safety Mechanism: SRAM Address Monitor Alarm: Unified PCACHE/PSPR Address error
ALM0[9]	CPU0	Safety Mechanism: SRAM Monitor Alarm Unified PCACHE/PSPR Address Buffer overflow
ALM0[10]	CPU0	Safety Mechanism(s): SRAM ECC Alarm: Unified DCACHE/DSPR single-bit correction
ALM0[11]	CPU0	Safety Mechanism(s): SRAM ECC Alarm: Unified DCACHE/DSPR uncorrectable error
ALM0[12]	CPU0	Safety Mechanism(s): SRAM Address Monitor Alarm: Unified DCACHE/DSPR address error
ALM0[13]	CPU0	Safety Mechanism(s): SRAM Monitor Alarm: Unified DCACHE/DSPR Address Buffer overflow
ALM[17:14]	Reserved	Reserved
ALM0[18]	CPU0	Safety Mechanism: SRI End-to-End EDC Alarm: CPU Instruction Fetch SRI Interface EDC Error

Safety Management Unit (SMU)

Table 9-17 TC26x B-Step Alarm Mapping related to Alarm 0 group (cont'd)

Alarm Index	Module	Safety Mechanism & Alarm Indication
ALM0[19]	CPU0	Safety Mechanism: SRI End-to-End EDC Alarm: CPU Data SRI Interface (Load/Store) EDC Error
ALM0[31:20]	Reserved	Reserved

Table 9-18 TC26x B-Step Alarm Mapping for related to ALM1 group

Alarm Index	Module	Description
ALM1[0]	CPU1	Safety Mechanism: Lockstep CPU Alarm: Lockstep Comparator Error
ALM1[1]	CPU1	Safety Mechanism: Bus-level Memory Protection Unit Alarm: Bus-level MPU violation. Safety Mechanism: Register Access Protection Alarm: Access Protection violation.
ALM1[2]	Reserved	Reserved
ALM1[3]	CPU1	Safety Mechanism: SRAM Error Detection Code (EDC) Alarm: PCACHE TAG uncorrectable error
ALM1[4]	CPU1	Safety Mechanism: SRAM Address Monitor Alarm: PCACHE TAG SRAM address error
ALM1[5]	CPU1	Safety Mechanism: SRAM Monitor Alarm: PCACHE TAG SRAM address buffer overflow
ALM1[6]	CPU1	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: Unified PCACHE/PSPR single-bit correction
ALM1[7]	CPU1	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: Unified PCACHE/PSPR uncorrectable error
ALM1[8]	CPU1	Safety Mechanism: SRAM Address Monitor Alarm: Unified PCACHE/PSPR address error
ALM1[9]	CPU1	Safety Mechanism: SRAM Monitor Alarm: Unified PCACHE/PSPR address buffer overflow
ALM1[13:10] connects to pre-alarms specified by Table 9-3 “HwAlarmOut[7:4]: CPU1 DCACHE/DSPR SRAM” on Page 9-15		
ALM1[10]	CPU1	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: Unified DCACHE/DSPR single-bit correction
ALM1[11]	CPU1	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: Unified DCACHE/DSPR uncorrectable error
ALM1[12]	CPU1	Safety Mechanism: SRAM Address Monitor Alarm: Unified DCACHE/DSPR address error
ALM1[13]	CPU1	Safety Mechanism: SRAM Monitor Alarm: Unified DCACHE/DSPR address buffer overflow
ALM1[14]	CPU1	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: DCACHE TAG SRAM correction

Safety Management Unit (SMU)
Table 9-18 TC26x B-Step Alarm Mapping for related to ALM1 group (cont'd)

Alarm Index	Module	Description
ALM1[15]	CPU1	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: DCACHE TAG SRAM uncorrectable error
ALM1[16]	CPU1	Safety Mechanism: SRAM Address Monitor Alarm: DCACHE TAG SRAM address error
ALM1[17]	CPU1	Safety Mechanism: SRAM Monitor Alarm: DCACHE TAG SRAM address buffer overflow
ALM1[18]	CPU1	Safety Mechanism: SRI End-to-End EDC Alarm: CPU Instruction Fetch SRI Interface EDC Error
ALM1[19]	CPU1	Safety Mechanism: SRI End-to-End EDC Alarm: CPU Data SRI Interface (Load/Store) EDC Error
ALM1[31:20]	Reserved	Reserved

Table 9-19 TC26x B-Step Alarm Mapping related to ALM2 group

Alarm Index	Module	Description
ALM2[0]	Reserved	Reserved
ALM2[1]	Reserved	Reserved
ALM2[6:2] connects to pre-alarms specified by Table 9-13 “HwAlarmOut[50:46]: PMU Alarms” on Page 9-20		
ALM2[2]	PMU	Safety Mechanism: PFLASH ECC Alarm: single bit correction
ALM2[3]	PMU	Safety Mechanism: PFLASH ECC Alarm: double bit correction
ALM2[4]	PMU	Safety Mechanism: PFLASH ECC Alarm: non correctable multiple bit
ALM2[5]	PMU	Safety Mechanism: PFLASH Alarm: Addressing error
ALM2[6]	PMU	Safety Mechanism: PFLASH Monitor Alarm: address buffer full
ALM2[7]	PMU	Safety Mechanism: PFLASH ECC monitor Alarm: PFLASH ECC error Note: All the PFLASH ECC monitor alarms are combined into a single alarm to the SMU.
ALM2[8]	PMU	Safety Mechanism: PFLASH EDC monitor Alarm: EDC comparator error Note: All the PFLASH EDC monitor alarms are combined into a single alarm to the SMU.
ALM2[9]	Reserved	Reserved
ALM2[10]	Reserved	Reserved
ALM2[11]	Reserved	Reserved
ALM2[12]	Reserved	Reserved
ALM2[13]	Reserved	Reserved
ALM2[14]	Reserved	Reserved
ALM2[15]	LMU	Safety Mechanism: SRAM ECC Monitor Alarm: ECC Error
ALM2[16]	LMU	Safety Mechanism: Register Access Protection Alarm: Register Access Protection error Safety Mechanism: Bus-level MPU Alarm: Bus-level MPU error

Safety Management Unit (SMU)

Table 9-19 TC26x B-Step Alarm Mapping related to ALM2 group (cont'd)

Alarm Index	Module	Description
ALM2[20:17] connects to pre-alarms specified by Table 9-8 “HwAlarmOut[31:28]: LMU sub-system SRAMs” on Page 9-17		
ALM2[17]	LMU	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM single-bit correction
ALM2[18]	LMU	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM uncorrectable error
ALM2[19]	LMU	Safety Mechanism: SRAM Address Monitor Alarm: SRAM Address error
ALM2[20]	LMU	Safety Mechanism: SRAM Monitor Alarm: SRAM Address buffer overflow
ALM2[23:21] connects to pre-alarms specified by Table 9-9 “HwAlarmOut[34:32]: SRI Agents” on Page 9-18		
ALM2[21]	SRI	Safety Mechanism: SRI Error Detection Code (EDC) Alarm: EDC Address phase error Scope: non-CPU modules connected to SRI
ALM2[22]	SRI	Safety Mechanism: SRI Error Detection Code (EDC) Alarm: EDC Write phase error Scope: non-CPU modules connected to SRI
ALM2[23]	SRI	Safety Mechanism: SRI Error Detection Code (EDC) Alarm: EDC Read phase error Scope: non-CPU modules connected to SRI
ALM2[24]	Reserved	Reserved
ALM2[25]	IR	Safety Mechanism: IR Monitor (Error Detection Code covering Configuration & Data Path) Alarm: EDC error
ALM2[26]	IOM	Safety Mechanism: Input/Output Monitor (IOM) Alarm: Pin Mismatch Indication
ALM2[27]	Reserved	Reserved
ALM2[28]	Reserved	Reserved
ALM2[29]	SMU	Safety Mechanism: Recovery Timer 0 Alarm: Timer time-out

Safety Management Unit (SMU)**Table 9-19** TC26x B-Step Alarm Mapping related to ALM2 group (cont'd)

Alarm Index	Module	Description
ALM2[30]	SMU	Safety Mechanism: Recovery Timer 1 Alarm: Timer time-out
ALM2[31]	SMU	Design Measure: ErrorPin Alarm: ErrorPin Fault State Activation This feature enables software to get a notification when the Error Pin is activated by hardware.

Table 9-20 TC26x B-Step Alarm Mapping related to ALM3 group

Alarm Index	Module	Description
ALM3[0]	SCU/CGU	Design Measure: System PLL Oscillator Watchdog (OSC_WDT) Alarm: input clock out of range
ALM3[1]	SCU/CGU	Design Measure: PLL loss-of-lock detection Alarm: System PLL VCO Loss-of-Lock Event
ALM3[2]	SCU/CGU	Design Measure: PLL loss-of-lock detection Alarm: PLL_ERAY VCO Loss-of-Lock Event
ALM3[3]	SCU/CGU	Safety Mechanism: Clock Monitor Alarm: STM clock out of range frequency
ALM3[4]	SCU/CGU	Safety Mechanism: Clock Monitor Alarm: PLL_ERAY out of range frequency
ALM3[5]	SCU/CGU	Safety Mechanism: Clock Monitor Alarm: System PLL out of range frequency
ALM3[6]	SCU/CGU	Safety Mechanism: Clock Monitor Alarm: SRI clock out of range frequency
ALM3[7]	SCU/CGU	Safety Mechanism: Clock Monitor Alarm: SPB clock out of range frequency
ALM3[8]	SCU/CGU	Safety Mechanism: Clock Monitor Alarm: GTM clock out of range frequency
ALM3[9]	SCU/CGU	Safety Mechanism: Clock Monitor Alarm: ADC clock out of range frequency
ALM3[10]	Reserved	Reserved
ALM3[11]	SCU/EVR	Safety Mechanism: Voltage Monitor Alarm: EVR 1.3V under voltage
ALM3[12]	SCU/EVR	Safety Mechanism: Voltage Monitor Alarm: EVR 1.3V over voltage
ALM3[13]	SCU/EVR	Safety Mechanism: Voltage Monitor Alarm: EVR 3.3V under voltage
ALM3[14]	SCU/EVR	Safety Mechanism: Voltage Monitor Alarm: EVR 3.3V over voltage
ALM3[15]	SCU/EVR	Safety Mechanism: Voltage Monitor Alarm: External Supply under voltage

Safety Management Unit (SMU)
Table 9-20 TC26x B-Step Alarm Mapping related to ALM3 group (cont'd)

Alarm Index	Module	Description
ALM3[16]	SCU/EVR	Safety Mechanism: Voltage Monitor Alarm: External Supply over voltage
ALM3[17]	SCU/WDTS	Safety Mechanism: Safety Watchdog Alarm: watchdog time-out
ALM3[18]	SCU/WDTCPU0	Safety Mechanism: CPU0 Watchdog Alarm: watchdog time-out
ALM3[19]	SCU/WDTCPU1	Safety Mechanism: CPU1 Watchdog Alarm: Watchdog time-out
ALM3[20]	SCU/WDTCPU2	Safety Mechanism: CPU2 Watchdog Alarm: watchdog time-out
ALM3[21] connects to pre-alarms specified by Table 9-12 “HwAlarmOut[45]: Watchdogs timeout” on Page 9-20		
ALM3[21]	SCU/WDT	Safety Mechanism: All Watchdogs Alarm: watchdog time-out. This alarm is a logical OR over all watchdog time-out alarms.
ALM3[22] connects to pre-alarms specified by Table 9-14 “HwAlarmOut[51]: Access Enable Alarms” on Page 9-21		
ALM3[22]	SPB Peripherals	Safety Mechanism: Register Access Protection Alarm: Access Enable Error
ALM3[23]	Reserved	Reserved
ALM3[24]	Reserved	Reserved
ALM3[25]	SCU/DTS	Design Measure: Die Temperature Sensor Alarm: Temperature underflow
ALM3[26]	SCU/DTS	Design Measure: Die Temperature Sensor Alarm: Temperature overflow
ALM3[27] connects to pre-alarms specified by Table 9-10 “HwAlarmOut[35]: Safety Flip-Flop” on Page 9-18		
ALM3[27]	Registers	Safety Mechanism: Safety Flip-Flop (redundant or triple-modular register structure). Alarm: Safety Flip-Flop error detection
ALM3[28]	Reserved	Reserved
ALM3[29]	SCU	Design Measure: Emergency Stop Alarm: External Emergency Stop Signal Event

Safety Management Unit (SMU)

Table 9-20 TC26x B-Step Alarm Mapping related to ALM3 group (cont'd)

Alarm Index	Module	Description
ALM3[30]	SRI	Design Measure: Built-in SRI Error Detection Alarm: SRI Bus error event
ALM3[31]	SPB	Design Measure: Built-in SPB Error Detection Alarm: SPB Bus error The SPB bus error can result from multiple root causes: protocol violation, incorrect address, register access protection violation

Table 9-21 TC26x B-Step Alarm Mapping related to ALM4 group

Alarm Index	Module	Description
ALM4[3:0] connects to pre-alarms specified by Table 9-5 “HwAlarmOut[19:16]: GTM SRAMs” on Page 9-16		
ALM4[0]	GTM	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM single-bit correction
ALM4[1]	GTM	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM uncorrectable error
ALM4[2]	GTM	Safety Mechanism: SRAM Address Monitor Alarm: SRAM address error
ALM4[3]	GTM	Safety Mechanism: SRAM Monitor SRAM Address Buffer overflow
ALM4[7:3] connects to pre-alarms specified by Table 9-7 “HwAlarmOut[27:24]: CAN SRAM” on Page 9-17		
ALM4[4]	CAN	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM single-bit correction
ALM4[5]	CAN	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM uncorrectable error
ALM4[6]	CAN	Safety Mechanism: SRAM Address Monitor Alarm: SRAM address error
ALM4[7]	CAN	Safety Mechanism: SRAM Monitor SRAM Address Buffer overflow
ALM4[11:8] connects to pre-alarms specified by Table 9-6 “HwAlarmOut[23:20]: ERAY (FLEXRAY) SRAMs” on Page 9-17		
ALM4[8]	FLEXRAY	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM single-bit correction
ALM4[9]	FLEXRAY	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM uncorrectable error
ALM4[10]	FLEXRAY	Safety Mechanism: SRAM Address Monitor Alarm: SRAM address error
ALM4[11]	FLEXRAY	Safety Mechanism: SRAM Monitor SRAM Address Buffer overflow
ALM4[12]	EMEM	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM single-bit correction

Safety Management Unit (SMU)

Table 9-21 TC26x B-Step Alarm Mapping related to ALM4 group (cont'd)

Alarm Index	Module	Description
ALM4[13]	EMEM	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM ECC uncorrectable error
ALM4[14]	Reserved	Reserved
ALM4[15]	EMEM	Safety Mechanism: SRAM Monitor Alarm: SRAM Address Buffer overflow
ALM4[19:16] connects to pre-alarms specified by Table 9-11 “HwAlarmOut[44:41]: Misc. SRAMs” on Page 9-19		
ALM4[16]	Misc SRAMs¹⁾	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM single-bit correction
ALM4[17]	Misc. SRAMs	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: SRAM uncorrectable error
ALM4[18]	Misc. SRAMs	Safety Mechanism: SRAM Address Monitor Alarm: SRAM address error
ALM4[19]	Misc. SRAMs	Safety Mechanism: SRAM Address Monitor Alarm: SRAM Address Buffer overflow
ALM4[31...20]	Reserved	Reserved

1) The list of miscellaneous SRAMs can be found in [Table 9-11 “HwAlarmOut\[44:41\]: Misc. SRAMs” on Page 9-19](#).

Safety Management Unit (SMU)

A dedicated alarm group **Table 9-22 “TC26x B-Step Alarm Mapping related to ALM5 group” on Page 9-37** enables software to submit alarms and use the SMU infrastructure for alarm management. Alarms can be submitted using the register command interface **SMU_CMD**.

Table 9-22 TC26x B-Step Alarm Mapping related to ALM5 group

Alarm Index	Module	Description
ALM5[0]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 0
ALM5[1]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 1
ALM5[2]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 2
ALM5[3]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 3
ALM5[4]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 4
ALM5[5]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 5
ALM5[6]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 6
ALM5[7]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 7
ALM5[8]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 8
ALM5[9]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 9
ALM5[10]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 10
ALM5[11]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 11
ALM5[12]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 12
ALM5[13]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 13
ALM5[14]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 14

Safety Management Unit (SMU)**Table 9-22 TC26x B-Step Alarm Mapping related to ALM5 group (cont'd)**

Alarm Index	Module	Description
ALM5[15]	Software	Safety Mechanism: Software Monitor Alarm: Software Alarm 15
ALM5[31..16]	Reserved	Reserved

Table 9-23 TC26x B-Step Alarm Mapping related to ALM6 group

Date	Module	Description
ALM6[0]	Reserved	Reserved
ALM6[1]	CPU2	Safety Mechanism: Bus-level Memory Protection Unit Alarm: Bus-level MPU violation. Safety Mechanism: Register Access Protection Alarm: Access Protection violation.
ALM6[2]	Reserved	Reserved
ALM6[3]	CPU2	Safety Mechanism: SRAM Error Detection Code (EDC) Alarm: PCACHE TAG SRAM uncorrectable error
ALM6[4]	CPU2	Safety Mechanism: SRAM Address Monitor Alarm: PCACHE TAG SRAM address error
ALM6[5]	CPU2	Safety Mechanism: SRAM Monitor Alarm: PCACHE TAG SRAM address buffer overflow
ALM6[6]	CPU2	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: Unified PCACHE/PSPR single-bit correction
ALM6[7]	CPU2	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: Unified PCACHE/PSPR uncorrectable error
ALM6[8]	CPU2	Safety Mechanism: SRAM Address Monitor Alarm: Unified PCACHE/PSPR address error
ALM6[9]	CPU2	Safety Mechanism: SRAM Monitor Alarm: Unified PCACHE/PSPR address buffer overflow
ALM6[10]	CPU2	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: Unified DCACHE/DSPR single-bit correction
ALM6[11]	CPU2	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: Unified DCACHE/DSPR uncorrectable error
ALM6[12]	CPU2	Safety Mechanism: SRAM Address Monitor Alarm: Unified DCACHE/DSPR address error
ALM6[13]	CPU2	Safety Mechanism: SRAM Monitor Alarm: Unified DCACHE/DSPR address buffer overflow
ALM6[14]	CPU2	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: DCACHE TAG SRAM single-bit correction
ALM6[15]	CPU2	Safety Mechanism: SRAM Error Correction Code (ECC) Alarm: DCACHE TAG SRAM uncorrectable error
ALM6[16]	CPU2	Safety Mechanism: SRAM Address Monitor Alarm: DCACHE TAG SRAM Address error

Safety Management Unit (SMU)

Table 9-23 TC26x B-Step Alarm Mapping related to ALM6 group (cont'd)

Date	Module	Description
ALM6[17]	CPU2	Safety Mechanism: SRAM Monitor Alarm: DCACHE TAG SRAM Address buffer overflow
ALM6[18]	CPU2	Safety Mechanism: SRI End-to-End EDC Alarm: CPU Instruction Fetch SRI Interface EDC Error
ALM6[19]	CPU2	Safety Mechanism: SRI End-to-End EDC Alarm: CPU Data SRI Interface (Load/Store) EDC Error
ALM6[31..20]	Reserved	Reserved

9.4.5 Alarm Handling

This section specifies the hardware and software alarm processes.

9.4.5.1 Alarm protocol

Each safety mechanism shall interface with the SMU using a pre-defined protocol. The protocol enables to cross clock domains in a reliable manner. The operation of the protocol has no influence to the software layers.

9.4.5.2 Alarm Configuration

Upon reception of an alarm event the SMU decodes the actions to be performed. The action can be classified into an internal behavior and an external behavior. Both the internal and external behavior can be configured for every alarm.

The external behavior is related to the Fault Signaling Protocol (see [“Fault Signaling Protocol \(FSP\)” on Page 9-52](#)). The external behavior is configured via the following registers:

- [SMU_AG0FSP](#)
- [SMU_AG1FSP](#)
- [SMU_AG2FSP](#)
- [SMU_AG3FSP](#)
- [SMU_AG4FSP](#)
- [SMU_AG5FSP](#)
- [SMU_AG6FSP](#)

Alarm Action Configuration Registers

The internal behavior of the SMU under the presence of an alarm is controlled via the following registers:

- [SMU_AG0CFx \(x=0-2\)](#)
- [SMU_AG1CFx \(x=0-2\)](#)
- [SMU_AG2CFx \(x=0-0\)](#), [SMU_AG2CFx \(x=1-1\)](#), [SMU_AG2CFx \(x=2-2\)](#)
- [SMU_AG3CFx \(x=0-0\)](#), [SMU_AG3CFx \(x=1-1\)](#), [SMU_AG3CFx \(x=2-2\)](#)
- [SMU_AG4CFx \(x=0-2\)](#)
- [SMU_AG5CFx \(x=0-2\)](#)
- [SMU_AG6CFx \(x=0-2\)](#)

Alarm Action Configuration Codes

The internal behavior is specified by a 3-bit code as follows:

- Code = SMU_AG<n>CF2. SMU_AG<n>CF1. SMU_AG<n>CF0, n=0...6

Table 9-24 SMU Alarm Configuration

Code	Name	Behavior
0x0	SMU_NA	No Action. Reset value. Alarm disabled.
0x1	SMU_RSVD	Reserved. No Action. Alarm disabled.
0x2	SMU_IGCS0	Sends an interrupt request to the interrupt system according to the Interrupt Generation Configuration Set 0 from the SMU_AGC register.
0x3	SMU_IGCS1	Sends an interrupt request to the interrupt system according to the Interrupt Generation Configuration Set 1 from the SMU_AGC register.
0x4	SMU_IGCS2	Sends an interrupt request to the interrupt system according to the Interrupt Generation Configuration Set 2 from the SMU_AGC register.
0x5	SMU_NMI	Sends an NMI request to the SCU
0x6	SMU_RESET	Sends a reset request to the SCU. The SCU shall be configured to generate an application or system reset.
0x7	SMU_IDLE	Triggers a CPU idle request using Idle Configuration Set from the SMU_AGC register

Unless otherwise specified the reset state of the alarms is “No Action”. Refer to reset values of registers listed in **“Alarm Action Configuration Registers” on Page 9-41** and special cases described in **“Watchdog Alarms” on Page 9-46**.

9.4.5.3 Alarm operation

Whenever an input alarm event is detected and the SMU state machine is in the RUN or FAULT state, the SMU checks for the corresponding actions to be done for the internal action and for the FSP in a concurrent way. If an input alarm event is detected and no action is specified for the alarm, the corresponding bit shall be set to 1 as well but no action takes place.

The processing of the incoming alarm events is performed as follows:

- The alarm groups are scanned in a linear order starting with alarm group 0. The switching from one alarm group to the next one, takes several clock cycles.
- During the switch operation to a new alarm group (AG_y) a snapshot of the pending alarms for AG_y is made by hardware.
- If there are no pending alarms for the active alarm group AG_y nothing happens during the processing time slot related to this alarm group.
- If there are pending alarms they are processed in a fixed order 0 up to 31.
 - If new alarms arrived for the active alarm group, they are only processed in the next time slot for the alarm group.
 - The processing of an alarm within an alarm group may take several fSPB cycles.
 - If the status flag related to a pending alarm event is already set to 1 the alarm event is ignored and the next available pending alarm within the active group is searched.
 - Whenever a pending alarm event is processed, the corresponding status bit is set to 1 by hardware in the AG<x> register. If an internal SMU action is configured the action counter (ACNT) in the **SMU_AFCNT** register is incremented.

9.4.5.4 Alarm Status Registers

Table 9-25 “Handling of Alarm Status” on Page 9-44 specifies the possible software actions on the AG<x> alarm group status registers depending on the SMU State machine state.

Table 9-25 Handling of Alarm Status

SMU State Machine	SW Action	Effect on AG<x>
START	SMU_ASCE(0) command Write Data at AG<x> Address	If (Data [i] == 1) AG<x>[i] = 0 else AG<x>[i] unchanged
START	Write Data at AG<x> Address	If (Data [i] == 1) AG<x>[i] = 1 else AG<x>[i] unchanged
RUN	SMU_ASCE(0) command Write Data at AG<x> Address	If (Data [i] == 1) AG<x>[i] = 0 else AG<x>[i] unchanged
RUN	Write Data at AG<x> Address	No Effect
FAULT	SMU_ASCE(0) command Write Data at AG<x> Address	If (Data [i] == 1) AG<x>[i] = 0 else AG<x>[i] unchanged
FAULT	Write Data at AG<x> Address	No Effect

In the START state software has the possibility to “emulate” the occurrence of input alarm events by writing at an AG<x> address. The resulting set of an AG<x> bit per software can take several hundreds of clock cycles depending on the hardware decoding state and the number of active alarms. Software shall read back the AG<x> register to ensure the completion of the operation if necessary.

9.4.5.5 Alarm Debug Registers

The Alarm debug registers enable the application to improve the diagnosis of the root cause that lead to a malfunction. In that context they may help to implement recovery strategies, if allowed by the application. The **SMU_ADx (x=0-6)** debug registers shall make a snapshot of the **SMU_AGx (x=0-6)** when:

- the action to be executed by the SMU is a reset when the SMU is in the RUN or in the FAULT state
- a SMU state machine (SSM) transition to the FAULT state takes place, either controlled by the SMU hardware or a software command

The **SMU_ADx (x=0-6)** shall only cleared by a power-on reset.

9.4.5.6 Port Emergency Stop

The port emergency stop feature enables forcing a pad into General Purpose Input Mode. The port emergency stop request to the SCU can be activated by any of the following situations:

- a `SMU_ActivatePES()` software command
- an alarm event with `SMU_AG<x>FSP` enabled and `SMU_FSP.PES` enabled
- an alarm event with an internal action configured in `SMU_AG<x>CFx` registers and `SMU_AGC.PES` enabled for that action.

9.4.5.7 Recovery Timer

A recovery timer (RT) is available to enable the monitoring of the duration or internal error handlers activated by an alarm NMI or Interrupt action. In the current SMU implementation two independent instances (RT0 and RT1) are available. The recovery timer duration (identical for all instances) is configured in the register `SMU_RTC`. It is possible to enable or disable each instance, however RT0 is enabled by default as it is required for the operation of the CPU watchdogs (see also “[Watchdog Alarms](#)” on [Page 9-46](#)). In addition to `SMU_RTC` an additional configuration register (`SMU_RTC<n>`) is available per instance to configure the alarm mapping.

The alarm mapping consists of a pair of parameters $\{GID<n>[2:0], ALID<n>[4:0]\}$, where $GID<n>[2:0]$ is a group identifier and $ALID<n>[4:0]$ is the alarm identifier belonging to the group. Four $\{GID[], ALID[]\}$ pairs can be configured per recovery timer instance. It is possible to configure multiple times the same group identifier. If less than four alarms need to be mapped to a recovery timer, the same $\{GID[], ALID[]\}$ shall be configured multiple times.

Note: The use of the recovery timer only makes sense if the internal action is an interrupt or NMI. However no hardware check is done, it is up to software to configure the SMU in the appropriate way.

If a recovery timer is enabled and for any of the $\{GID[], ALID[]\}$ pairs and an alarm event occurs and if an internal action is configured resulting to an internal action (the alarm status shall be cleared) the recovery timer is automatically started by hardware. Such situation is called a recovery timer event. An alarm without internal action shall not start a recovery timer.

Once an recovery timer event is occurs, the recovery timer starts and counts until software stops it with the `SMU_RTStop()`. If the timer expires an internal SMU alarm (Recovery Timer Timeout) is issued. During the time the recovery timer is running any other action that requests the recovery timer is ignored. If such event happens, the bit `RTME` (Recovery Timer Missed Event) is set to ‘1’ by hardware in the `SMU_STS` register. The bit `RTME` can only be cleared by software. The bit `RTS` (Recovery Timer Status) is set to ‘1’ by hardware in the `SMU_STS` register during the time the recovery

Safety Management Unit (SMU)

timer is running: from the timer activation until a SMU_RTStop() is received or the timer expires. The bit RTS is cleared by hardware upon reception of SMU_RTStop() or the timer expires.

If a SMU_RTStop() command is received when the recovery timer is not active, the command returns an error response.

9.4.5.8 Watchdog Alarms

The watchdogs (WDT) timeout alarms require a special processing in order to ensure a correct microcontroller behavior if the watchdogs are not serviced by software or firmware. It shall be ensured that the microcontroller is reset after a pre-warning phase, where software can still perform some critical actions.

- Every Timeout Alarm shall activate an NMI
- Recovery Timer 0 shall be configured to service WDT timeout alarms
- Recovery Timer 0 timeout alarm shall be configured to issue a reset request and activate the Fault Signalling Protocol.

The aforementioned properties are implemented as reset values for the watchdog(s) timeout alarm(s) and for the recovery timer 0.

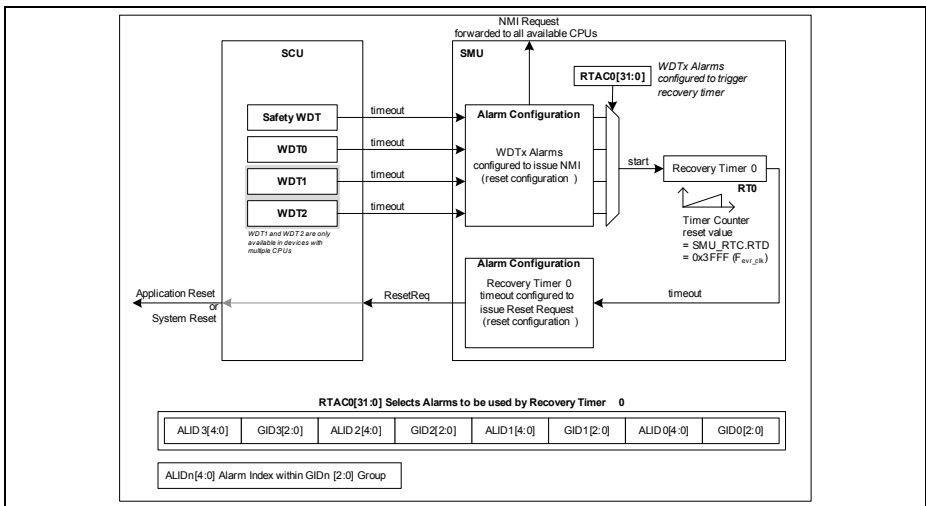


Figure 9-5 Watchdog timeout alarm configuration

The watchdog timeout detection is required from the very first instruction executed by a CPU: the SMU shall process any watchdog timeout alarm during the START state.

Note: If the same behavior is expected for all WDT alarms, it is recommended to use the global WDT timeout alarm that implements a logical OR among all WDT timeout

Safety Management Unit (SMU)

*alarms, thus freeing some {GID[], ALID[]} configuration pairs in **SMU_RTACO** for other purposes.*

9.4.6 SMU Control Interface

The core functionality of the SMU is introduced through its control interface. The control interface defines how the SMU can be controlled by software, as summarized in [Table 9-26 “SMU Commands” on Page 9-48](#). The control interface is directly linked to the SMU state machine (SSM) operation described in [“SMU state machine” on Page 9-50](#) and to the Fault Signaling Protocol (FSP) described in [“Fault Signaling Protocol \(FSP\)” on Page 9-52](#).

The control interface is implemented by the [SMU_CMD](#) register using the CMD and ARG fields. The command completion status is available via the [SMU_STS](#) register.

Table 9-26 SMU Commands

Command	Description	Code
SMU_Start(ARG)	Forces the SSM to go to the RUN state from the START state. Argument ARG shall be set to 0.	0x0
SMU_ActivateFSP(ARG)	Activates the Fault Signaling Protocol. This action is possible in any state of the SSM. Argument ARG shall be set to 0.	0x1
SMU_ReleaseFSP(ARG)	Turns the FSP into the fault free state. In the START state, SMU_ActivateFSP() and SMU_ReleaseFSP() can be called as many times as necessary to perform self tests of every alarm source. Argument ARG shall be set to 0.	0x2
SMU_ActivatePES(ARG)	Triggers the activation of the Port Emergency Stop (PES). The PES is also directly controlled by the SMU when entering the FAULT state. Argument ARG shall be set to 0.	0x3
SMU_RTStop()	Stop the recovery Timer. Argument ARG shall be set to the recovery timer instance available in the product.	0x4

Safety Management Unit (SMU)

Table 9-26 SMU Commands (cont'd)

Command	Description	Code
SMU_ASCE(ARG)	Alarm Status Clear Enable Command. Software shall execute this command prior to clear a AG<n> alarm status bit. This command sets the ASCE bit in the SMU_STS register. Argument ARG shall be set to 0.	0x5
SMU_Alarm(ARG)	Triggers a software based alarm. ARG specifies the alarm index according to the mapping defined in “Alarm Mapping” on Page 9-14 . A software alarm has the same properties as an hardware alarm.	0x6

Note: If the argument does not comply with the specification of the command the command is ignored and returns an error code.

The next table specifies the legal conditions that shall be followed for the execution of the commands. The conditions depend on the SMU state machine (SSM) states (see **“SMU state machine” on Page 9-50**). Any situation not specified leads to an error code.

Table 9-27 SMU commands and valid conditions

Command	SSM state	Other conditions
SMU_Start(ARG)	START	ARG == 0
SMU_ActivateFSP(ARG)	Any	ARG == 0
SMU_ReleaseFSP(ARG)	START	ARG == 0
SMU_ReleaseFSP(ARG)	FAULT	ARG == 0 & SMU_AGC.EFRST == 1 ¹⁾
SMU_ActivatePES(ARG)	Any	ARG == 0
SMU_RTStop(ARG)	Any	ARG >= 0 and ARG <= Number of Recovery Timer Instances and Recovery Timer Enabled
SMU_ASCE(ARG)	Any	ARG == 0
SMU_Alarm(ARG)	RUN, FAULT ²⁾	ARG >= 0

1) See also **“Fault Signaling Protocol (FSP)” on Page 9-52**.

2) In the START state of the SMU input alarms are not processed, therefore software triggered alarms will have no effect.

Safety Management Unit (SMU)

9.4.7 SMU state machine

Figure 9-6 “SMU state machine (SSM): transition conditions and actions” on Page 9-50 and Figure 9-6 “SMU state machine (SSM): transition conditions and actions” on Page 9-50 fully specifies the behavior of the SMU state machine (SSM).

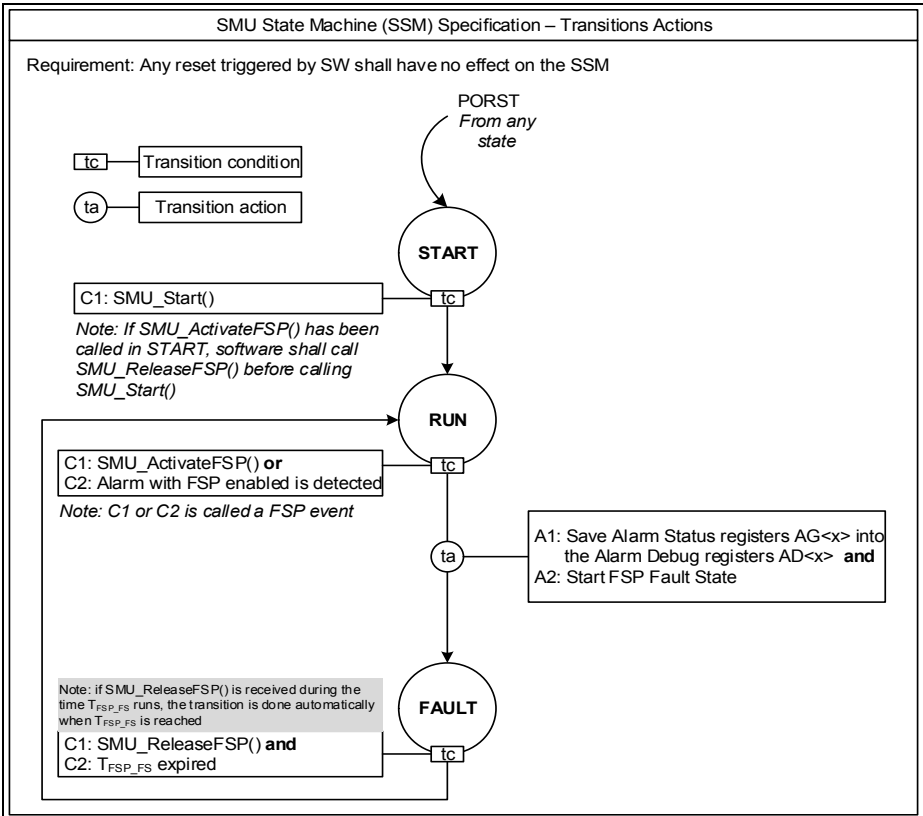


Figure 9-6 SMU state machine (SSM): transition conditions and actions

Safety Management Unit (SMU)

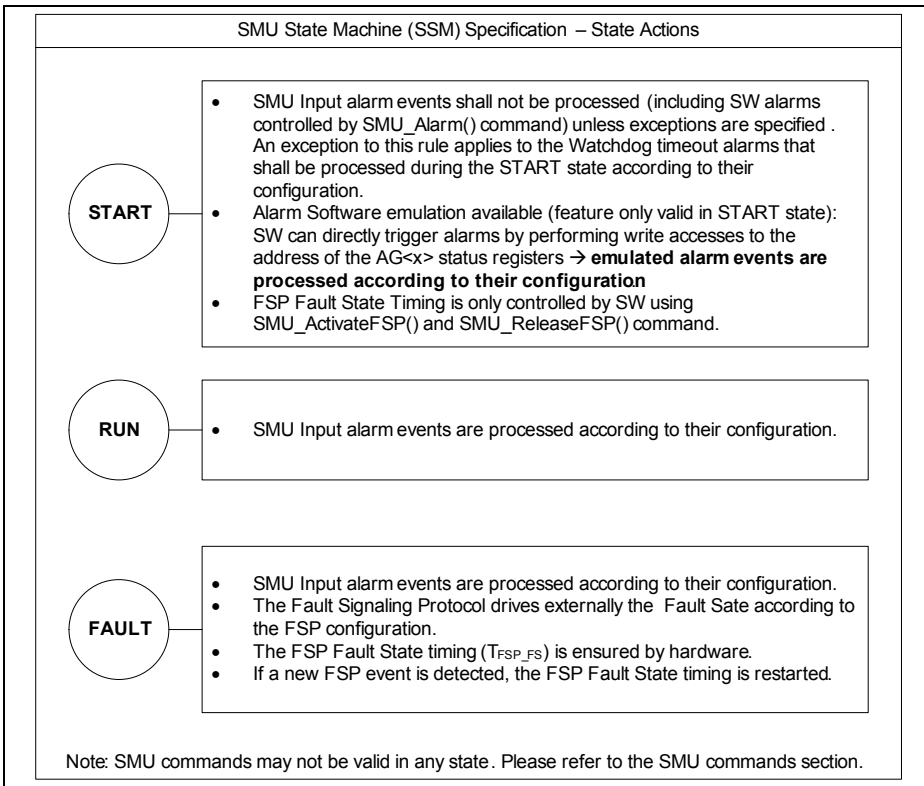


Figure 9-7 SMU state machine: state actions

Fault Counter

The SMU implements a Fault Counter (**SMU_AFCNT**) that counts the number of transitions from the RUN state to the FAULT state. The Fault Counter register is only reset by a power-on-reset.

9.4.8 Fault Signaling Protocol (FSP)

The Fault Signaling Protocol enables the microcontroller to report a critical situation to an external safety controller device in order to control the safe state of the safety system.

9.4.8.1 Introduction

The fault signaling protocol is configured via the **SMU_FSP** command register. The FSP has three states:

- The power-on reset state. After power-on reset the SMU is disconnected from the ports (see **“SMU Integration Guidelines” on Page 9-13**). After power-on reset the SMU FSP output shall be the Fault State.
- The Fault-free State. Whenever the fault-free state is controlled by a timing, the timing will be called T_{FSP_FFS} and is controlled by the **SMU_FSP** register.
- The Fault State. The timing of the fault state is controlled by the **SMU_FSP** register. The minimum active fault state time is called T_{FSP_FS} .

The Fault-free and fault state behavior can be configured with the following protocols:

- Bi-stable protocol (default)
- Time-switching protocol

The FSP can be controlled by:

- Software using the **SMU_ActivateFSP()** and **SMU_ReleaseFSP()** commands using the **SMU_CMD** register
- Hardware based on the **SMU_AG0FSP**, **SMU_AG1FSP**, **SMU_AG2FSP**, **SMU_AG3FSP**, **SMU_AG4FSP**, **SMU_AG5FSP**, **SMU_AG6FSP** configuration registers.
- The logic state of the FSP PAD can be observed by reading the FSP field of the **SMU_STS** register.

Figure 9-8 “Reference clocks for FSP timings” on Page 9-53 specifies the intermediate clocks to generate the T_{FSP_FFS} and T_{FSP_FS} timings.

Safety Management Unit (SMU)

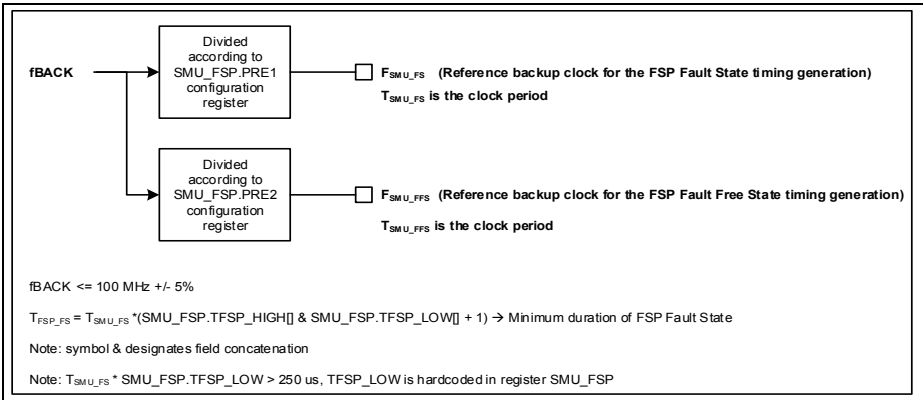


Figure 9-8 Reference clocks for FSP timings

9.4.8.2 Bi-stable fault signaling protocol

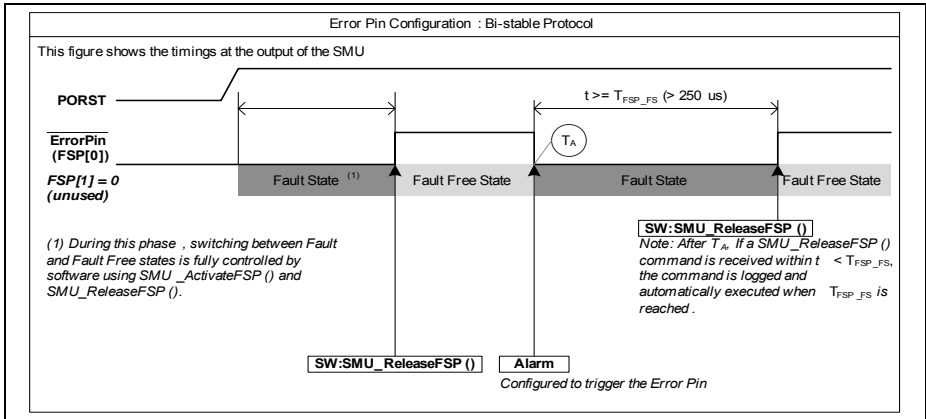


Figure 9-9 Bi-stable fault signaling protocol

Operation

FSP[0] is switched to a static logic level 0 or logic level 1 under software or hardware control.

- During power-on-reset FSP[0] = 0 (fault state)
- After power-on reset FSP[0] stays in the fault state
- FSP[0] must be set to the fault free state per software (`SMU_ReleaseFSP()`).
- Upon detection of an alarm event configured to activate the FSP, FSP[0] goes to the fault state and remains in this state until a `SMU_ReleaseFSP()` command is received and T_{FSP_FS} is satisfied or a Power-on Reset takes place.

9.4.8.3 Time switching protocol

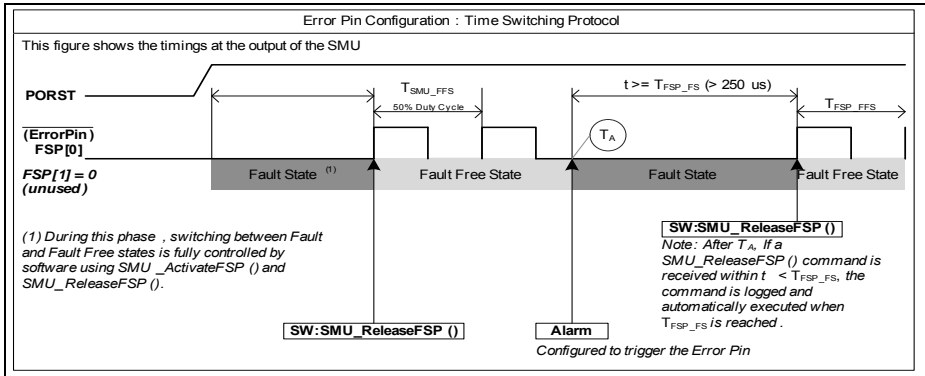


Figure 9-10 Time switching protocol

Operation

FSP[0] is toggled between logic level 0 and logic level 1 with a defined frequency. This frequency modulation protocol is violated when the SMU enters the FAULT state.

- During power-on-reset FSP[0] = 0 (fault state).
- After power-on-reset FSP[0] stays in the fault state.
- FSP[0] must be set to the fault free state per software (**SMU_ReleaseFSP()**).
- In the fault free state, FSP[0] oscillates between logic level 0 and logic level 1 with the frequency configured via the **SMU_FSP** register (see **Figure 9-10**).
- Upon detection of an alarm event configured to activate the FSP, FSP[0] goes immediately to the fault state and remains in this state until a **SMU_ReleaseFSP()** command is received and T_{FSP_FS} is satisfied or a Power-on Reset takes place.

9.4.8.4 FSP Fault State

When an alarm configured to activate the FSP, the SMU automatically switches to the FAULT state. During this time it is also possible for the safety-related software to try to analyze the root cause (when the microcontroller is still operational) and decide about the severity of the error. As the FSP is active for at least T_{FSP_FS} (see **SMU_FSP** register), it is ensured that the safe state of the system is entered through external mechanisms independent from the microcontroller (besides FSP itself).

While in the Fault State, if a new alarm event configured to activate the FSP is received and the T_{FSP_FS} has not yet been reached, the T_{FSP_FS} timing shall be restarted.

During the time T_{FSP_FS} FSP is in the Fault State, software may have concluded that the fault is uncritical and decides to issue a `SMU_ReleaseFSP()` command, notifying the SMU it can return to the RUN state (the run-time of the software error handler is not directly correlated with the T_{FSP_FS} duration and in practice shall be much shorter). This feature shall be used with caution: a microcontroller reset is highly recommended to restart the operation of the safety function when a fault reported by the SMU is assessed to be uncritical. Therefore this feature is per default disabled and shall be configured with the `EFRST` (Enable Fault to Run State Transition) field in the **SMU_AGC** register.

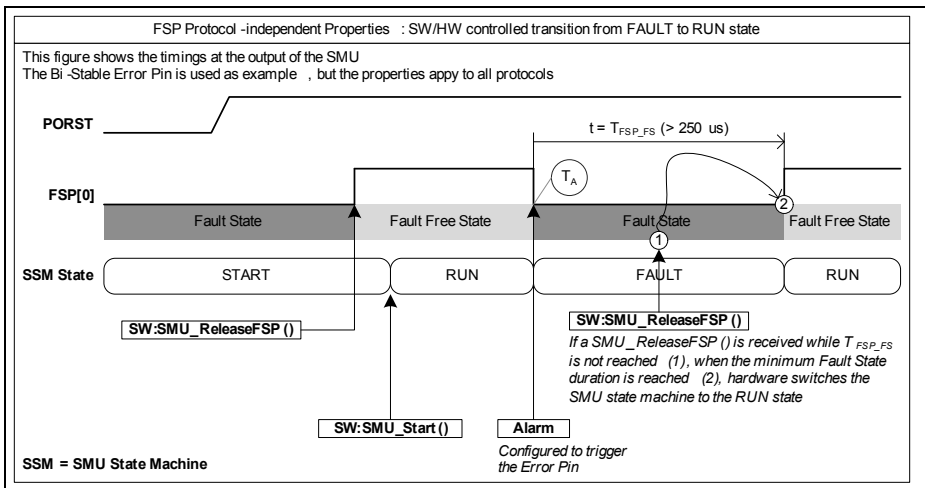


Figure 9-11 FSP: Fault State to Fault Free State Transition

9.4.8.5 FSP and SMU START State

Figure 9-12 “Software Control of FSP during SMU START State” on Page 9-57 shows a typical use case where the FSP transitions between the Fault State and Fault Free State are controlled by software using the SMU_ReleaseFSP() and SMU_ActivateFSP() commands.

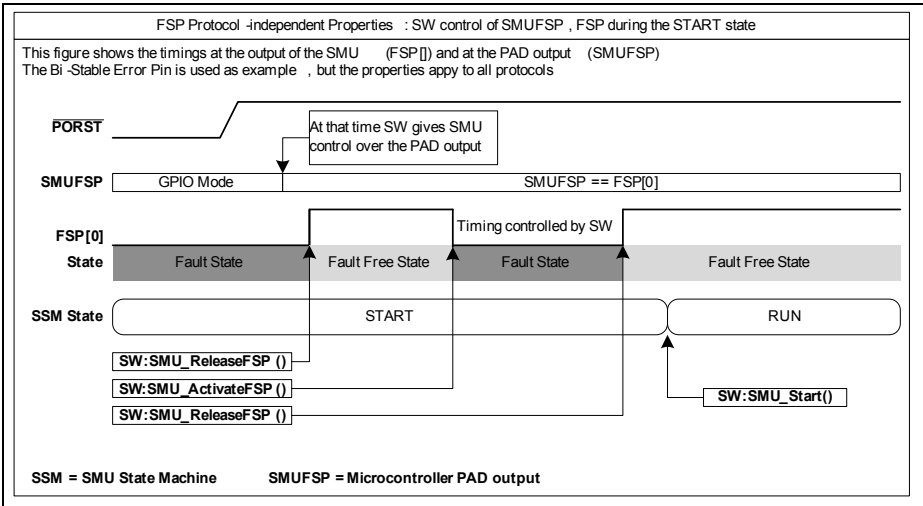


Figure 9-12 Software Control of FSP during SMU START State

Conditions of Use

- Software shall ensure that the FSP is in the Fault Free State before entering the RUN state with the SMU_Start() command.

9.4.9 OCDS Trigger Bus (OTGB) Interface

The SMU concentrates all failure indicator signals (alarms) of the device. By using them as MCDS trace information (failure type) and trace control (stop trace recording), the analysis of a failure's root cause is supported. This is even the case, when the alarm handling within the system includes a PORST, since the content of the trace memory will be still valid after the PORST.

The alarms handled in the SMU are very rare and sporadic. So it's acceptable that they are visible on the Trigger Bus with a small delay and not all in parallel with a cycle accurate timing resolution. The SMU Trigger Set is shown in [Table 9-28](#). It is output on OTGB0 or OTGB1 controlled by the [SMU_OCS](#) register.

Table 9-28 TS16_SMU Trigger Set SMU

Bits	Name	Description
0	AA	Any ALM bit active
1		Reserved
[3:2]	ABI	ALM Byte Index, selecting the byte within the ALM group
[6:4]	AGI	ALM Group Index
7		Reserved
[15:8]	ALM	Selected byte of the ALM group

If no alarm is active, TS16_SMU will be all zero. If alarms are only active within one ALM Byte, TS16_SMU will show statically this byte. If alarms are active in two or more ALM Bytes, TS16_SMU will change between these bytes.

Note that due to the implementation, that all the ALM Bytes are being scanned one per clock cycle, the TS16_SMU ALM Byte information on OTGB0/1 can be delayed by up to 27 clock cycles for the second alarm. TS16_SMU.AA will however become active (and inactive) immediately. The implementation will make sure, that the ALM Byte with the first causing alarm will be traced with MCDS before a reset as safety action clears this information.

9.4.10 Register Properties

9.4.10.1 Register Write Protection

The SMU registers are write protected against illegal master accesses by the master protection mechanism implemented in the System Peripheral Bus interface logic. The registers controlling the master access protection are described in the section “**System Registers description**” on Page 9-68. In addition the SMU registers can only be written if the SafeEinit is enabled. Read accesses have no restriction as there is no side effect specified when reading registers. The SafeEinit has the same properties as the system Einit but it is generated by the safety watchdog. In order to access a SMU register the software must first activate the safety watchdog via a signature protected sequence. The safety watchdog is configured to enable only safety-related software to activate the SafeEinit.

Configuration Lock

In addition to the aforementioned standard features, additional mechanisms are implemented to control and protect the SMU configuration. In order to configure and lock the SMU configuration the following steps shall be followed:

- The SMU configuration is only possible if the CFGLOCK field of the **SMU_KEYS** register is set to 0xBC.
- If the PERLOCK field of the **SMU_KEYS** register is set to 0xFF no further SMU configuration is possible, including the **SMU_KEYS** register itself. No SMU configuration is possible anymore until the PERLOCK field of the **SMU_KEYS** register is reset to 0x00 by an application reset.

The SMU configuration registers controlled by the **SMU_KEYS** register properties are:

- **SMU_FSP**
- **SMU_AGC**
- **SMU_RTC**
- **SMU_RTAC0**
- **SMU_RTAC1**
- **SMU_AG0CFx (x=0-2)**
- **SMU_AG1CFx (x=0-2)**
- **SMU_AG2CFx (x=0-0)**
- **SMU_AG3CFx (x=0-0)**
- **SMU_AG4CFx (x=0-2)**
- **SMU_AG5CFx (x=0-2)**
- **SMU_AG6CFx (x=0-2)**
- **SMU_AG0FSP**
- **SMU_AG1FSP**

Safety Management Unit (SMU)

- **SMU_AG2FSP**
- **SMU_AG3FSP**
- **SMU_AG4FSP**
- **SMU_AG5FSP**
- **SMU_AG6FSP**
- **SMU_PCTL**
- **SMU_RMCTL**

The **SMU_CMD** register is not locked as it is used for run-time hardware/software interaction, it is not a configuration register.

The **SMU_AGx (x=0-6)** registers are not locked as it is possible during run-time to clear the alarm flags by software.

The **SMU_OCS**, **SMU_KRSTCLR**, **SMU_KRST1**, **SMU_KRST0**, **SMU_ACCEN1**, **SMU_ACCEN0** do not belong to the SMU kernel but to the standard bus interface module, therefore they are not controlled by the **SMU_KEYS** register.

The read only registers are not protected by the lock mechanism.

9.4.10.2 Safety Flip-Flops

Safety Flip-Flops are special Flip-Flops that implement an hardware mechanism capable to detect single event effects that may lead to single event upsets (bit flip). The SMU configuration registers that shall be implemented with safety flip-flops are:

- **SMU_FSP**
- **SMU_AGC**
- **SMU_RTC**
- **SMU_KEYS**
- **SMU_PCTL**
- **SMU_RTAC0**
- **SMU_RTAC1**
- **SMU_AG0CFx (x=0-2)**
- **SMU_AG1CFx (x=0-2)**
- **SMU_AG2CFx (x=0-0)**
- **SMU_AG3CFx (x=0-0)**
- **SMU_AG4CFx (x=0-2)**
- **SMU_AG5CFx (x=0-2)**
- **SMU_AG6CFx (x=0-2)**
- **SMU_AG0FSP**
- **SMU_AG1FSP**
- **SMU_AG2FSP**
- **SMU_AG3FSP**
- **SMU_AG4FSP**
- **SMU_AG5FSP**
- **SMU_AG6FSP**

Safety Management Unit (SMU)

Additionally the following SMU functions shall also be implemented with safety flip-flops:

- SMU state machine registers
- Registers implementing the FSP function

Safety Management Unit (SMU)

9.5 SMU Module Registers

Figure 9-13 shows the SMU module register map.

Table 9-29 shows the SMU Address Space

Table 9-30 lists all registers implemented in the SMU.

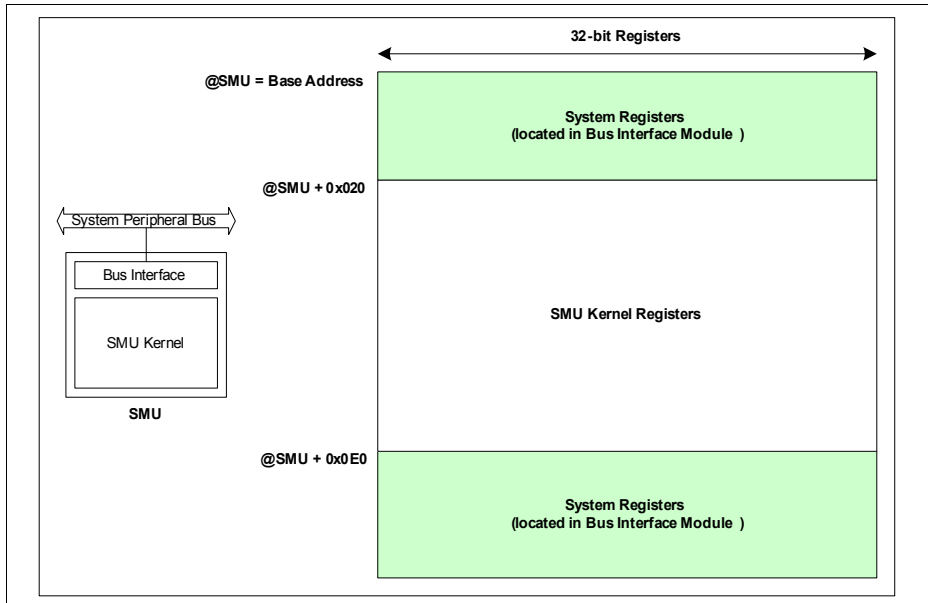


Figure 9-13 SMU Register Map

Table 9-29 Registers Address Space for TC26x B-Step

Module	Base Address	End Address	Note
SMU	F003 6800 _H	F003 6FFF _H	Registers

Table 9-30 Registers Overview

Short Name	Description	Offset Addr	Access Mode		Reset Type	Description See
			Read	Write		
System Registers						
CLC	Clock Control Register	00 _H	U, SV	SV,P	Application Reset	Page 9-68
ID	Module Identifier	08 _H	U, SV	BE	Application Reset	Page 9-70
Kernel Registers:						
CMD	Command interface	20 _H	U, SV	SV,P,32	Application Reset	Page 9-79
STS	Status	24 _H	U, SV	SV,P,32	Application Reset	Page 9-80
FSP	FSP control	28 _H	U, SV	SV,P,SE,32	Power-on Reset	Page 9-83
AGC	Alarm Global Configuration	2C _H	U, SV	SV,P,SE,32	Application Reset	Page 9-85
RTC	Recovery Timer Configuration	30 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-87
KEYS	Register access keys	34 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-88
DBG	Hardware debug	38 _H	U, SV	BE	Power-on Reset	Page 9-89
PCTL	FSP Port Control Register	3C _H	U, SV	SV,P,SE,32	Power-on Reset	Page 9-90
AFCNT	Alarm and Fault Counter Register	40 _H	U, SV	BE	Power-on Reset	Page 9-92
RTAC0	Recovery Timer 0 Alarm Configuration	60 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-93
RTAC1	Recovery Timer 1 Alarm Configuration	64 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-95

Safety Management Unit (SMU)
Table 9-30 Registers Overview (cont'd)

Short Name	Description	Offset Addr	Access Mode		Reset Type	Description See
			Read	Write		
AG0CF0	Alarm configuration	100 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-97
AG0CF1	Alarm configuration	104 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-97
AG0CF2	Alarm configuration	108 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-97
AG1CF0	Alarm configuration	10C _H	U, SV	SV,P,SE,32	Application Reset	Page 9-98
AG1CF1	Alarm configuration	110 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-98
AG1CF2	Alarm configuration	114 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-98
AG2CF0	Alarm configuration	118 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-99
AG2CF1	Alarm configuration	11C _H	U, SV	SV,P,SE,32	Application Reset	Page 9-99
AG2CF2	Alarm configuration	120 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-99
AG3CF0	Alarm configuration	124 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-102
AG3CF1	Alarm configuration	128 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-102
AG3CF2	Alarm Configuration	12C _H	U, SV	SV,P,SE,32	Application Reset	Page 9-102
AG4CF0	Alarm configuration	130 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-105
AG4CF1	Alarm configuration	134 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-105
AG4CF2	Alarm Configuration	138 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-105

Safety Management Unit (SMU)
Table 9-30 Registers Overview (cont'd)

Short Name	Description	Offset Addr	Access Mode		Reset Type	Description See
			Read	Write		
AG5CF0	Alarm configuration	13C _H	U, SV	SV,P,SE,32	Application Reset	Page 9-106
AG5CF1	Alarm configuration	140 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-106
AG5CF2	Alarm Configuration	144 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-106
AG6CF0	Alarm configuration	148 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-107
AG6CF1	Alarm configuration	14C _H	U, SV	SV,P,SE,32	Application Reset	Page 9-107
AG6CF2	Alarm Configuration	150 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-107
AG0FSP	FSP configuration for Alarm Group 0	180 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-108
AG1FSP	FSP configuration for Alarm Group 1	184 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-109
AG2FSP	FSP configuration for Alarm Group 2	188 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-110
AG3FSP	FSP configuration for Alarm Group 3	18C _H	U, SV	SV,P,SE,32	Application Reset	Page 9-111
AG4FSP	FSP configuration for Alarm Group 4	190 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-112
AG5FSP	FSP configuration for Alarm Group 5	194 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-113
AG6FSP	FSP configuration for Alarm Group 6	198 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-114
AG0	Alarm Group 0 Status	1C0 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-115
AG1	Alarm Group 1 Status	1C4 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-115

Safety Management Unit (SMU)
Table 9-30 Registers Overview (cont'd)

Short Name	Description	Offset Addr	Access Mode		Reset Type	Description See
			Read	Write		
AG2	Alarm Group 2 Status	1C8 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-115
AG3	Alarm Group 3 Status	1CC _H	U, SV	SV,P,SE,32	Application Reset	Page 9-115
AG4	Alarm Group 4 Status	1D0 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-115
AG5	Alarm Group 5 Status	1D4 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-115
AG6	Alarm Group 6 Status	1D8 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-115
AD0	Alarm Group 0 Debug	200 _H	U, SV	BE	Power-on Reset	Page 9-116
AD1	Alarm Group 1 Debug	204 _H	U, SV	BE	Power-on Reset	Page 9-116
AD2	Alarm Group 2 Debug	208 _H	U, SV	BE	Power-on Reset	Page 9-116
AD3	Alarm Group 3 Debug	20C _H	U, SV	BE	Power-on Reset	Page 9-116
AD4	Alarm Group 4 Debug	210 _H	U, SV	BE	Power-on Reset	Page 9-116
AD5	Alarm Group 5 Debug	214 _H	U, SV	BE	Power-on Reset	Page 9-116
AD6	Alarm Group 6 Debug	218 _H	U, SV	BE	Power-on Reset	Page 9-116
Special Safety Registers: Safety Flip-Flop Safety Mechanism						
RMCTL	Safety Flip-Flop Test Mode Control	300 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-117
RMEF	Safety Flip-Flop Error Flag	304 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-118
RMSTS	Safety Flip-Flop Self Test Status	308 _H	U, SV	SV,P,SE,32	Application Reset	Page 9-118

Table 9-30 Registers Overview (cont'd)

Short Name	Description	Offset Addr	Access Mode		Reset Type	Description See
			Read	Write		
System Registers						
OCS	OCDS Control and Status Register	7E8 _H	U, SV	SV,P	Debug Reset	Page 9-71
KRSTCLR	Reset Status Clear Register	7EC _H	U, SV	SV,P	Application Reset	Page 9-73
KRST1	Reset Control Register 1	7F0 _H	U, SV	SV,P	Application Reset	Page 9-74
KRST0	Reset Control Register 0	7F4 _H	U, SV	SV,P	Application Reset	Page 9-75
ACCEN1	Access Enable Register	7F8 _H	U, SV	BE	Application Reset	Page 9-77
ACCEN0	Access Enable Register	7FC _H	U, SV	SV,SE	Application Reset	Page 9-78

9.5.1 System Registers description

SMU Clock Control Register (CLC)

The Clock Control Register allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The SMU shall be enabled per default

SMU_CLC

Clock Control Register

(00_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS	FDIS	DISS	DISR
r												rw	rw	rh	rw

Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module. 0 _B Module disable is not requested 1 _B Module disable is requested
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module. 0 _B Module is enabled 1 _B Module is disabled
FDIS	2	rw	Force Disable The feature controlled by this field is not used. Read as 0; should be written with 0.
EDIS	3	rw	Sleep Mode Enable Control Used to control module's sleep mode. Sleep Mode is not supported safety applications. During the process of entering and resuming from sleep mode the intended processing of alarm events is not guaranteed.

Safety Management Unit (SMU)

Field	Bits	Type	Description
0	[31:4]	r	Reserved Read as 0; should be written with 0.

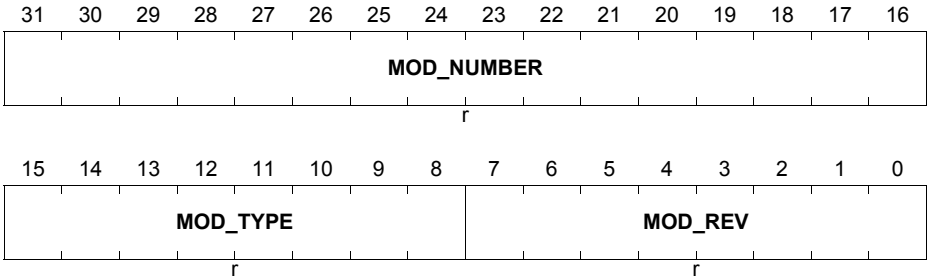
Note: The other features controlled by the CLC register are not supported by the SMU.

Safety Management Unit (SMU)

SMU Module Identification Register

SMU_ID

Module Identification Register (08_H) Reset Value: 0089 C001_H



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MOD_TYPE	[15:8]	r	Module Type The bit field is set to C0 _H which defines the module as a 32-bit module.
MOD_NUMBER	[31:16]	r	Module Number Value This bit field defines a module identification number. The value for the SMU module is 0089 _H .

Safety Management Unit (SMU)

SMU OCDS Control Register. This register is implemented in the BPI.

The OCDS Control and Status (OCS) register is reset by Debug Reset.

The OCS register includes the module related control bits for the OCDS Trigger Bus (OTGB).

The register can only be written and the OCS control register bits are only effective while the OCDS is enabled (OCDS enable = '1'). While OCDS is not enabled, OCS reset values/modes are effective.

SMU_OCS
OCDS Control and Status
(7E8_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SUS STA	SUS_P	SUS				0								
r	rh	w	rw				r								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TG_P	TGB	TGS	
r												w	rw	rw	

Field	Bits	Type	Description
TGS	[1:0]	rw	Trigger Set for OTGB0/1 0 _H No Trigger Set output 1 _H TS16_SMU others , reserved
TGB	2	rw	OTGB0/1 Bus Select 0 _B Trigger Set is output on OTGB0 1 _B Trigger Set is output on OTGB1
TG_P	3	w	TGS, TGB Write Protection TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
SUS	[27:24]	rw	OCDS Suspend Control No effect. SMU will not suspend. Read as 0; must be written with 0.
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.

Safety Management Unit (SMU)

Field	Bits	Type	Description
SUSSTA	29	rh	Suspend State Read as 0; must be written with 0.
0	[23:4], [31:30]	r	Reserved Read as 0; must be written with 0.

Safety Management Unit (SMU)

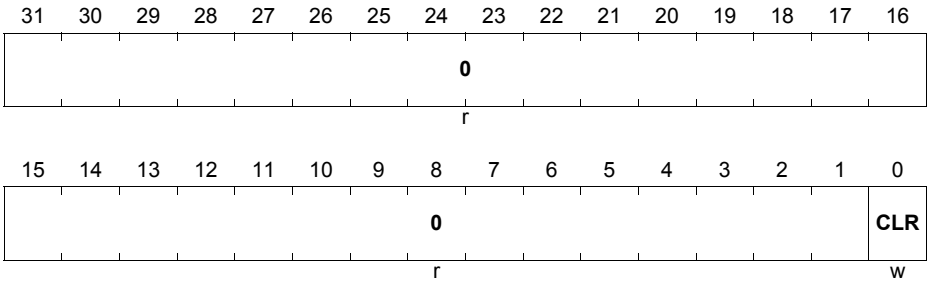
The Kernel Reset Register Clear register is used to clear the Kernel Reset Status bit (<>_KRST0.RSTSTAT).

SMU_KRSTCLR

SMU Reset Status Clear Register

(7EC_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Safety Management Unit (SMU)

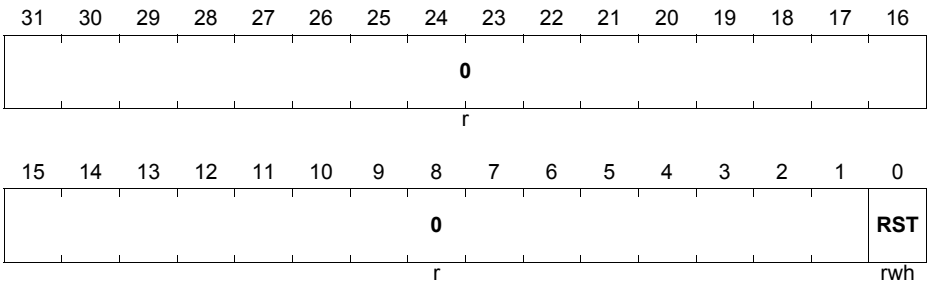
The Kernel Reset Register 1 is used to reset the related module kernel. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers (<>_KRSTx1.RST and <>_KRSTx0.RST) related to the module kernel that should be reset (kernel 0 or kernel 1). The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

SMU_KRST1

SMU Reset Register 1

(7F0_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set.</p> <p>0_B No kernel reset was requested 1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p>
0	[31:1]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

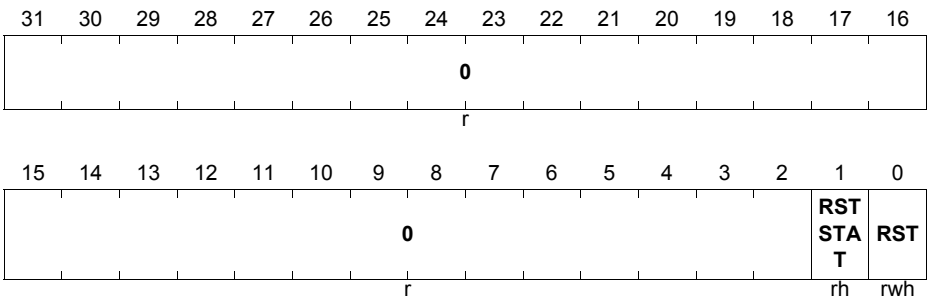
Safety Management Unit (SMU)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers related to the module kernel that should be reset (kernel 0 or kernel 1). In order support modules with two kernel the BPI_FPI provides two set of kernel reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the related KRSTCLR_x.CLR register bit.

SMU_KRST0

SMU Reset Register 0 (7F4_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set.</p> <p>0_B No kernel reset was requested</p> <p>1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p>

Safety Management Unit (SMU)

Field	Bits	Type	Description
RSTSTAT	1	rh	<p>Kernel Reset Status</p> <p>This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits.</p> <p>0_B No kernel reset was executed 1_B Kernel reset was executed</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.</p>
0	[31:2]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Safety Management Unit (SMU)

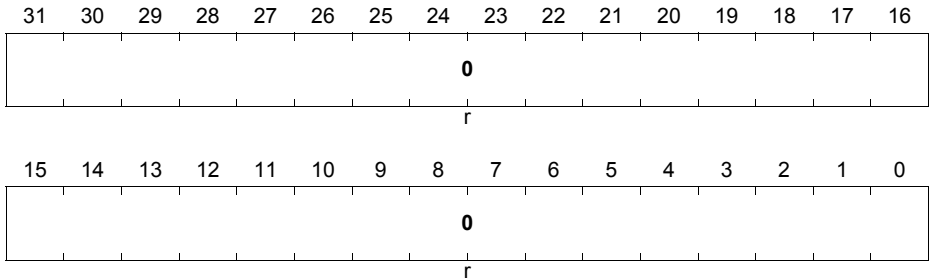
The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

SMU_ACCEN1

SMU Access Enable Register 1

(7F8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Safety Management Unit (SMU)

The Access Enable Register 0 controls write access for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

SMU_ACCEN0
SMU Access Enable Register 0
(7FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN3	EN3	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN1	EN1	EN1	EN1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN1	EN1	EN1	EN1	EN1	EN1	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
5	4	3	2	1	0										
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

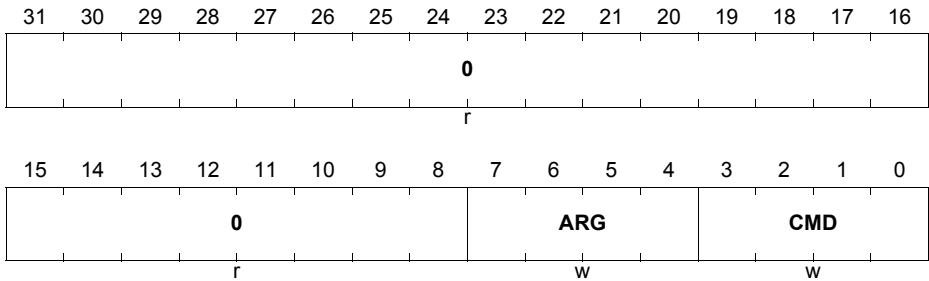
Note: The mapping with the on-chip master-capable modules is described in the bus chapters of the TC26x B-Step microcontroller.

9.5.2 SMU Configuration Registers

SMU Command Register.

SMU_CMD

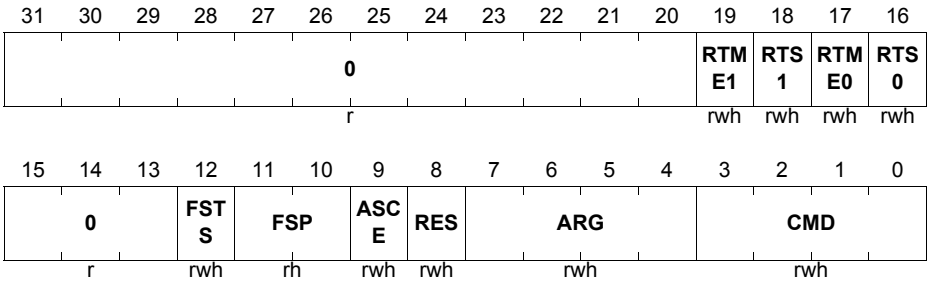
Command Register (20_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
CMD	[3:0]	w	Implements the SMU Command Interface. See Table 9-26 “SMU Commands” on Page 9-48 for the command encoding. Read as 0.
ARG	[7:4]	w	Implements the SMU Command Interface. Argument to be used with the command. See Table 9-26 “SMU Commands” on Page 9-48 for the argument encoding. Read as 0.
0	[31:8]	r	Reserved Read as 0; should be written with 0

Safety Management Unit (SMU)

SMU Status Register.

SMU_STS
Status Register (24_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
CMD	[3:0]	rwh	Last command received Same encoding as CMD field of SMU_CMD register
ARG	[7:4]	rwh	Last command argument received Same encoding as ARG field of SMU_CMD register
RES	8	rwh	Result of last received command 0 _B Command was successful 1 _B Command failed
ASCE	9	rwh	Alarm Status Clear Enable This bit controls if a status flag set in an AG<x> register upon detection of an alarm event can be cleared by software or not. When ASCE is enabled software shall write a 1 to the bit position in AG<x> to clear the bit (W1C). When a W1C action takes place the ASCE bit is automatically cleared to 0 by hardware and software shall set the ASCE bit again by using the SMU_ASCE() command. 0 _B Alarm status bits AG<x> can not be cleared 1 _B Alarm status bits AG<x> can be cleared.

Safety Management Unit (SMU)

Field	Bits	Type	Description
FSP	[11:10]	rh	Fault Signaling Protocol status FSP[0] = FSP_STS[0] input signal FSP[1] = FSP_STS[1] input signal This field is updated by hardware every clock cycle, therefore a software clear on write is not meaningful for this field.
FSTS	12	rwh	Fault State Timing Status This bit indicates if the minimum timing duration of the FSP fault state has been reached or not. The bit is cleared by hardware when the fault state is entered. 0 _B Minimum timing duration not reached 1 _B Minimum timing duration reached
RTS0	16	rwh	Recovery Timer 0 Status See “Recovery Timer” on Page 9-45 for the usage of this field. 0 _B Recovery Timer not running 1 _B Recovery Timer running
RTME0	17	rwh	Recovery Timer 0 Missed Event See “Recovery Timer” on Page 9-45 for the usage of this field. 0 _B Recovery Timer event not detected 1 _B Recovery Timer event detected
RTS1	18	rwh	Recovery Timer 1 Status See “Recovery Timer” on Page 9-45 for the usage of this field. 0 _B Recovery Timer not running 1 _B Recovery Timer running
RTME1	19	rwh	Recovery Timer 1 Missed Event See “Recovery Timer” on Page 9-45 for the usage of this field. 0 _B Recovery Timer event not detected 1 _B Recovery Timer event detected
0	[31:20], [15:13]	r	Reserved Read as 0; should be written with 0.

Safety Management Unit (SMU)

Note: A write to this register (regardless of the data written) clears all the fields with the wrh type. If on the same cycle a software write event and hardware event is detected the hardware action wins.

Safety Management Unit (SMU)

SMU Fault Signalling Timing Configuration.

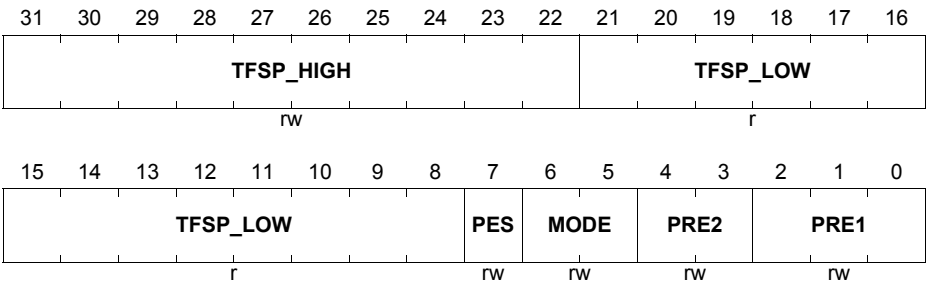
This register controls the timing of the fault signaling protocol.

SMU_FSP

Fault Signaling Protocol

(28_H)

Reset Value: 003F FF00_H



Field	Bits	Type	Description
PRE1	[2:0]	rw	<p>Prescaler1</p> <p>Dividing factor to apply to the reference clock fBACK. It is assumed that the maximal value for fBACK is 100 Mhz with a precision of 5%. The divided clock is used as reference to generate the timing of the fault signaling protocol fault state. The frequency of the divided clock (called F_{SMU_FS}) is defined as follows:</p> <p>0_H reference clock frequency divided by 2 1_H reference clock frequency divided by 4 2_H reference clock frequency divided by 8 3_H reference clock frequency divided by 16 4_H reference clock frequency divided by 32 5_H reference clock frequency divided by 64 6_H reference clock frequency divided by 128 7_H reference clock frequency divided by 256</p>

Safety Management Unit (SMU)

Field	Bits	Type	Description
PRE2	[4:3]	rw	Prescaler2 Dividing factor to apply to the reference clock fBACK in order to generate the timing of the fault free state for the time switching modes of the fault signaling protocol. The frequency of the divided clock (called F_{SMU_FFS}) is defined as follows: 0_H reference clock frequency divided by 512 1_H reference clock frequency divided by 1024 2_H reference clock frequency divided by 2048 3_H reference clock frequency divided by 4096
MODE	[6:5]	rw	Fault Signaling Protocol configuration 0_H Bi-stable protocol 1_H Reserved 2_H Time switching protocol 3_H Reserved
PES	7	rw	Port Emergency Stop (PES) When this bit is set a Port Emergency Stop is automatically requested when an alarm event configured to start the Fault Signalling Protocol is detected. 0_B Port Emergency Stop disabled 1_B Port Emergency Stop enabled
TFSP_LOW	[21:8]	r	Specification of the FSP fault state duration $T_{FSP_FS} = TFSP_HIGH \& TPSP_LOW$. Note: symbol & designates a concatenation TFSP_LOW shall be specified as a number of FSMU_FS ticks. TFSP_LOW is defined so that the minimum duration is greater than 250 us. It can not be changed by software. Refer to Figure 9-8 “Reference clocks for FSP timings” on Page 9-53
TFSP_HIGH	[31:22]	rw	Specification the FSP fault state duration $T_{FSP_FS} = TFSP_HIGH \& TPSP_LOW$. Note: symbol & designates a concatenation TFSP_HIGH shall be specified as a number of FSMU_FS ticks. TFSP_HIGH and PRE1 shall enable to configure a fault state duration of 500 ms.

Safety Management Unit (SMU)

SMU Alarm Global Configuration.

This register controls some properties related to the behavior of the SMU to alarm.

SMU_AGC

Alarm Global Configuration

(2C_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		EFR ST	PES				0				ICS				
r		rw		rw				r				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		IGCS2				0	IGCS1		0	IGCS0					
r		rw				r	rw		r	rw					

Field	Bits	Type	Description
IGCS0	[2:0]	rw	Interrupt Generation Configuration Set 0 Defines the output value of the interrupt request vector when the alarm configuration flag selects the interrupt configuration set 0. Enables to issue an interrupt request to several CPUs: see “Interfaces to the Interrupt Router” on Page 9-8.
IGCS1	[6:4]	rw	Interrupt Generation Configuration Set 1 Defines the output value of the interrupt request vector when the alarm configuration flag selects the interrupt configuration set 1. Enables to issue an interrupt request to several CPUs: see “Interfaces to the Interrupt Router” on Page 9-8.
IGCS2	[10:8]	rw	Interrupt Generation Configuration Set 2 Defines the output value of the interrupt request vector when the alarm configuration flag selects the interrupt configuration set 2. Enables to issue an interrupt request to several CPUs: see “Interfaces to the Interrupt Router” on Page 9-8.

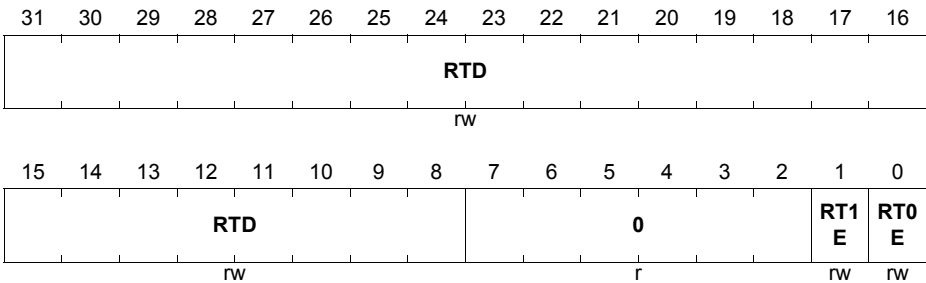
Safety Management Unit (SMU)

Field	Bits	Type	Description
ICS	[18:16]	rw	Idle Configuration Set Defines the output value of the idle request vector when the alarm configuration flag selects the Idle Configuration Set. Enables to issue an idle request to several CPUs if required. More complex idle scenarios can be handled by using software interrupts.
PES	[28:24]	rw	Port Emergency Stop This field enables control of the Port Emergency Stop (PES) feature independently for each internal action. When an action is triggered and if the corresponding bit (as defined below) is set, the hardware triggers automatically a port emergency stop request. Each bit of PES is allocated to an action as follows: 1 _H SMU_IGCS0 activates PES 2 _H SMU_IGCS1 activates PES 4 _H SMU_iGCS2 activates PES 8 _H SMU_NMI activates PES 10 _H SMU_IDLE activates PES
EFRST	29	rw	Enable FAULT to RUN State Transition See “FSP Fault State” on Page 9-56 chapter for the usage of this field.
0	[31:30], [23:19], [15:11], 7, 3	r	Reserved Read as 0; should be written with 0

Safety Management Unit (SMU)

SMU Recovery Timer Duration Configuration.

This register controls the timing duration of the recovery timer.

SMU_RTC
Recovery Timer Configuration
(30_H)
Reset Value: 003F FF01_H


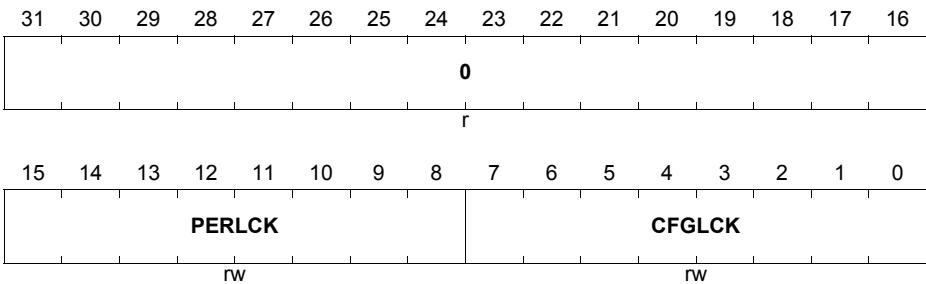
Field	Bits	Type	Description
RTD	[31:8]	rw	Recovery Timer Duration This field specifies the maximum duration of the recovery timer. When the timer counter reaches the programmed value, the internal alarm <code>rt_timeout</code> is issued. The timer is stopped by a <code>SMU_RTStop()</code> command before the recovery timer. RTD shall be specified as a number of the F_{SMU_FS} clock ticks.
RT1E	1	rw	RT1 Enable Bit 0 _B Recovery Timer 1 is disabled 1 _B Recovery Timer 1 is enabled
RT0E	0	rw	RT0 Enable Bit 0 _B Recovery Timer 0 is disabled 1 _B Recovery Timer 0 is enabled
0	[7:2]	r	Reserved Read as 0; should be written with 0

Safety Management Unit (SMU)

SMU Keys Register.

SMU_KEYS
Key Register

 (34_H)

 Reset Value: 0000 0000_H


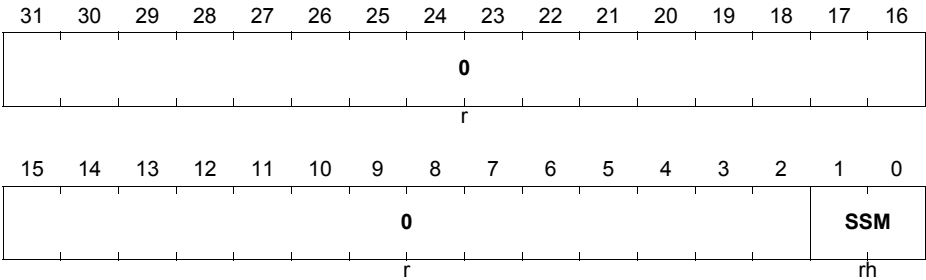
Field	Bits	Type	Description
CFGLCK	[7:0]	rw	Configuration Lock The SMU configuration is only possible if this field is set to 0xBC. Refer to “Register Properties” on Page 9-59 for the list of registers controlled by this field.
PERLCK	[15:8]	rw	Permanent Lock If this field is set to 0xFF, no further configuration of the SMU is possible. Refer to “Register Properties” on Page 9-59 for the list of registers controlled by this field.
0	[31:16]	r	Reserved Read as 0; shall be written with 0

Safety Management Unit (SMU)

SMU Debug Register.

This register enables to observe some internal states of the SMU hardware. Definition will be completed based on design information.

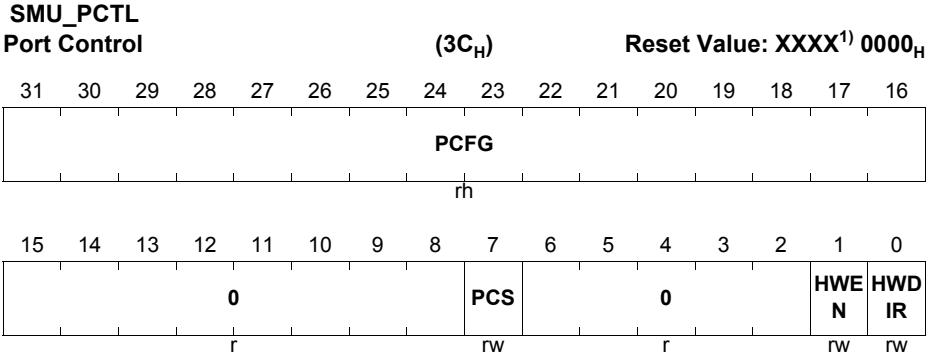
SMU_DBG
Debug Register (38_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
SSM	[1:0]	rh	Running state of the SMU State Machine 0 _H START state 1 _H RUN state 2 _H FAULT state 3 _H unspecified state
0	[31:2]	r	Reserved Read as 0; should be written with 0

Safety Management Unit (SMU)

SMU register controlling the connectivity with the Ports.



1) The PCFG reset value depends on several conditions. The PFCG value is only relevant after user software configuration.

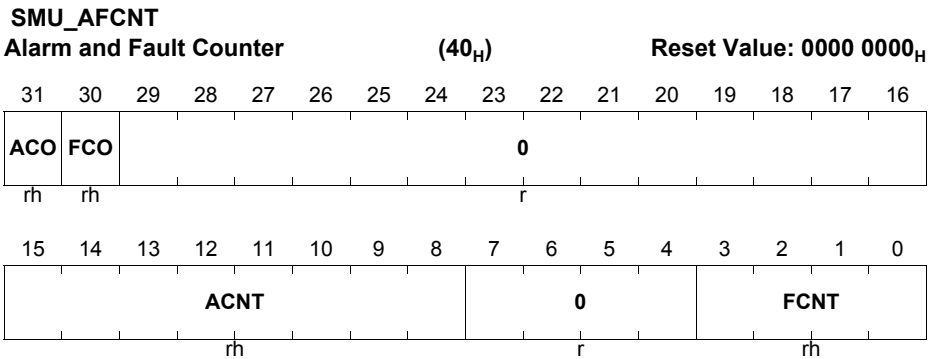
Field	Bits	Type	Description
HWDIR	0	rw	<p>Port Direction.</p> <p>This bit directly controls the value of the FSP_DIR SMU output signal. Also refer to the General Purpose I/O Ports chapter for the HW_DIR signal specification.</p> <p>0_B sets the PAD to input state 1_B sets the PAD to output state</p>
HWEN	1	rw	<p>Port Enable.</p> <p>This bit directly controls the value of the FSP_EN SMU output signal. When set to 1 the port output is directly driven by SMU FSP signal (Error Pin). Also refer to the General Purpose I/O Ports chapter for the HW_EN signal specification.</p>

Safety Management Unit (SMU)

Field	Bits	Type	Description
PCS	7	rw	<p>PAD Configuration Select</p> <p>This bit controls the latching of the SMU FSP (Error Pin) PAD configuration signals to ensure that upon an application reset or system reset the SMU FSP (Error Pin) PAD configuration is not affected. This field is only reset by power-on reset.</p> <p>0_B The PAD configuration is controlled by the PORT registers.</p> <p>1_B The PAD configuration is controlled by the SMU.</p> <ul style="list-style-type: none"> • Only with the first transition from 0 to 1 of this field the SMU FSP is operational. Any further configuration change in this bit field has no effect to the hardware. • The fields HWDIR, HWEN and PCS shall be configured with a single software write command. Configuring each bit-field separately may lead to configuration inconsistencies. Refer to “Interface to the Ports (Error Pin)” on Page 9-8 for the overview of the SMU FSP (Error Pin) connectivity.
PCFG	[31:16]	rh	<p>PAD Configuration</p> <p>This field contains the FSP PAD configuration input signals.</p> <p>The PAD configuration related to FSP[0] is provided by PCFG[7:0]</p> <p>The upper-bits PCFG[15:8] are not used.</p>
0	[15:8], [6:2]	r	<p>Reserved</p> <p>Read as 0; should be written with 0</p>

Safety Management Unit (SMU)

SMU Alarm and Fault Counter Register.



Field	Bits	Type	Description
FCNT	[3:0]	rh	Fault Counter. This field is incremented by hardware when the SMU state machine goes from the RUN state to the FAULT state (see Figure 9.4.7 “SMU state machine” on Page 9-50). The counter value holds if the maximum value is reached.
ACNT	[15:8]	rh	Alarm Counter. This field is incremented by hardware when the SMU processes an internal action related to an alarm event (see Figure 9.4.5.3 “Alarm operation” on Page 9-43). The counter value holds if the maximum value is reached.
FCO	30	rh	Fault Counter Overflow. This bit is set by hardware if the FCNT counter reached the maximum value and an increment condition is present.
ACO	31	rh	Alarm Counter Overflow. This bit is set by hardware if the ACNT counter reached the maximum value and an increment condition is present.
0	[29:16], [7:4]	r	Reserved Read as 0; should be written with 0.

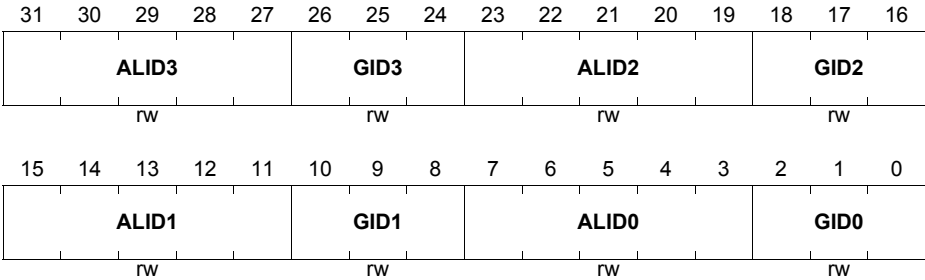
Safety Management Unit (SMU)

SMU Recovery Timer 0 Alarm Configuration.

SMU_RTAC0

Recovery Timer Alarm Configuration (60_H)

Reset Value: A39B 938B_H



Field	Bits	Type	Description
GID0	[2:0]	rw	Group Index 0. This field enables to specify if an alarm from this alarm group can use the recovery timer 0. The functionality of this field is described in “Recovery Timer” on Page 9-45.
ALID0	[7:3]	rw	Alarm Identifier 0. This field specifies the Alarm Index related to the Group Index specified in GID0.
GID1	[10:8]	rw	Group Index 1. This field enables to specify if an alarm from this alarm group can use the recovery timer 0. The functionality of this field is described in “Recovery Timer” on Page 9-45.
ALID1	[15:11]	rw	Alarm Identifier 1. This field specifies the Alarm Index related to the Group Index specified in GID1.
GID2	[18:16]	rw	Group Index 2. This field enables to specify if an alarm from this alarm group can use the recovery timer 0. The functionality of this field is described in “Recovery Timer” on Page 9-45.

Safety Management Unit (SMU)

Field	Bits	Type	Description
ALID2	[23:19]	rw	Alarm Identifier 2. This field specifies the Alarm Index related to the Group Index specified in GID2.
GID3	[26:24]	rw	Group Index 3. This field enables to specify if an alarm from this alarm group can use the recovery timer 0. The functionality of this field is described in “Recovery Timer” on Page 9-45.
ALID3	[31:27]	rw	Alarm Identifier 3. This field specifies the Alarm Index related to the Group Index specified in GID3.

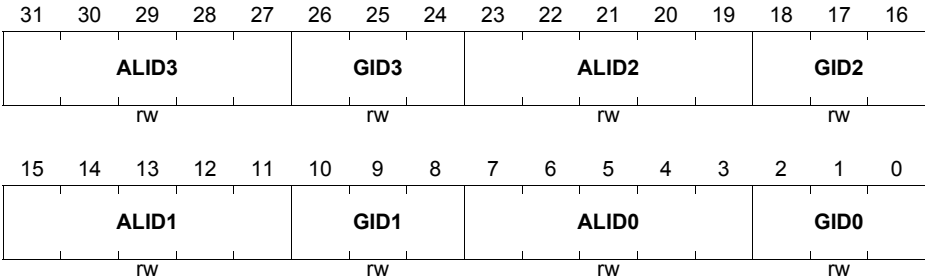
Note: It is possible to configure multiple times the same group identifier in GID0/1/2/3 fields. The reset value maps the watchdog timeout alarms to the recovery timer 0. The number of available watchdogs is product dependent.

Note: The reset values corresponds to Alarms 17, 18, 19, 20 from Alarm Group 3

Safety Management Unit (SMU)

SMU Recovery Timer 1 Alarm Configuration.

SMU_RTAC1
Recovery Timer Alarm Configuration (64_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
GID0	[2:0]	rw	Group Index 0. This field enables to specify if an alarm from this alarm group can use the recovery timer 1. The functionality of this field is described in “Recovery Timer” on Page 9-45.
ALID0	[7:3]	rw	Alarm Identifier 0. This field specifies the Alarm Index related to the Group Index specified in GID0.
GID1	[10:8]	rw	Group Index 1. This field enables to specify if an alarm from this alarm group can use the recovery timer 1. The functionality of this field is described in “Recovery Timer” on Page 9-45.
ALID1	[15:11]	rw	Alarm Identifier 1. This field specifies the Alarm Index related to the Group Index specified in GID1.
GID2	[18:16]	rw	Group Index 2. This field enables to specify if an alarm from this alarm group can use the recovery timer 1. The functionality of this field is described in “Recovery Timer” on Page 9-45.

Safety Management Unit (SMU)

Field	Bits	Type	Description
ALID2	[23:19]	rw	Alarm Identifier 2. This field specifies the Alarm Index related to the Group Index specified in GID2.
GID3	[26:24]	rw	Group Index 3. This field enables to specify if an alarm from this alarm group can use the recovery timer 1. The functionality of this field is described in “Recovery Timer” on Page 9-45.
ALID3	[31:27]	rw	Alarm Identifier 3. This field specifies the Alarm Index related to the Group Index specified in GID3.

Note: It is possible to configure multiple times the same group identifier in GID0/1/2/3 fields.

9.5.3 SMU Alarm Configuration Registers

SMU Alarm Configuration Registers related to Alarm Group 0.

SMU_AG0CFx (x=0-2)

Alarm Configuration Register (100_H + x*04_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFn (n = 0-31)	n	rw	<p>Configuration flag x (x=0-2) for alarm n belonging to alarm group 0.</p> <p>The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU when a fault state is reported by the alarm n belonging to this group (see “Alarm Configuration” on Page 9-41).</p> <p><i>Note: If the alarm is specified as reserved in “Alarm Signals” on Page 9-24, software shall configure the behavior to “No Action”.</i></p> <p>0_B Configuration flag x (x=0-2) is set to 0 1_B Configuration flag x (x=0-2) is set to 1</p>

Safety Management Unit (SMU)

SMU Alarm Configuration Registers related to Alarm Group 1.

SMU_AG1CFx (x=0-2)

Alarm Configuration Register (10C_H + x*04_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFn (n = 0-31)	n	rw	<p>Configuration flag x (x=0-2) for alarm n belonging to alarm group 1.</p> <p>The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU when a fault state is reported by the alarm n belonging to this group (see “Alarm Configuration” on Page 9-41).</p> <p><i>Note: If the alarm is specified as reserved in “Alarm Signals” on Page 9-24, software shall configure the behavior to “No Action”.</i></p> <p>0_B Configuration flag x (x=0-2) is set to 0 1_B Configuration flag x (x=0-2) is set to 1</p>

Safety Management Unit (SMU)

SMU Alarm Configuration Registers related to Alarm Group 2.

SMU_AG2CFx (x=0-0)
Alarm Configuration Register ($118_H + x \cdot 04_H$) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFn (n = 0-31)	n	rw	<p>Configuration flag x (x=0-2) for alarm n belonging to alarm group 2.</p> <p>The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU when a fault state is reported by the alarm n belonging to this group (see “Alarm Configuration” on Page 9-41).</p> <p><i>Note: If the alarm is specified as reserved in “Alarm Signals” on Page 9-24, software shall configure the behavior to “No Action”.</i></p> <p>0_B Configuration flag x (x=0-2) is set to 0 1_B Configuration flag x (x=0-2) is set to 1</p>

Safety Management Unit (SMU)

SMU Alarm Configuration Registers related to Alarm Group 2.

SMU_AG2CFx (x=1-1)
Alarm Configuration Register ($118_H + x \cdot 04_H$) **Reset Value: 2000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFn (n = 0-31)	n	rw	<p>Configuration flag x (x=0-2) for alarm n belonging to alarm group 2.</p> <p>The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU when a fault state is reported by the alarm n belonging to this group (see “Alarm Configuration” on Page 9-41).</p> <p><i>Note: If the alarm is specified as reserved in “Alarm Signals” on Page 9-24, software shall configure the behavior to “No Action”.</i></p> <p>0_B Configuration flag x (x=0-2) is set to 0 1_B Configuration flag x (x=0-2) is set to 1</p>

Safety Management Unit (SMU)

SMU Alarm Configuration Registers related to Alarm Group 2.

SMU_AG2CFx (x=2-2)

Alarm Configuration Register (118_H + x*04_H) Reset Value: 2000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFn (n = 0-31)	n	rw	<p>Configuration flag x (x=0-2) for alarm n belonging to alarm group 2.</p> <p>The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU when a fault state is reported by the alarm n belonging to this group (see “Alarm Configuration” on Page 9-41).</p> <p><i>Note: If the alarm is specified as reserved in “Alarm Signals” on Page 9-24, software shall configure the behavior to “No Action”.</i></p> <p>0_B Configuration flag x (x=0-2) is set to 0 1_B Configuration flag x (x=0-2) is set to 1</p>

Safety Management Unit (SMU)

SMU Alarm Configuration Registers related to Alarm Group 3.

SMU_AG3CFx (x=0-0)
Alarm Configuration Register ($124_H + x \cdot 04_H$) **Reset Value: 001E 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
CFn (n = 0-31)	n	rW	<p>Configuration flag x (x=0-0) for alarm n belonging to alarm group 3.</p> <p>The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU when a fault state is reported by the alarm n belonging to this group (see “Alarm Configuration” on Page 9-41).</p> <p><i>Note: If the alarm is specified as reserved in “Alarm Signals” on Page 9-24, software shall configure the behavior to “No Action”.</i></p> <p>0_B Configuration flag x (x=0-2) is set to 0 1_B Configuration flag x (x=0-2) is set to 1</p>

Note: The reset value specifies the NMI internal action for the watchdog timeout alarms. The reset value is the same regardless of the number of watchdogs present in the device.

Safety Management Unit (SMU)

SMU Alarm Configuration Registers related to Alarm Group 3.

SMU_AG3CFx (x=1-1)
Alarm Configuration Register $(124_H + x \cdot 04_H)$ **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
CFn (n = 0-31)	n	rW	<p>Configuration flag x (x=1-1) for alarm n belonging to alarm group 3.</p> <p>The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU when a fault state is reported by the alarm n belonging to this group (see “Alarm Configuration” on Page 9-41).</p> <p><i>Note: If the alarm is specified as reserved in “Alarm Signals” on Page 9-24, software shall configure the behavior to “No Action”.</i></p> <p>0_B Configuration flag x (x=0-2) is set to 0 1_B Configuration flag x (x=0-2) is set to 1</p>

Note: The reset value specifies the NMI internal action for the watchdog timeout alarms. The reset value is the same regardless of the number of watchdogs present in the device.

Safety Management Unit (SMU)

SMU Alarm Configuration Registers related to Alarm Group 3.

SMU_AG3CFx (x=2-2)
Alarm Configuration Register $(124_H + x \cdot 04_H)$ **Reset Value: 001E 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
CFn (n = 0-31)	n	rW	<p>Configuration flag x (x=2) for alarm n belonging to alarm group 3.</p> <p>The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU when a fault state is reported by the alarm n belonging to this group (see “Alarm Configuration” on Page 9-41).</p> <p><i>Note: If the alarm is specified as reserved in “Alarm Signals” on Page 9-24, software shall configure the behavior to “No Action”.</i></p> <p>0_B Configuration flag x (x=0-2) is set to 0 1_B Configuration flag x (x=0-2) is set to 1</p>

Note: The reset value specifies the NMI internal action for the watchdog timeout alarms. The reset value is the same regardless of the number of watchdogs present in the device.

Safety Management Unit (SMU)

SMU Alarm Configuration Registers related to Alarm Group 4.

SMU_AG4CFx (x=0-2)
Alarm Configuration Register $(130_H + x \cdot 04_H)$ **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFn (n = 0-31)	n	rw	<p>Configuration flag x (x=0-2) for alarm n belonging to alarm group 4.</p> <p>The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU when a fault state is reported by the alarm n belonging to this group (see “Alarm Configuration” on Page 9-41).</p> <p><i>Note: If the alarm is specified as reserved in “Alarm Signals” on Page 9-24, software shall configure the behavior to “No Action”.</i></p> <p>0_B Configuration flag x (x=0-2) is set to 0 1_B Configuration flag x (x=0-2) is set to 1</p>

Safety Management Unit (SMU)

SMU Alarm Configuration Registers related to Alarm Group 5.

SMU_AG5CFx (x=0-2)
Alarm Configuration Register (13C_H + x*04_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFn (n = 0-31)	n	rw	<p>Configuration flag x (x=0-2) for alarm n belonging to alarm group 5.</p> <p>The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU when a fault state is reported by the alarm n belonging to this group (see “Alarm Configuration” on Page 9-41).</p> <p><i>Note: If the alarm is specified as reserved in “Alarm Signals” on Page 9-24, software shall configure the behavior to “No Action”.</i></p> <p>0_B Configuration flag x (x=0-2) is set to 0 1_B Configuration flag x (x=0-2) is set to 1</p>

Safety Management Unit (SMU)

SMU Alarm Configuration Registers related to Alarm Group 6.

SMU_AG6CFx (x=0-2)

Alarm Configuration Register ($148_H + x \cdot 04_H$) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24	CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8	CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CFn (n = 0-31)	n	rw	<p>Configuration flag x (x=0-2) for alarm n belonging to alarm group 6.</p> <p>The configuration flags 0, 1 and 2 must be used together to define the behavior of the SMU when a fault state is reported by the alarm n belonging to this group (see “Alarm Configuration” on Page 9-41).</p> <p><i>Note: If the alarm is specified as reserved in “Alarm Signals” on Page 9-24, software shall configure the behavior to “No Action”.</i></p> <p>0_B Configuration flag x (x=0-2) is set to 0 1_B Configuration flag x (x=0-2) is set to 1</p>

Safety Management Unit (SMU)

9.5.4 SMU Alarm Configuration Registers (Fault Signaling Protocol)

SMU Fault Signalling Protocol Alarm Configuration Registers related to Alarm Group 0.

SMU_AG0FSP

FSP Configuration Register (180_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FE31	FE30	FE29	FE28	FE27	FE26	FE25	FE24	FE23	FE22	FE21	FE20	FE19	FE18	FE17	FE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	FE14	FE13	FE12	FE11	FE10	FE9	FE8	FE7	FE6	FE5	FE4	FE3	FE2	FE1	FE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEn (n = 0-31)	n	rw	Fault signalling configuration flag for alarm n belonging to alarm group 0. 0 _B FSP disabled for this alarm event 1 _B FSP enabled for this alarm event

Safety Management Unit (SMU)

SMU Fault Signalling Protocol Alarm Configuration Registers related to Alarm Group 1.

SMU_AG1FSP

FSP Configuration Register

(184_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FE31	FE30	FE29	FE28	FE27	FE26	FE25	FE24	FE23	FE22	FE21	FE20	FE19	FE18	FE17	FE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	FE14	FE13	FE12	FE11	FE10	FE9	FE8	FE7	FE6	FE5	FE4	FE3	FE2	FE1	FE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEn (n = 0-31)	n	rw	Fault signalling configuration flag for alarm n belonging to alarm group 1. 0 _B FSP disabled for this alarm event 1 _B FSP enabled for this alarm event

Safety Management Unit (SMU)

SMU Fault Signalling Protocol Alarm Configuration Registers related to Alarm Group 2.

SMU_AG2FSP

FSP Configuration Register

(188_H)

Reset Value: 2000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FE31	FE30	FE29	FE28	FE27	FE26	FE25	FE24	FE23	FE22	FE21	FE20	FE19	FE18	FE17	FE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	FE14	FE13	FE12	FE11	FE10	FE9	FE8	FE7	FE6	FE5	FE4	FE3	FE2	FE1	FE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEn (n = 0-31)	n	rw	Fault signalling configuration flag for alarm n belonging to alarm group 2. 0 _B FSP disabled for this alarm event 1 _B FSP enabled for this alarm event

Safety Management Unit (SMU)

SMU Fault Signalling Protocol Alarm Configuration Registers related to Alarm Group 3.

SMU_AG3FSP

FSP Configuration Register

(18C_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FE31	FE30	FE29	FE28	FE27	FE26	FE25	FE24	FE23	FE22	FE21	FE20	FE19	FE18	FE17	FE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	FE14	FE13	FE12	FE11	FE10	FE9	FE8	FE7	FE6	FE5	FE4	FE3	FE2	FE1	FE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEn (n = 0-31)	n	rw	Fault signalling configuration flag for alarm n belonging to alarm group 3. 0 _B FSP disabled for this alarm event 1 _B FSP enabled for this alarm event

Safety Management Unit (SMU)

SMU Fault Signalling Protocol Alarm Configuration Registers related to Alarm Group 4.

SMU_AG4FSP

FSP Configuration Register

(190_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FE31	FE30	FE29	FE28	FE27	FE26	FE25	FE24	FE23	FE22	FE21	FE20	FE19	FE18	FE17	FE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	FE14	FE13	FE12	FE11	FE10	FE9	FE8	FE7	FE6	FE5	FE4	FE3	FE2	FE1	FE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEn (n = 0-31)	n	rw	Fault signalling configuration flag for alarm n belonging to alarm group 4. 0 _B FSP disabled for this alarm event 1 _B FSP enabled for this alarm event

Safety Management Unit (SMU)

SMU Fault Signalling Protocol Alarm Configuration Registers related to Alarm Group 5.

SMU_AG5FSP

FSP Configuration Register

(194_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FE31	FE30	FE29	FE28	FE27	FE26	FE25	FE24	FE23	FE22	FE21	FE20	FE19	FE18	FE17	FE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	FE14	FE13	FE12	FE11	FE10	FE9	FE8	FE7	FE6	FE5	FE4	FE3	FE2	FE1	FE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEn (n = 0-31)	n	rw	Fault signalling configuration flag for alarm n belonging to alarm group 5. 0 _B FSP disabled for this alarm event 1 _B FSP enabled for this alarm event

Safety Management Unit (SMU)

SMU Fault Signalling Protocol Alarm Configuration Registers related to Alarm Group 6.

SMU_AG6FSP

FSP Configuration Register

(198_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FE31	FE30	FE29	FE28	FE27	FE26	FE25	FE24	FE23	FE22	FE21	FE20	FE19	FE18	FE17	FE16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE15	FE14	FE13	FE12	FE11	FE10	FE9	FE8	FE7	FE6	FE5	FE4	FE3	FE2	FE1	FE0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
FEn (n = 0-31)	n	rw	Fault signalling configuration flag for alarm n belonging to alarm group 6. 0 _B FSP disabled for this alarm event 1 _B FSP enabled for this alarm event

9.5.5 SMU Alarm Status Registers

SMU Alarm status registers related to the different alarm groups.

SMU_AGx (x=0-6)

Alarm Status Register (1C0_H + x*04_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SF31	SF30	SF29	SF28	SF27	SF26	SF25	SF24	SF23	SF22	SF21	SF20	SF19	SF18	SF17	SF16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SF15	SF14	SF13	SF12	SF11	SF10	SF9	SF8	SF7	SF6	SF5	SF4	SF3	SF2	SF1	SF0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
SFn (n = 0-31)	n	rwh	Status flag for alarm n belonging to alarm group x (x=0-6). 0 _B Status flag n does not report a fault condition 1 _B Status flag n reports a fault condition

Refer to [“Alarm Status Registers” on Page 9-44](#) for the conditions to set and reset the status flag by software.

9.5.6 SMU Alarm Debug Registers

SMU Alarm debug registers related to the different alarm groups.

SMU_AD_x (x=0-6)

Alarm Status Register (200_H + x*04_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DF31	DF30	DF29	DF28	DF27	DF26	DF25	DF24	DF23	DF22	DF21	DF20	DF19	DF18	DF17	DF16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DF15	DF14	DF13	DF12	DF11	DF10	DF9	DF8	DF7	DF6	DF5	DF4	DF3	DF2	DF1	DF0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

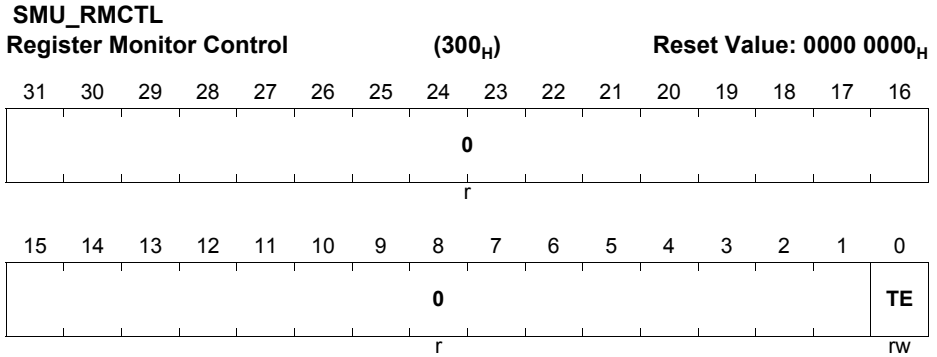
Field	Bits	Type	Description
DF_n (n = 0-31)	n	rh	Debug flag for alarm n belonging to alarm group x (x=0-6). 0 _B Status flag n does not report a fault condition 1 _B Status flag n reports a fault condition

Note: Writing to this register has no effect

Safety Management Unit (SMU)

9.5.7 SMU Special Safety Registers: Register Monitor

SMU register controlling the test mode of the Safety Flip-Flop safety mechanism (also called Register Monitor).



Field	Bits	Type	Description
TE	0	rw	Test Enable. This bit controls the timing of the test mode of the Safety Flip-Flop safety mechanism. This bit is broadcast to all modules that implement the Safety Flip-Flop safety mechanism. 0 _B Test mode disabled 1 _B Test mode enabled
0	[31:1]	r	Reserved Read as 0; should be written with 0

Safety Management Unit (SMU)

SMU register holding the test execution status of the Safety Flip-Flop safety mechanism.

SMU_RMEF

Register Monitor Error Flags

(304_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EF31	EF30	EF29	EF28	EF27	EF26	EF25	EF24	EF23	EF22	EF21	EF20	EF19	EF18	EF17	EF16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EF15	EF14	EF13	EF12	EF11	EF10	EF9	EF8	EF7	EF6	EF5	EF4	EF3	EF2	EF1	EF0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
EFn (n = 0-31)	n	rwh	<p>Status flag related to the different instances of the Safety Flip-Flop safety mechanism.</p> <p>In non-test mode (SMU_RMCTL.TE=0), the flag also reports a normal flip flop error condition. It can be used by software to identify the module where an error happened upon detection of an alarm event in ALM3[27].</p> <p>The mapping of EFn flags to Modules is: EF0: SMU EF1: SCU EF2: MTU EF3...31 not used</p> <p>This flag can only be cleared by software, a set by software has no effect.</p> <p>0_B Error flag n does not report a fault condition 1_B Error flag n reports a fault condition</p>

Safety Management Unit (SMU)

SMU register holding the test status of the Safety Flip-Flop safety mechanism.

SMU_RMSTS

Register Monitor Self Test Status (308_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS	STS
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
STSn (n = 0-31)	n	rwh	<p>Ready flag related to the different instances of the Safety Flip-Flop safety mechanism.</p> <p>This bit indicates whether the Safety Flip-Flop test is finished or not (when SMU_RMCTL.TE=1). This bit can only be cleared by software, a set by software has no effect.</p> <p>The mapping of STSn flags to Modules is:</p> <p>STS0: SMU STS1: SCU STS2: MTU STS3...31 not used</p> <p>0_B Self-test has not completed 1_B Self-test has completed</p>

10 Program Memory Unit (PMU)

The Program Memory Unit (PMU) controls the Flash memory and the BootROM and connects them to the SRI crossbar.

The devices of the AURIX family have at least one PMU. This is named “PMU0”. Depending on the amount of Flash memory more PMUs are added which are named “PMU1”, and so on. All PMUs are independent from each other.

TC26x B-Step has 1 PMU(s). Throughout this document a generic PMU is specified. The [Chapter 10.2](#) lists the configuration parameters of each PMU and their values in the TC26x B-Step. When PMU functionality depends on a parameter this is indicated in the text.

This chapter has the following structure:

- Generic feature list and block diagram ([Chapter 10.1](#)).
- PMU configuration of the TC26x B-Step ([Chapter 10.2](#)).
- Functionality of the BootROM ([Chapter 10.3](#)).
- Functionality of the tuning protection ([Chapter 10.4](#)).
- Functionality of the Flash memory ([Chapter 10.5](#)).
- Signaling to the Safety Management Unit “SMU” ([Chapter 10.6](#)).
- Register Set ([Chapter 10.7](#)).
- Application Hints ([Chapter 10.8](#)).

10.1 Generic Feature List

In general a PMU has the following features:

- Only in PMU0: BootROM.
- Program Flash “PFlash” with one or more banks “PFx”.
- Data Flash “DFlash” with one or more banks “DFx” containing:
 - Sectors for EEPROM emulation (only in PMU0).
 - UCB sectors for protection data.
- One Flash Standard Interface “FSI” that controls all Flash operations.
- Access control for safety and security for all assigned memories.
- Only in PMU0: Tuning protection (commonly called “Secure Watchdog”).
- Configuration and control registers.
- SRI interface(s) to all included resources. Separate SRI interfaces for DFlash and for each bank of PFlash.

[Figure 10-1](#) shows an abstracted block diagram of PMU, FSI, Flash and BootROM.

Program Memory Unit (PMU)

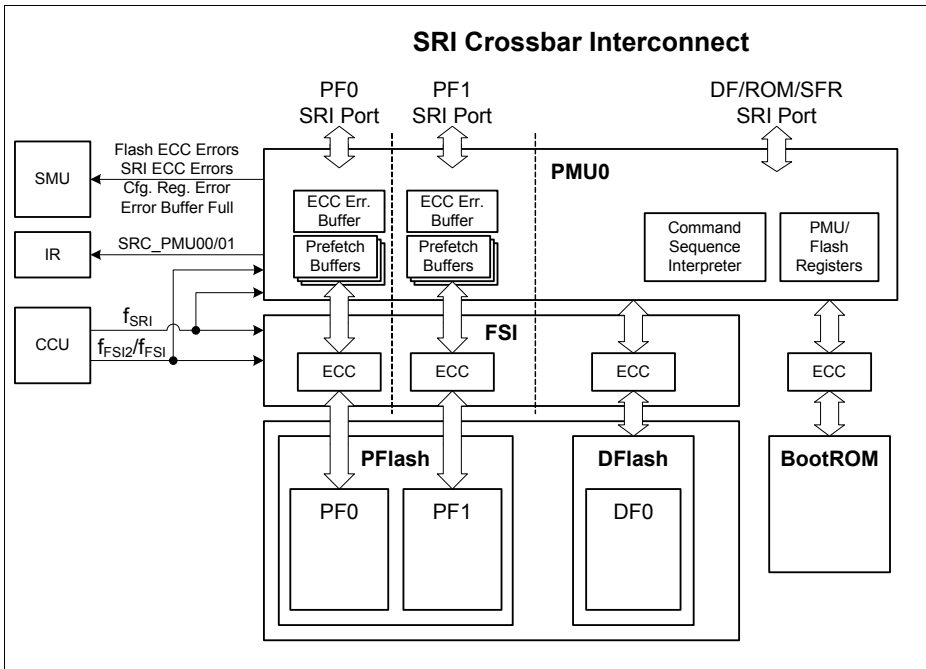


Figure 10-1 PMU/FSI/Flash Basic Block Diagram

10.2 PMU Configuration of TC26x B-Step

This chapter describes the superset of a range of devices with the following configuration:

- Number of PMUs: 1

Configuration of PMU0:

- PF0: 1 MByte.
- PF1: 1.5 MByte.
- DF0 consisting of:
 - DF_EEPROM: 96 KByte (12 logical sectors “EEPROM0 ... EEPROM11”).
 - DF_UCB: Flash area for protection data (16 logical sectors “UCB0 ... UCB15” with 1 KByte each).

Address Map

The following table(s) contains the address map.

Program Memory Unit (PMU)
Table 10-1 Address Map of PMU0

Start Address	Size	Range	Start Address Symbol
8000 0000 _H	1 MByte	PF0	AC_PF0
8010 0000 _H	1.5 MByte	PF1	AC_PF1
8FFF 8000 _H	32 KByte	BootROM	AC_BROM0
A000 0000 _H	1 MByte	PF0	AN_PF0
A010 0000 _H	1.5 MByte	PF1	AN_PF1
AF00 0000 _H	96 KByte	DF0 (DF_EEPROM) ¹⁾	AN_DFlash_B0
AF10 0000 _H	16 KByte	DF0 (DF_UCB)	
AF11 0000 _H	64 KByte	Not implemented (Reserved for DF1)	AN_DFlash_B1
AFFF 8000 _H	32 KByte	BootROM	AN_BROM0
F800 0500 _H	256 Byte	PMU Registers	
F800 1000 _H	5 KByte	Flash Registers	
FF11 0000 _H	64 KByte	Not implemented (Reserved for DF1, HSM private access)	AN_DFlash_B1F

1) In derivatives with reduced amount of DF0 (see data sheet) the available range is contiguous and starts also AN_DFlash_B0.

Derived Devices

This specification describes a superset of a range of devices. The concrete number and size of implemented Flash ranges is given in the respective Data Sheet.

In devices with PFlash banks of reduced size the sectors (see [Chapter 10.5.2.1](#)) with higher sector number are unavailable. The complete PFlash range is contiguous in all devices starting on AC_PF0/AN_PF0.

The “unavailable” PFlash sectors cannot exist or can be blocked during device manufacturing (the PMU handles these sectors then as not existing) or they stay accessible. In the latter case these sectors are delivered in the erased state. Full reliability is not ensured. These sectors shall not be read, programmed or erased. The write and OTP protection can be activated for these sectors to prevent later modification.

In devices with DFlash of reduced size the sectors EEPROM_x (see [Chapter 10.5.2.2](#)) with higher sector number *x* are unavailable (the whole DF_EEPROM range stays contiguous).

Program Memory Unit (PMU)

The “unavailable” DFlash sectors cannot exist or can be blocked during device manufacturing (the PMU handles these sectors then as not existing) or they stay accessible. In the latter case these sectors are delivered in the erased state.

The available DFlash sectors must be used round-robin to prevent accumulation of disturbs in unevenly cycled sectors (see [Chapter 10.5.2.2](#)).

10.2.1 Features of the BootROM

The BootROM contains the firmware:

- Startup software “SSW”: this software is executed at every reset.
- Test firmware: factory test routines for Infineon purposes.

Additionally the BootROM and its control logic implement the Tuning protection feature (“Secure Watchdog”).

10.2.2 Features of the Program and Data Flash

Depending on the PMU configuration the following Flash features are implemented. Timing and reliability figures are just indicative. Binding data can be found in the “Data Sheet”.

10.2.2.1 Program Flash Features:

Summary of Program Flash features:

- Consists of 2 bank(s). All banks are concurrently readable.
- Separate SRI ports, ECC decoders and pre-fetch logic for each bank.
- Commonly used for instructions and constant data.
- High throughput burst read based on a 256-bit Flash access (see parameter table t_{PF}).
- Application optimized sector structure with logical sectors ranging from 16 KBytes to 256 KBytes.
- Write protection separately configurable per logical sector (one-time programmable and 256-bit password protected).
- Password based read protection combined with write protection.
- Sectors (one 16 KB and two 64 KB) configurable as HSM code sectors by setting HSM_exclusive flag.
- Separate read, write and OTP protection for the HSM code sectors.
- Erase counter counts each sector erase (saturating).
- Fast programming of 32 byte pages (see Data Sheet t_{PRP}).
- High throughput burst programming of 256 byte units (see Data Sheet t_{PRPB}).
- Erase time per logical sector: see Data Sheet t_{ERP} .
- For development purposes sectors may incur more program/erase cycles than for productive purposes. Additionally further restrictions apply as an increased failure rate and lower retention.

Program Memory Unit (PMU)

- High throughput erase by multi-sector erase commands: see Data Sheet t_{MERP} .
- Erase and program performed by a Flash specific control logic independent of the CPU.
- Fast suspend erase to read and suspend burst programming to read command.
- Erase verify command.
- End of erase and program operations reported by interrupt.
- Configurable stall or bus error generation when reading busy PFlash banks.
- Dynamic correction of single-bit and double-bit errors and detection of triple-bit errors (“DEC-TED”).
- “Safe read path” ensuring detection of transient and permanent errors including addressing faults.
- ASIL-D optimized ECC for detection of >99% of faults in the white-noise model.
- Error reporting to the SMU, additionally local status flags and bus error generation.
- Margin reads for quality assurance.
- Delivery in the erased state.
- Configurable wait-state configuration.
- Endurance and retention figures are documented in the Data Sheet.
- Pad supply voltage used for program and erase. Separate programming modes for 5V and 3.3V supplied devices.

10.2.2.2 Data Flash Features

Summary of Data Flash features:

- Contains 12 EEPROMx sectors commonly used for EEPROM emulation (data storage at application run-time).
- UCBx sectors used for protection installation and the erase-cycle counter.
- Read-only UCB configured by Infineon with unique chip identifier and trimming data.
- Password based read protection combined with write protection.
- Flash read access based on 64-bit reads (see Data Sheet t_{DF}).
- Read path separated from PFlash (DFlash reads don't influence PFlash read accesses).
- Burst read is not supported.
- Requested read (data reading performed in background, read data supplied in registers).
- Fast programming of 8 byte pages (see Data Sheet t_{PRD}).
- High throughput burst programming of 32 byte units (see Data Sheet t_{PRDB}).
- Erase time per sector: see Data Sheet t_{ERD} .
- High throughput erase by multi-sector erase commands: see Data Sheet t_{MERD} .
- Erase and program performed by a Flash specific control logic independent of the CPU.
- Fast suspend erase to read command.
- End of erase and program operations reported by interrupt.

Program Memory Unit (PMU)

- Dynamic correction of single-bit, double-bit and triple-bit errors and detection of quad-bit errors ("TEC-QED").
- Error reporting to the SMU, additionally local status flags and bus error generation.
- Margin reads for quality assurance.
- Delivery in the erased state.
- Configurable wait-state configuration.
- The high endurance (see Data Sheet N_E) is granted under the condition of a robust EEPROM emulation algorithm (see corresponding section in the PMU chapter of the User's Manual).
- Endurance and retention figures are documented in the Data Sheet.
- Pad supply voltage used for program and erase.

10.3 BootROM

The content of the BROM is described in a separate chapter.

The BROM is readable and executable for user software but its functions shall only be executed when especially advised by Infineon.

All write accesses to the BootROM are refused with a bus error.

Its content is protected with ECC having SEC-DED capability and address error detection.

In the SRI address range it is mapped to the following start addresses:

- AC_BROM0 (cached address range).
- AN_BROM0 (non-cached address range).

10.4 Tuning Protection

The special tuning protection support represents a security function provided additionally to Flash read/write/OTP protection.

For details on the tuning protection please contact your Infineon representative.

10.5 Flash

This chapter introduces the Flash memory of the TC26x B-Step. It is split into the following sections:

- Definition of terms (**Chapter 10.5.1**): what means “program”, “erase”, “sector”, “pages”, ...
- Structure of a Flash module (**Chapter 10.5.2**): separation into “banks”, “sectors”, “word-lines”, “pages”, ...
- Reading Flash (**Chapter 10.5.3**).
- Command sequences for Flash (**Chapter 10.5.4**): programming, erasing, handling protection.
- Flash protection (**Chapter 10.5.5**): read and write protection.
- Data integrity and safety (**Chapter 10.5.6**): ECC and margin checks.
- Interrupt and traps (**Chapter 10.5.7**).
- Reset and startup (**Chapter 10.5.8**).
- Power reduction by sleep and idle (**Chapter 10.5.9**).

10.5.1 Definition of Terms

The description of Flash memories uses a specific terminology for operations and the hierarchical structure.

Flash Operation Terms

- **Erasing**: The erased state of a Flash cell is logical ‘0’. Forcing a cell to this state is called “erasing”. Depending on the Flash area always complete physical sectors, logical sectors or word-lines are erased. All Flash cells in this area incur one “cycle” that counts for the “endurance”.
- **Programming**: The programmed state of a cell is logical ‘1’. Changing an erased Flash cell to this state is called “programming”. The 1-bits of a page are programmed concurrently.
- **Retention**: This is the time during which the data of a flash cell can be read reliably. The retention time is a statistical figure that depends on the operating conditions of the device (e.g. temperature profile) and is affected by operations on other Flash cells in the same word-line and physical sector. With an increasing number of program/erase cycles (see endurance) the retention is lowered.
- **Endurance**: As described above the data retention is reduced with an increasing number of program/erase cycles. The maximum number of program/erase cycles of each Flash cell is called “endurance”. As said for the retention it is a statistical figure that depends on operating conditions and the use of the flash cells and not to forget on the required quality level. The endurance is documented in the data sheet separately for PFlash logical sectors, UCBs and EEPROM sectors.

Flash Structure Terms

- **Flash Module:** A PMU contains one “Flash module” with its own operation control logic.
- **Bank:** A “Flash module” contains separate “banks”. In the PFlash there are one or more PFX banks and in the DFlash two DFX banks. “Banks” support concurrent operations with some limitations due to common logic.
- **Sub-Sector:** A Flash “bank” consists of sub-sectors. There are “physical sub-sectors” which are isolated from each other and “logical sub-sectors”. A PFlash bank is built from 512 KByte physical sub-sectors. A DFlash bank is separated into logical sub-sectors.
- **Logical Sector:** The sub-sectors are further separated into “logical sectors” which are groups of word-lines. They can be erased with a single operation.
- **Sector:** The plain term “sector” means “logical sector”.
- **User Configuration Block “UCB”:** A “UCB” is a specific logical sector. It is part of a DF_UCB sub-sector of bank DF0. It contains the protection settings and other data configured by the user.
- **Configuration Sector:** The “configuration sector” is a logical sector of the PFlash. It is not directly accessible to the user.
- **Page:** In the PFlash a page is an aligned group of 32 bytes and in DFlash of 8 bytes. It is the smallest unit that can be programmed (ECC width).
- **Program Burst:** A “burst” is the maximum amount of data that can be programmed with one command. The programming throughput is higher than for programming single pages. A burst in PFlash consists of 8 pages (256 bytes) and in DFlash 4 pages (32 bytes).
- **Word-Line:** In the PFlash and DFlash a word-line is an aligned group of 512 bytes.

10.5.2 Flash Structure

The PMU0 of TC26x B-Step contains the following Flash banks. The offset address of each sector is relative to the base address of its bank which is given in [Table 10-1](#). In devices of the same family the Flash banks keep the same base address.

10.5.2.1 PFlash

All banks PFX of the PFlash of all PMUs are based on the same sectorization (see [Table 10-2](#)). PFX banks with reduced capacity are built by cutting off sectors at the end. A 2 MByte bank implements logical sectors S0 to S26, a 1.5 MByte bank logical sectors S0 to S24 and a 1 MByte bank logical sectors S0 to S22. The implemented bank sizes are described in the Data Sheet.

Program Memory Unit (PMU)
Table 10-2 Sector Structure of PFx of PMU0

Logical Sector	Phys. Sub-Sector	Size	Offset Address ¹⁾
S0	PS0 (512 KB)	16 KB	00'0000 _H
S1		16 KB	00'4000 _H
S2		16 KB	00'8000 _H
S3		16 KB	00'C000 _H
S4		16 KB	01'0000 _H
S5		16 KB	01'4000 _H
S6		16 KB	01'8000 _H
S7		16 KB	01'C000 _H
S8		32 KB	02'0000 _H
S9		32 KB	02'8000 _H
S10		32 KB	03'0000 _H
S11		32 KB	03'8000 _H
S12		32 KB	04'0000 _H
S13		32 KB	04'8000 _H
S14		32 KB	05'0000 _H
S15		32 KB	05'8000 _H
S16		64 KB	06'0000 _H
S17	64 KB	07'0000 _H	
S18	PS1 (512 KB)	64 KB	08'0000 _H
S19		64 KB	09'0000 _H
S20		128 KB	0A'0000 _H
S21		128 KB	0C'0000 _H
S22		128 KB	0E'0000 _H
S23	PS2 (512 KB)	256 KB	10'0000 _H
S24		256 KB	14'0000 _H
S25	PS3 (512 KB)	256 KB	18'0000 _H
S26		256 KB	1C'0000 _H

1) Offset with respect to AC_PFx and AN_PFx (see [Table 10-1](#)).

The sectors S0, S3, S7 and S8 of PF0 in PMU0 contain the BMI headers.

Program Memory Unit (PMU)

The sectors S6, S16 and S17 of PF0 in PMU0 are used as HSM code sectors in devices with HSM. Their access control (read, program, erase) is different from other PFlash sectors.

The sector S5 of PF0 in PMU0 has a specific purpose with an enabled Tuning Protection.

10.5.2.2 DFlash of PMU0

Bank DF0 of the DFlash of PMU0 (see [Table 10-3](#)).

Attention: *In the DF_EEPROM all sectors must be used round-robin in order to prevent the accumulation of erase disturbs in static sectors¹⁾. In derivatives with reduced amount of DFlash all sectors in this reduced DFlash range must be used round-robin.*

Attention: *In DF_EEPROM the overall number of erase commands over lifetime is limited by the parameter N_{ERD0} ²⁾, in DF_HSM by N_{ERD1} ³⁾.*

Attention: *In DF_EEPROM and DF_HSM at most 384 KByte shall be erased with one erase command.*

Table 10-3 Sector Structure of DFlash Bank 0

Logical Sector	Log. Sub-Sector	Size	Offset Address ¹⁾
EEPROM0	DF_EEPROM	8 KByte	00'0000 _H
EEPROM1		8 KByte	00'2000 _H
EEPROM2		8 KByte	00'4000 _H
EEPROM3		8 KByte	00'6000 _H
...	
EEPROM11		8 KByte	01'6000 _H

1) The parameter N_{DFD} (see Electrical Parameters or Data Sheet) documents the erase disturb limit. It is chosen higher than needed for pure round-robin usage to cover also error cases (e.g. repetition of erase commands on the same range due to power failures during operation).

2) This parameter is chosen to not reduce the endurance when always with one erase command at least 1/6th of the available DF_EEPROM is erased.

3) This parameter is chosen to not reduce the endurance when always with one erase command at least 1/4th of the available DF_HSM is erased

Program Memory Unit (PMU)
Table 10-3 Sector Structure of DFlash Bank 0 (cont'd)

Logical Sector	Log. Sub-Sector	Size	Offset Address ¹⁾
UCB0 = UCB_PFlash	DF_UCB	1 KByte	10'0000 _H
UCB1 = UCB_DFlash		1 KByte	10'0400 _H
UCB2 = UCB_HSMCOTP		1 KByte	10'0800 _H
UCB3 = UCB_OTP		1 KByte	10'0C00 _H
UCB4 = UCB_IFX		1 KByte	10'1000 _H
UCB5 = UCB_DBG		1 KByte	10'1400 _H
UCB6 = UCB_HSM		1 KByte	10'1800 _H
UCB7		1 KByte	10'1C00 _H
UCB8		1 KByte	10'2000 _H
UCB9		1 KByte	10'2400 _H
UCB10		1 KByte	10'2800 _H
UCB11		1 KByte	10'2C00 _H
UCB12		1 KByte	10'3000 _H
UCB13		1 KByte	10'3400 _H
UCB14		1 KByte	10'3800 _H
UCB15		1 KByte	10'3C00 _H

1) Offset with respect to AN_DFlash_B0 (see [Table 10-1](#)).

10.5.3 Flash Read Access

Flash banks that are active and in read mode can be directly (memory mapped) read as ROM.

The wait cycles for the Flash access must be configured as described in ([Chapter 10.5.3.3](#)).

The PFlash allows SRI bursts BTR4 and BTR2 and single transfers.

The Tricore uses BTR4 for code fetches from the cached address range in order to fill one cache line. Code fetches from the non-cached area are also performed with BTR4. Data reads from the cached range are performed with BTR4. Data reads from the non-cached address range are performed with single transfers.

The DFlash allows only single transfers. Therefore they are only accessible in the non-cached address range and Tricore can't fetch instructions from them.

Read accesses from Flash can be blocked by the read protection (see [Chapter 10.5.5](#)).

Program Memory Unit (PMU)

Read accesses from busy Flash banks are not possible (see [Chapter 10.5.4.1](#)).

ECC errors can be detected and corrected (see [Chapter 10.5.6](#)).

The DFlash sectors EEPROMx can be also read with requested reads (see [Chapter 10.5.3.4](#)).

10.5.3.1 Read Ports

For optimum performance the PFlash is read via dedicated SRI slave ports. Each PFp bank is assigned to one SRI port determined by its address (see [Table 10-4](#)). The DFlash, the BootROM and the registers share a separate SRI slave port.

Table 10-4 Memory Range to SRI Port Assignment

SRI Port	Address Range	Memory Ranges
p = 0 mapped to SCI 7	8000 0000 _H – 800F FFFF _H	PF0 (cached)
p = 8 mapped to SCI 9	8010 0000 _H – 8027 FFFF _H	PF1(cached)
p = 0 mapped to SCI 7	A000 0000 _H – A00F FFFF _H	PF0 (non-cached)
p = 1 mapped to SCI 8	A010 0000 _H – A027 FFFF _H	PF1(non-cached)
mapped to SCI 6	8FFF 8000 _H – 8FFF FFFF _H AF00 0000 _H – AF01 7FFF _H AF10 0000 _H – AF10 3FFF _H AF11 0000 _H – AF11 FFFF _H AFFF 8000 _H – AFFF FFFF _H F800 0500 _H – F800 05FF _H F800 1000 _H – F800 23FF _H FF11 0000 _H – FF11 FFFF _H	BootROM DF0 (DF_EEPROM) DF0 (DF_UCB) reserved for DF1 BootROM PMU Registers Flash Registers reserved for HSM private Flash command interface and DF1.

Each SRI port can be independently read. Each has its own path to its Flash range and its own ECC decoding logic.

As additional performance increasing measure each of the PFp ports has its own prefetching logic and a set of 3 read buffers.

Every read buffer is assigned by configuration (see [Chapter 10.7.2.7](#)) to one bus master. Additionally all read data passes through a pipeline stage in the FSI called “global read buffer”. The read buffers can be filled with a single PFlash read which delivers 256 data bits.

Upon finishing a pre-fetch and with no request from the bus the PMU reads the sequentially following 256 bits into the global read buffer. Buffer hits in this read buffer are also supported and don’t cause a Flash read. When this read buffer is also filled the PMU sends already the address for the following 256 bits to the Flash and keep this

Program Memory Unit (PMU)

address stable (this feature can be disabled with FCON.IDLE). A hit with this address is also detected and will reduce the Flash access time.

All read buffers are invalidated upon executing a command sequence that changed PFlash content. The read buffers are not invalidated by changes to the configuration registers (e.g. changing wait-states or changing the ECC correction).

10.5.3.2 DFlash, BootROM Read Port

The other resources (DFlash, BootROM, registers, etc) share a separate SRI slave port independent of the PFlash read ports.

Read accesses to the DFlash are always serviced by reading the Flash. Buffer hits are not supported.

10.5.3.3 Configuring Flash Wait Cycles

The Flash read path including the ECC decoders and read buffer accesses use the following clocks:

- PFX read ports: clocked with f_{FSI2} .
- DFlash, BootROM, register read port: clocked with f_{FSI} .

The wait cycles in register FCON are counted with this reference clock. The minimum number of wait cycles can be calculated based on the delays given in the Data Sheet or Electrical Specification as described in [Table 10-5](#).

Note: Please note that in case of using the FM-PLL the maximum frequency of these clocks has to enter the calculation.

Attention: After System Reset and Power-On Reset the wait cycles are configured for the clock configuration used during startup, i.e. f_{FSI} and f_{FSI2} running with maximum 100 MHz. Before changing to higher clock frequencies this default configuration must be changed.

The clock f_{FSI2} can be configured to be identical to f_{SRI} or f_{FSI} . Selecting the higher clock frequency increases the read performance.

Table 10-5 Wait Cycle Calculation

	Register Field	Minimum Value¹⁾
PFlash read access delay	FCON.WSPFLASH	$\text{Ceil}(t_{PF} * f_{FSI2}) - 1$
PFlash ECC decode delay	FCON.WSECPF	$\text{Ceil}(t_{PF ECC} * f_{FSI2}) - 1$
DFlash read access delay	FCON.WSDFLASH	$\text{Ceil}(t_{DF} * f_{FSI}) - 1$
DFlash ECC decode delay	FCON.WSECDF	$\text{Ceil}(t_{DF ECC} * f_{FSI}) - 1$

1) The Ceil(r) function rounds a real number up, i.e. the result is the smallest integer not less than the real argument. The mathematical function name is "Ceiling(r)".

Program Memory Unit (PMU)**Examples**

Example configuration for $t_{PF} = 30$ ns, $t_{PFECC} = 10$ ns, $t_{DF} = 100$ ns, $t_{DFECC} = 20$ ns with $f_{FSI2} = 200$ MHz and $f_{FSI} = 100$ MHz:

- FCON.WSPFLASH = 5
- FCON.WSECPF = 1
- FCON.WSDFLASH = 9
- FCON.WSECDF = 1

10.5.3.4 Requested DFlash Read

The direct memory mapped DFlash read stalls the requesting master for the duration of the read access. Applications that favor latency over throughput can use the “requested read” feature. This supports interrupt driven background reads from the DFlash.

The start address is written into RRAD.ADD. RRCT.CNT contains the number of 8 byte aligned transfers that will be performed.

The reading of each 8 byte block is started by setting RRCT.START. When the PMU starts the read process is clears RRCT.START and sets RRCT.BUSY.

When the data is available in RRD0-1 the flag RRCT.BUSY is cleared and RRCT.DONE is set. If RRCT.EOBM is set the requested read interrupt (SRC_PMU1) is triggered. The address in RRAD.ADD is incremented to the next address and the counter RRCT.CNT is decremented by 1.

The reading of the next 8 byte block has to be triggered again by setting RRCT.START. This clears RRCT.DONE.

The requested read can be aborted by setting RRCT.STP. This clears RRCT.START, RRCT.BUSY and RRCT.DONE. The requested read interrupt is not triggered.

An error is reported by RRCT.ERR when the read access addresses an illegal address (outside of the assigned DFlash ranges, insufficient access rights) or when started with CNT=0. This sets also RRCT.DONE.

The occurrence of an uncorrectable ECC error sets RRCT.ERR. The ECC error flags in the FSR are not influenced by requested reads.

Direct reads to the DFlash are not blocked by ongoing requested reads. They are executed with higher priority than requested reads.

10.5.4 Flash Operations

The Flash memory knows operations for reading, changing data (program/erase) and handling protection settings. This section and the following describe only the features as such. How to use them is described in the application notes ([Chapter 10.8](#)).

10.5.4.1 Modes of Operation

A Flash module can be in one of the following states:

- Active (normal) mode.
- Sleep mode (see [Chapter 10.5.8](#)).

In sleep mode write and read accesses to all Flash ranges of this PMU are refused with a bus error.

When the Flash module is in normal mode a Flash bank can be in one of these modes:

- Read mode.
- Command mode.

In read mode a Flash bank can be read and command sequences are interpreted. In read mode a Flash bank can additionally enter page mode which enables it to receive data for programming.

In command mode an operation is performed. During its execution the Flash bank reports BUSY in FSR. In this mode read accesses to this Flash bank are refused with a bus error or the ready is suppressed until BUSY clears. This can be configured (see [FCON.STALL](#)). At the end of an operation the Flash bank returns to read mode and BUSY is cleared. Only operations with a significant duration (shown in the command documentation) set BUSY.

Note: Using FCON.STALL = "STALL" is not recommended because the stalled CPU is inoperable and also the PMU SRI port is blocked as long the ready is suppressed. This feature should be only used under controlled conditions by a Flash loader.

Note: Using FCON.STALL = "STALL" together with Flash sleep ([Chapter 10.5.9](#)) is prohibited as it might lead to device hang-up.

In command mode further command sequences in this flash module are not allowed. They are refused with a bus error.

Flash modules of different PMUs are completely independent.

Register read and write accesses are not affected by these modes.

10.5.4.2 Command Sequences

All Flash operations except memory mapped reads are performed with command sequences. Every write access to the DF0 memory range is interpreted as command cycle belonging to a command sequence.

Write accesses to the PFlash memory range are refused with bus error.

Command sequences consist of 1 to 9 command cycles. The command interpreter checks that a command cycle is correct in the current state of command interpretation. Else a SQER is reported.

Program Memory Unit (PMU)

When the command sequence is accepted the last command cycle finishes read mode and the Flash bank transitions into command mode.

Command sequences can be blocked by the register access protection (see [Chapter 10.5.5.2](#)).

Generally when the command interpreter detects an error it reports a sequence error by setting FSR.SQER. Then the command interpreter is reset and a page mode is left. The next command cycle must be the 1st cycle of a command sequence. The only exception is “Enter Page Mode” when a Flash is already in page mode (see below).

10.5.4.3 Command Sequence Definitions

The command sequence descriptions use the following nomenclature (symbolic assembly language):

ST addr, data: Symbolic representation of a command cycle moving “data” to “addr”.

The parameter “addr” is defined as followed:

- **CCCC_H:** The “addr” must point into the DFlash. The last 16 address bits must match CCCC_H.

The parameter “data” can be one of the following:

- **PA:** Absolute start address of the Flash page. Must be aligned to burst size for “Write Burst” or to the page size for “Write Page”.
- **SA:** Absolute start address of a Flash sector. Allowed are the PFlash sectors Sx and the DFlash sectors EEPROMx, UCBx.
- **WD:** 64-bit or 32-bit write data to be loaded into the page assembly buffer.
- **xxYY:** 8-bit write data as part of a command cycle. Only the byte “YY” is used for command interpretation. The higher order bytes “xx” are ignored.
 - **xx5y:** Specific case for “YY”. The “y” can be “0_H” for selecting the PFlash or “D_H” to select the DFlash.
 - **xxnn:** Specific case for “YY”. The “nn” quantifies the number of logical sectors that are erased or verified. This number underlies certain restrictions (see [Erase Logical Sector Range](#) on [Page 10-20](#)).
- **UC:** Identification of the UCBx for which the password checking shall be performed: xx00_H for UCB0 = UCB_PFlash, xx01_H for UCB1 = UCB_DFlash, xx05_H for UCB5 = UCB_DBG.
- **PWx:** 32-bit word of a 256-bit password.

When using for command cycles 64-bit transfers the “data” is expected in the correct 32-bit word as indicated by the address “addr”.

Command Sequence Overview Table

The [Table 10-6](#) summarizes all commands sequences. The following sections describe each command sequence in detail.

Table 10-6 Command Sequences for Flash Control

Command Sequence		1. Cycle	2./6. Cycle	3./7. Cycle	4./8. Cycle	5./9. Cycle	6. Cycle
Reset to Read	Address	.5554					
	Data	..xxF0					
Enter Page Mode	Address	.5554					
	Data	..xx5y					
Load Page	Address	.55F0					
	Data	¹⁾ WD					
Write Page	Address	.AA50	.AA58	.AAA8	.AAA8		
	Data	PA	..xx00	..xxA0	..xxAA		
Write Page Once	Address	.AA50	.AA58	.AAA8	.AAA8		
	Data	PA	..xx00	..xxA0	..xx9A		
Write Burst	Address	.AA50	.AA58	.AAA8	.AAA8		
	Data	PA	..xx00	..xxA0	..xx7A		
Erase Logical Sector Range	Address	.AA50	.AA58	.AAA8	.AAA8		
	Data	SA	..xxnn	..xx80	..xx50		
Verify Erased Logical Sector Range	Address	.AA50	.AA58	.AAA8	.AAA8		
	Data	SA	..xxnn	..xx80	..xx5F		
Resume Prog/Erase	Address	.AA50	.AA58	.AAA8	.AAA8		
	Data	PA/SA	..xxnn	..xx70	..xxCC		
Disable Protection	Address	.553C	.553C	.553C	.553C	.553C	
	Data	UC	PW0/4	PW1/5	PW2/6	PW3/7	
Resume Protection	Address	.5554					
	Data	..xxF5					
Clear Status	Address	.5554					
	Data	..xxFA					

1) Address depends on data width of WD (see description of Load Page below).

Reset to Read

Calling:

- ST 5554_H, xxF0_H

Function:

This function resets the addressed command interpreter to its initial state (i.e. the next command cycle must be the 1st cycle of a sequence). A page mode is aborted.

Program Memory Unit (PMU)

This command is the only that is accepted without SQER when the command interpreter has already received command cycles of a different sequence but is still not in command mode. Thus “Reset to Read” can cancel every command sequence before its last command cycle has been received.

The error flags of **FSR** (OPER, SQER, PROER, PFSBER, PFDBER, PFMBER, DFSBER, DFDBER, DFTBER, DFMBER, ORIER, PVER, EVER) are cleared. The flags can be also cleared in the status registers without command sequence.

Enter Page Mode

Calling:

- ST 5554_H, xx5y_H

Function:

The PFlash or the DFlash assembly buffer enter page mode selected by the parameter “y”.

The write pointer of the page assembly buffer is set to 0, its previous content is maintained.

The page mode is signaled by the flags PFPAGE/DFPAGE in the FSR separately for PFlash and DFlash.

If a new “Enter Page Mode” command sequence is received while any Flash is already in page mode SQER is set, the existing page mode is aborted, and the new “Enter Page Mode” sequence is correctly executed (i.e. in this case the command interpreter is not reset).

Load Page

Calling:

- ST 55F0_H/55F4_H, WD

Function:

Loads the data “WD” into the page assembly buffer and increments the write pointer to the next position.

All WD transfers for one page must have the same width (either all 32-bit or all 64-bit). Else the transfer is refused with SQER.

When using 64-bit transfers then all have to address the offset address 55F0_H.

When using 32-bit transfers then the lower order word has to be written first to offset 55F0_H and then the higher order word to offset 55F4_H. This alternating pattern is repeated for all WD.

If “Load Page” is called more often than allowed by the available buffer space of 256 bytes the overflow data is discarded, the page mode is not left. This overflow is reported by the following Write Page/Burst command with SQER.

Program Memory Unit (PMU)**Write Page**

Calling:

- ST AA50_H, PA
- ST AA58_H, xx00_H
- ST AAA8_H, xxA0_H
- ST AAA8_H, xxAA_H

Function:

This function starts the programming process for one page with the data transferred previously by “Load Page” commands. Upon entering command mode the page mode is finished (indicated by clearing the corresponding PAGE flag) and the BUSY flag of the bank is set.

A page shall be programmed only once after erasing.

This command is refused with SQER when the addressed Flash is not in page mode.

SQER is also issued when PA addresses an unavailable Flash range or when PA does not point to a legal page start address.

If after “Enter Page Mode” too few data, no data or too much data was transferred to the assembly buffer with “Load Page” then “Write Page” programs the page but sets SQER. Missing data is programmed with the previous content of the assembly buffer.

When the page “PA” is located in a sector with active write protection or the Flash module has an active global read protection the execution fails and PROER is set.

When the programming process incurs an error the flag PVER is set.

Write Page Once

Calling:

- ST AA50_H, PA
- ST AA58_H, xx00_H
- ST AAA8_H, xxA0_H
- ST AAA8_H, xx9A_H

Function:

This function starts the programming process for one page as the normal “Write Page” does. But before programming it checks if the page is erased. If the page is not erased (allowing correctable errors) the command fails with PVER and EVER.

When the programming itself incurs an error the flag PVER is set.

On sectors with “write-once” protection only this write command can be applied.

This command is only supported for PFlash. When applied to DFlash the command fails with SQER.

Program Memory Unit (PMU)**Write Burst**

Calling:

- ST AA50_H, PA
- ST AA58_H, xx00_H
- ST AAA8_H, xxA0_H
- ST AAA8_H, xx7A_H

Function:

This function starts the programming process for an aligned group of pages. The programming process is more efficient than for a single page, achieving a significantly higher throughput (see data sheet).

In the PFlash 8 pages (256 bytes) are programmed and in DFlash 4 (32 bytes). The pages are programmed one after the other with increasing addresses.

Please note that with disabled ECC generation with ECCW.PECENCDIS or DECENCDIS this command sequence must not be used. It causes unpredictable results.

For further details see “Write Page”.

Erase Logical Sector Range

Calling:

- ST AA50_H, SA
- ST AA58_H, xxnn_H
- ST AAA8_H, xx80_H
- ST AAA8_H, xx50_H

Function:

This command erases “nn” sectors starting at the sector addressed by “SA”. Upon entering command mode the BUSY flag of the corresponding bank is set. When erasing PFlash sectors additionally D0BUSY is set (because the erase counters are programmed in DF0).

SQER is returned when SA does not point to the base address of a correct sector (as specified at the beginning of this section) or to an unavailable sector.

When SA or any of the following nn-1 sectors has an active write protection or the Flash module has an active global read protection the execution fails and PROER is set.

The command fails with SQER if the range of logical sectors is not contained in one physical sector.

The error flag EVER is set to indicate an error during the erase process. The flags EVER and PVER are set when the erase counter programming incurs an error.

Note: The duration of an erase command applied to DFlash can be much shorter than documented as maximum duration (t_{ERD} , t_{MERD}).

Verify Erased Logical Sector Range

Calling:

- ST AA50_H, SA
- ST AA58_H, xxnn_H
- ST AAA8_H, xx80_H
- ST AAA8_H, xx5F_H

Function:

This command verifies if “nn” sectors starting at the sector addressed by “SA” are correctly erased, i.e. contain 0 data and ECC bits. Upon entering command mode the BUSY flag of the corresponding bank is set.

When the PFlash Safety ECC is enabled (see [Chapter 10.5.6.2](#)) the PFlash cannot be checked for “0” content by direct reads.

The error flag EVER is set to identify a found verification error.

SQER is returned when SA does not point to the base address of a correct sector (as specified at the beginning of this section) or to an unavailable sector.

The command fails with SQER if the range of logical sectors is not contained in one physical sector.

Especially security concerned customers can limit this function by configuring PROCONHSMCOTP.BLKFLAN to ‘1’:

- If PROCONHSMCOTP.BLKFLAN is 0 then “Verify Erase Logical Sector Range” is enabled for every address.
- If PROCONHSMCOTP.BLKFLAN is 1 then “Verify Erase Logical Sector Range” is blocked as follows:
 - If one of the PROCONCOTP.HSMsX = 1: blocked on the corresponding HSM code sector “s”.
 - If FPRO.PROINP = 1 and FPRO.PRODISP = 0: blocked on UCB_PFlash.
 - If FPRO.PROIND = 1 and FPRO.PRODISD = 0: blocked on UCB_DFlash.
 - If FPRO.PROINDBG = 1 and FPRO.PRODISDBG = 0: blocked on UCB_DBG.

If this function is blocked the command fails by setting PROER.

Note: The execution of this command has an effect on the Flash cells (they are slightly disturbed). Therefore the number of calls of this function is limited, see Data Sheet parameters N_{VERD} , N_{VERP} .

Resume Prog/Erase

Calling:

- ST AA50_H, PA/SA
- ST AA58_H, xxnn_H
- ST AAA8_H, xx70_H
- ST AAA8_H, xxCC_H

Program Memory Unit (PMU)

Function:

A suspended programming or erasing process is restarted as described in [Chapter 10.5.4.4](#).

Disable Protection

Calling:

- ST 553C_H, UC
- ST.W 553C_H, PW0
- ST.W 553C_H, PW1
- ST.W 553C_H, PW2
- ST.W 553C_H, PW3
- ST.W 553C_H, PW4
- ST.W 553C_H, PW5
- ST.W 553C_H, PW6
- ST.W 553C_H, PW7

Function:

The password protection of the selected UCB (if this UCB offers this feature) is temporarily disabled by setting FPRO.PRODIS_x (with “x” indicating the UCB) when all the passwords PW0–PW7 match their configured values in the corresponding UCB.

The command fails by setting PROER when any of the supplied PWs does not match. In this case until the next application reset all further calls of “Disable Protection” fail with PROER independent of the supplied password.

Resume Protection

Calling:

- ST 5554_H, xxF5_H

Function:

This command clears all FPRO.PRODIS_x effectively enabling again the Flash protection as it was configured.

Clear Status

Calling:

- ST 5554_H, xxFA_H

Function:

The flags FSR.PROG and FSR.ERASE and the error flags of **FSR** (OPER, SQER, PROER, PFSBER, PFDBER, PFMBER, DFSBER, DFDBER, DFTBER, DFMBER, ORIER, PVER, EVER) are cleared. These flags can be also cleared in the status registers without command sequence.

10.5.4.4 Operation Suspend and Resume

The operations “Erase Logical Sector Range”, “Verify Erased Logical Sector Range” and “Write Page”¹⁾ and “Write Burst”²⁾ can be suspended.

An operation is suspended by writing ‘1’ to **MARD.SPND**. The Flash operation stops when it reaches an interruptible state. After that the flag **FSR.SPND** is set and busy is cleared.

There can be at most one operation in the suspended state. Erroneous suspend and resume requests are detected and prevented by the PMU.

The target range of a suspend program or erase operation is in an undefined state. Reading this range delivers unpredictable results.

The suspended operation can be resumed with the command sequence “Resume Prog/Erase”. The flag **FSR.SPND** is cleared and the busy flag is set again.

The arguments “PA” or “SA” and “nn” of the resume command must be identical to the original argument of the suspended command³⁾. Using a different argument is detected by the PMU and lets the resume fail with **FSR.SQER** and **FSR.SPND** stays set.

When the operation had already finished when receiving the suspend request the request is ignored. The flag **FSR.SPND** is not set. Also **MARD.SPND** is immediately cleared again.

If a resume is requested without any suspended operation **SQER** is set.

Attention: Please ensure that between start or resume of an erase process and the suspend request normally at least ~1 ms erase time can pass. Shorter erase durations are allowed but the erase process will not advance.

In the suspended programming state:

- Reading Flash is allowed.
- New programming and erase commands are rejected with **SQER**.

In the suspended erase state:

- Reading Flash is allowed.
- New erase commands are rejected with **SQER**.
- Programming commands can be performed on any bank.
- However programming in the target range of the suspended erase fails with **SQER**.
- Suspending a programming command is not possible and fails by setting **MARD.SPNDERR**.

1) The “Write Page” operation is suspended between end of the programming and start of the verification. Therefore the programming process itself is not suspended.

2) The “Write Burst” operation in PFlash is suspended during the programming process. In DFlash it is only suspended after end of the programming before starting the verification.

3) For resuming a “Write Burst” the argument “nn” of “Resume Prog/Erase” has to be “00”.

Program Memory Unit (PMU)

In the suspended “Verify Erased Logical Sector Range” state:

- Reading Flash is allowed.
- New erase commands are rejected with SQER.
- Programming commands can be performed on any bank.
- Suspending a programming command is not possible and fails by setting **MARD.SPNDERR**.

10.5.4.5 Programming Voltage Selection

In devices which support 5 V supply via V_{EXT} this control bit allows to select the external supply as source of the programming voltage.

Attention:

When **FCON.PR5V** is “P5V” the programming voltage is directly taken from the external supply V_{EXT} . This shall be supplied with nominal 5V (for allowed tolerances see data sheet).

With **FCON.PR5V** set to “P3V” the programming voltage is internally generated from V_{DDP3} supplied with nominal 3.3 V. Consequently the allowed range of V_{EXT} is in this configuration much larger.

In the PFlash the P5V mode has a significantly shorter duration of the programming process (see data sheet). For the DFlash the P3V mode is recommended as this offers higher robustness against V_{EXT} fluctuations and has the same performance.

10.5.5 Protection

Protection has several aspects: security, safety, preventing maloperation.

The safety protection ensures that reading from PFlash can't be disturbed by software crashes or accesses by “unsafe” masters. Further on it ensures integrity of the read data. This is achieved by the following features:

- Safety ENDINIT protection of configuration registers.
- Master specific protection of PFlash relevant control and configuration registers.
- Protection of these registers against single-event effects “SEEs”.
- Master specific access control for command sequences.
- Integrity of read data is ensured by ECC (see [Chapter 10.5.6](#)).

The security protection prevents unauthorized Flash read and write from internal masters or from external:

- Flash read protection: protected Flash sections can only be read when booting from internal Flash.
- Flash write protection: protected Flash sections can only be changed after authentication.
- Flash OTP (One-Time Programmable) protection: these Flash sections can't be changed anymore.

Program Memory Unit (PMU)

- HSM specific protection: it ensures that the HSM can protect its private data from access by other software. Additionally selected data can only be read during HSM boot phase.
- Device debug protection (see [Chapter 10.5.5.6](#)): depending on the boot mode and the configuration of debug protection and the Flash read protection the debug access to the device is enabled.

The following features protect the Flash against maloperation including software crashes (even of “safe” masters):

- ENDINIT and SV mode protection of configuration registers.
- Command sequences that can change Flash content need more command cycles. SQER stops command interpretation.

Security features supplement safety because Flash data protected by the write protection is also safe against software crashes.

The following table gives an overview of all protection features and for which resources they are effective.

Table 10-7 Protection Layers

	PFlash	DFlash	Registers
Master Specific Access Control	–	–	Restricts Write Accesses
ENDINIT, SV, “Safety ENDINIT”	Restricts Commands (only SE)	–	Restrict Write Accesses
Flash Write Protection	Restricts Commands	Restricts Commands	–
Flash Read Protection	Restricts Read	Restricts Read	–

Any access to these resources has to pass all checks before the operation is performed.

Note: This means that an activation of a protection after start of the operation doesn't affect the operation anymore. Especially the “Safety ENDINIT” can become active as soon as a Flash command (e.g. program or erase) has started.

10.5.5.1 Master Specific Access Control

The master specific access control is configured and controlled by the registers ACCEN0 and ACCEN1 (see [Chapter 10.7.2.1](#)).

Program Memory Unit (PMU)

The registers ACCEN determine which master is allowed to perform writes to the protected resources. In the PMU the following resources are protected by this mechanism:

- Write access to all PMU and Flash registers with following exceptions (see also “Registers Overview” tables [Table 10-16](#) and [Table 10-18](#)):
 - Write access to ACCEN is controlled by the Safety ENDINIT.

Write accesses that are blocked by this mechanism create a bus error as response.

10.5.5.2 Register Access Control

The master unspecific register access control features “Safety ENDINIT”, “ENDINIT”, “Supervisor Mode” are described in the SCU chapter.

In the PMU the following resources are protected by Safety ENDINIT:

- When the Safety ENDINIT is active program and erase commands on the PFlash are rejected with PROER.
- Write access to ACCEN.

10.5.5.3 Effective Flash Read Protection

The Flash read protection depends on the master and the Flash address range.

The following Flash ranges have separate read protections:

- PFlash (with specific handling of the HSM code sectors in PF0).
- And in the DFlash:
 - EEPROMs.
 - UCBs.

PFlash Read Protection

A read access to PFlash (with the exception of sectors S6, S16 and S17 in PF0 of PMU0) fails with bus error under the following conditions:

- Tricore CPU0 PMI (code fetch): FPRO.DCFP.
- Tricore CPU0 DMI (data read): FPRO.DDFP.
- All other masters: FPRO.DDFPX.

The read access to the HSM code sectors S6, S16 and S17 depends on the setting of PROCONHSMCOTP.HSMsX which defines the “HSM_exclusive” attribute of these sectors.

If PROCONHSMCOTP.HSMsX is set for such a sector:

- HSM: full read access.
- All masters except HSM: a read fails with a bus error.
- Cerberus: with enabled HSM debug same access rights as HSM itself, with disabled HSM debug access rights as all other masters.

Program Memory Unit (PMU)

If PROCONHSMCOTP.HSMsX is cleared for such a sector: the same rules as for the other PFlash0 sectors based on PROCONP0 as described above are valid.

DFlash Read Protection

A read access to the DFlash sectors EEPROMx fails with a bus error under the following conditions:

- All masters: FPRO.DDFD.

The read access to the UCBs is described in [Chapter 10.5.5.5](#).

Activating Read Protection

The bits FPRO.DCFP, FPRO.DDFP, FPRO.DDFPX and FPRO.DDFD are initialized by the startup software depending on the configured protection and the startup mode. They can also be directly modified by the user software under conditions noted in the description of [FPRO](#).

10.5.5.4 Effective Flash Write Protection

A range of Flash can be write protected by several means that are all configured in the UCBs.

PFlash Write Protection

Programming or erasing of a group of PFlash sectors (with the exception of sectors S6, S16 and S17 in PF0 of PMU0) fails with PROER if any of the following conditions is true:

- PROCONP0.RPRO and not FPRO.PRODISP (global write protection).
- PROCONPx.SxL and not FPRO.PRODISP (sector specific write protection).
- PROCONOTPx.SxROM (sector specific OTP protection).
- PROCONWOPx.SxWOP (sector specific write-once protection).
 - A set PROCONWOPx.SxWOP prohibits “Write Page”, “Write Burst” and erase commands. These fail with PROER.
 - Only “Write Page Once” is accepted by the PMU. The FSI checks if the addressed Flash range is erased. If this is not the case the FSI reports a PVER and doesn’t start programming.

Programming and erasing of the HSM reserved code sectors S6, S16 and S17 in PF0 of PMU0 depends on the setting of PROCONHSMCOTP.HSMsX which defines the “HSM_exclusive” attribute of these sectors and which master performs the access.

Programming or erase of such sectors fails if any of the following conditions is true:

- Not PROCONHSMCOTP.HSMsX and PROCONP0.RPRO and not FPRO.PRODISP (“global” write protection).

Program Memory Unit (PMU)

- Not PROCONHSMCOTP.HSMsX and PROCONP0.SxL and not FPRO.PRODISP (sector specific write protection).
- PROCONHSMCOTP.SxROM (HSM code sector specific OTP protection).
- For all masters except HSM: PROCONHSMCOTP.HSMsX.

For Cerberus the following rules apply: with enabled HSM debug same access rights as HSM itself, with disabled HSM debug access rights as all other masters.

DFlash Write Protection

Programming and erasing of the DFlash sectors EEPROMx fails with PROER when any of the following conditions is true:

- PROCOND.RPRO and not FPRO.PRODISD.
- PROCOND.L and not FPRO.PRODISD.

Program and erase conditions for the UCBs are described in [Chapter 10.5.5.5](#).

10.5.5.5 Configuring Protection in the UCB

As indicated above the effective protection is determined by the content of the PROCONx registers. These are loaded during startup after each reset from the UCBs. Each UCB comprises 1 KByte of DFlash organized in 128 UC pages of 8 bytes.

An UCB is erased with the command “Erase Logical Sector Range”. It can be programmed with “Write Page” and “Write Burst”. Each UCB has its own access control.

When programming or erasing fail because of this access control PROER is set. When reading fails a bus error is returned.

The following UCBs are defined:

UCB_PFlash

Password protection of the PFlash.

Table 10-8 UCB_PFlash Content

Offset ¹⁾	Content	Description
00 _H	PROCONP0	Read protection for complete PFlash. Write protection for logical sectors of PF0.
04 _H	PROCONP1	Write protection for logical sectors of PF1.
08 _H		Reserved for PROCONP2.
0C _H		Reserved for PROCONP3.
10 _H	PROCONP0	Copy.
14 _H	PROCONP1	Copy.

Program Memory Unit (PMU)
Table 10-8 UCB_PFlash Content (cont'd)

Offset ¹⁾	Content	Description
18 _H		Reserved for copy of PROCONP2.
1C _H		Reserved for copy of PROCONP3.
20 _H	PW0 – PW7	256-bit password, from least significant word to most significant word (8 words).
40 _H	PW0 – PW7	Copy of 256-bit password.
70 _H	Confirmation	4 bytes.
78 _H	Confirmation	Copy.

1) The offsets are given in number of bytes.

See [PROCONPp \(p=0-1\)](#) on [Page 10-67](#).

The confirmed state of this UCB is indicated by FPRO.PROINP.

In the errored state of this UCB also FPRO.PROINP is set. Disabling the protection is not possible in this state.

This UCB is write and read protected if:

- FPRO.PROINP and not FPRO.PRODISP.

UCB_DFlash

Password protection of the DFlash.

This UCB is also used for configuring the PFlash ECC algorithm between the “Safety” and the “Legacy” version, configuring the memory initialization and contains an oscillator configuration.

Table 10-9 UCB_DFlash Content

Offset ¹⁾	Content	Description
00 _H	PROCOND	Read and write protection for the DFlash sectors EEPROMs, oscillator configuration, and enable of RAM initialization.
10 _H	PROCOND	Copy.
20 _H	PW0 – PW7	256-bit password, from least significant word to most significant word (8 words).
40 _H	PW0 – PW7	Copy of 256-bit password.
70 _H	Confirmation	4 bytes.
78 _H	Confirmation	Copy.

Program Memory Unit (PMU)

1) The offsets are given in number of bytes.

See **PROCOND** on **Page 10-68**.

The confirmed state of this UCB is indicated by FPRO.PROIND.

In the errored state of this UCB also FPRO.PROIND is set. Disabling the protection is not possible in this state.

This UCB is write and read protected if:

- FPRO.PROIND and not FPRO.PRODISD.

UCB_OTP

This UCB configures the one-time programmable “OTP” and the write-page once “WOP” protection. It offers the possibility to add this type of protection to Flash sections incrementally.

Table 10-10 UCB_OTP Content

Offset ¹⁾	Content	Description
00 _H	PROCONOTP0	OTP protection for the logical sectors of PF0, and enable of TP.
04 _H	PROCONOTP1	OTP protection for the logical sectors of PF1.
08 _H		Reserved for PROCONOTP2.
0C _H		Reserved for PROCONOTP3.
10 _H	PROCONOTP0	Copy.
14 _H	PROCONOTP1	Copy.
18 _H		Reserved for copy of PROCONOTP2.
1C _H		Reserved for copy of PROCONOTP3.
20 _H	PROCONWOP 0	WOP protection for the logical sectors of PF0.
24 _H	PROCONWOP 1	WOP protection for the logical sectors of PF1.
28 _H		Reserved for PROCONWOP2.
2C _H		Reserved for PROCONWOP3.
30 _H	PROCONWOP 0	Copy.
34 _H	PROCONWOP 1	Copy.

Program Memory Unit (PMU)

Table 10-10 UCB_OTP Content (cont'd)

Offset ¹⁾	Content	Description
38 _H		Reserved for copy of PROCONWOP2.
3C _H		Reserved for copy of PROCONWOP3.
70 _H	Confirmation	4 bytes.
78 _H	Confirmation	Copy.
80 _H		Start of next set.
...		

1) The offsets are given in number of bytes.

See **PROCONOTPp (p=0-1)** on **Page 10-71** and **PROCONWOPp (p=0-1)** on **Page 10-73**.

This configuration set can be stored 8 times. Each set has its own confirmation code. A protection set can be programmed when it is in the “unlocked” state, else it is write-protected.

The content of the PROCONOTP and PROCONWOP registers is initialized by the first configuration set at offset 0 (if this is “unlocked” or “confirmed”). All following confirmed sets are or-ed to this initial value.

An errored set activates all protection bits and activates write protection for this complete UCB.

Attention: This means that always each of the 8 confirmation codes and their copies shall be either in the “confirmed” or “unlocked” state (see [UCB Confirmation on Page 10-36](#)).

The confirmed state of the complete UCB is indicated by FPRO.PROINOTP. It is set when there is at least one protection set in the confirmed state.

If any set is in the errored state FPRO.PROINOTP is also set.

This UCB is protected from erasing if:

- FPRO.PROINOTP.

This UCB is readable.

UCB_IFX

This UCB contains data supplied by Infineon during device production.

Table 10-11 UCB_IFX Content

Offset ¹⁾	Content	Description
00 _H	UID	128-bit Unique ID.
10 _H	Reserved	Reserved.
80 _H		Up to 128 bytes used for DSADC trimming (see DSADC chapter in devices with DSADC).
100 _H		Reserved for calibration data.
200 _H		Reserved for test information data.
380 _H		4 bytes "Test Pass Marker".

1) The offsets are given in number of bytes.

This UCB is read-only. It is readable by every master.

The unique ID is a device unique 128-bit number. Its least significant byte has the value 64_D which identifies an Infineon device.

Note: In order to ensure proper traceability in case of FARs, the unique ID shall be stored by the customer in an adequate database.

The unique ID is supplemented with identification data in the SCU registers CHIPID (device identification and revision, speed grade, Flash size and μ Code revision) and MANID (manufacturer identification). See SCU chapter.

These SCU registers are also sufficient to identify the needed Flash driver (replacement for the QRY record of the CFI standard).

The 4 bytes "Test Pass Marker" contains during test the value FFFFFFFF_H. When all tests have passed it signals this with the value 80658383_H. If customers receive a productive device with a different value in this address they can discard it.

UCB_DBG

This UCB configures the password protection for the debug interface.

Table 10-12 UCB_DBG Content

Offset ¹⁾	Content	Description
00 _H	PROCONDBG	Protection of the debug interface.
10 _H	PROCONDBG	Copy.
20 _H	PW0 – PW7	256-bit password, from least significant word to most significant word (8 words).
40 _H	PW0 – PW7	Copy of 256-bit password.

Program Memory Unit (PMU)
Table 10-12 UCB_DBG Content (cont'd)

Offset ¹⁾	Content	Description
70 _H	Confirmation	4 bytes.
78 _H	Confirmation	Copy.

1) The offsets are given in number of bytes.

See **PROCONDBG** on [Page 10-74](#).

The confirmed state of this UCB is indicated by FPRO.PROINDBG.

In the errored state of this UCB also FPRO.PROINDBG is set. Disabling the protection is not possible in this state.

This UCB is write and read protected if:

- (FPRO.PROINDBG and not FPRO.PRODISDBG) or (PROCONHSMCOTP.DESTDBG = “destructive” and (FDEST = 0 or “not executing Startup Software”)) or PROCONDBG.EDM = “debug entered”.

UCB_HSMCOTP

This UCB configures the OTP and HSM_exclusive protection for the dedicated HSM code sectors S6, S16 and S17 of PF0 of PMU0. It offers the possibility to add this type of protection to Flash sections incrementally.

Table 10-13 UCB_HSMCOTP Content

Offset ¹⁾	Content	Description
00 _H	PROCONHSM COTP	OTP and HSM_exclusive protection for HSM code sectors and configuration of the HSM boot.
10 _H	PROCONHSM COTP	Copy.
70 _H	Confirmation	4 bytes.
78 _H	Confirmation	Copy.
80 _H		Start of next set.
...		

1) The offsets are given in number of bytes.

See **PROCONHSMCOTP** on [Page 10-75](#).

This configuration set can be stored 2 times. Each set has its own confirmation code.

Program Memory Unit (PMU)

The content of the PROCONHSMCOTP register is initialized by the first configuration set at offset 0 (if this is “unlocked” or “confirmed”). All following confirmed sets are or-ed to this initial value.

The confirmed state of the complete UCB is indicated by FPRO.PROINHSMCOTP. It is set when there is at least one protection set in the confirmed state.

If any set is in the errored state FPRO.PROINHSMCOTP is also set. An errored set activates all protection bits.

Attention: This means that always each of the 2 confirmation codes and their copies shall be either in the “confirmed” or “unlocked” state (see UCB Confirmation on Page 10-36).

This UCB is protected from erasing when:

- FPRO.PROINHSMCOTP is set.

The set at offset 0 is protected from programming for all masters when it is in confirmed or errored state.

The set at offset 80_H is protected from programming for all masters except HSM when:

- FPRO.PROINHSMCOTP is set.

It is protected from programming by HSM when it is in the confirmed or errored state¹⁾.

This UCB is readable.

With enabled HSM debug Cerberus has the same access rights as HSM itself. With disabled HSM debug Cerberus is treated as any other master.

UCB_HSM

This UCB can be programmed and erased by all CPUs. However when its content is confirmed or errored only the HSM can modify this UCB.

Its content is only used in devices with activated HSM.

With enabled HSM debug Cerberus has the same access rights as HSM itself. With disabled HSM debug Cerberus is treated as any other master.

Table 10-14 UCB_HSM Content

Offset ¹⁾	Content	Description
00 _H	PROCONHSM	HSM Configuration.
10 _H	PROCONHSM	Copy.

1) Rationale: all masters can configure the protection in delivery state (both sets unlocked). When the first set is confirmed only HSM is allowed to program the second set. When this is also confirmed the complete UCB is OTP protected.

Program Memory Unit (PMU)
Table 10-14 UCB_HSM Content (cont'd)

Offset ¹⁾	Content	Description
70 _H	Confirmation	4 bytes.
78 _H	Confirmation	Copy.

1) The offsets are given in number of bytes.

See **PROCONHSM** on **Page 10-78**.

The confirmed state of this UCB is indicated by FPRO.PROINHSM.

In the errored state of this UCB also FPRO.PROINHSM is set.

This UCB is protected from programming and erasing by non-HSM masters if:

- FPRO.PROINHSM.

It can be read by every master.

Eraser Counter UCB12 – UCB15

These UCBs are used to implement erase counters. From user point of view their content is read-only. They are readable by every master.

For every call of an “Erase Logical Sector Range” the FSI increases automatically the corresponding erase counters until the maximum counting range is reached.

Each UCB12 – UCB15 contains the erase counters for one bank of PFlash PF0 – PF3.

Each logical sector S_x of one PFlash bank has an assigned UCB range of 32 bytes starting at offset address x * 32_D.

The counting is performed by programming the least significant unprogrammed nibble with 1111_B. For reliability reasons a nibble is considered as “unprogrammed” when it contains at most one 1-bit.

Each counter saturates when the last nibble is programmed. Therefore each counter saturates after 64_D erases.

For evaluating the counters the DFlash ECC correction must be disabled with ECCRD.ECCORDIS because this “thermometer code” is not ECC correct.

Remaining UCB8 – UCB11

Currently not used.

These UCBs are read-only.

 Program Memory Unit (PMU)

UCB Confirmation

As described above the UCBs (or sets of a UCB in case of UCB_OTP and UCB_HSMCOTP) can be in the states “confirmed”, “unlocked” and “errored”. This state is determined by the content of the confirmation code:

- 57B5 327F_H: the state is “confirmed”.
- 4321 1234_H: the state is “unlocked”. For over-programming this state (see below) the 4 bytes following the confirmation code need to be kept as 0.
- all other content including uncorrectable ECC errors: the state is “errored”.

An UCB is also in its errored state when at least one of its entries has an uncorrectable error in the original and its copy.

When during evaluation of the UCBs an errored state is detected the flag FSR.PROER is set. The derived values in the PROCONx registers are not set to the Flash content but as indicated in the register description the protection is fully activated. When during startup an errored UCB is detected by the SSW it recognizes this as failed Flash startup and consequently doesn't boot the device.

As also the erased state is considered as “errored” the transition from “unlocked” to “confirmed” state can be done without erasing the UCB. For this the “unlocked” code in the pages with the confirmation code and the copied confirmation code can be over-programmed with 57B5 327F_H. It has to be ensured that the 4 bytes following the confirmation code (e.g. at offset 74_H, 7C_H) are kept 0000 0000_H in the unlocked state and in the over-programmed data. Only then the result after over-programming is ECC clean.

Attention: In case of the multi-set UCBs “UCB_OTP” and “UCB_HSMCOTP” each of the n (8 or 2) confirmation codes and their copies have to be in the state “unlocked” or “confirmed” to prevent an errored state.

10.5.5.6 System Wide Effects of Flash Protection

An active Flash read protection needs to be respected in the complete system.

Additionally the customer can configure in UCB_DBG that OCDS or the debug interface are disabled.

This results in the following lock conditions:

- OCDS disabled: PROCONDBG.OCSDIS and not FPRO.PRODISDBG.
- Debug interface locked: OSTATE.IF_LCK¹⁾ or (PROCONDBG.DBGIFLCK and not FPRO.PRODISDBG).

The SSW performs the following operations:

- The SSW leaves the debug interface locked (OSTATE.IF_LCK stays 1) if any Flash read protection is configured (PROCONP0.RPRO or PROCOND.RPRO are 1).

1) Only for illustration, this bit and its or-combination with the following signals is part of the OCDS not the PMU. The distribution of OCDS documentation is restricted.

Program Memory Unit (PMU)

- If the selected boot mode executes from internal PFlash:
 - The SSW clears the FPRO bits DCFP, DDFP, DDFPX, DDFD.
- If the selected boot mode does not execute from internal PFlash:
 - The SSW sets the FPRO bits DCFP, DDFP, DDFPX, DDFD if their corresponding read protection is configured in PROCONP0.RPRO or PROCOND.RPRO.

Destructive Debug Entry

As described before the debug interface can be opened by supplying the correct password of UCB_DBG to the PMU.

A special destructive debug entry can be configured with **PROCONHSMCOTP.DESTDBG**. When this is configured as “destructive” only the SSW can accept the UCB_DBG password via debug interface (see BootROM chapter). Additionally the password is only accepted when the device pin “FDEST” is logic ‘1’ and the field **PROCONDBG.EDM** is “debug not entered”. When all these conditions match the field **PROCONDBG.EDM** is programmed by the SSW to “debug entered” before opening the debug interface. When EDM is “debug entered” automatically the clock system is manipulated to disallow communication via CAN or Flexray.

10.5.6 Data Integrity and Safety

The data in Flash is stored with error correcting codes “ECC” in order to protect against data corruption. Also data transfer over the SRI bus is protected with ECC. The healthiness of Flash data can be checked with margin checks.

The following measures ensure for reading from PFlash a single point failure metric of >99% and a latent fault metric of >90%.

10.5.6.1 SRI ECC (Safe Fetch Path)

The data transfer over the SRI bus is protected by ECC. This is described in the Safety Concept. The address phase signals are accompanied with an ECC code with Hamming distance 4. The write and read data phase signals are accompanied with an ECC code (also calculated over the address) with Hamming distance 4.

10.5.6.2 Flash ECC

The data in Flash is stored with ECC codes. These are automatically generated when the data is programmed. When data is read these codes are evaluated. Depending on the Flash bank different algorithms with different error correction capabilities are used:

- Data in PFlash uses an ECC code with DEC-TED (Double Error Correction, Triple Error Detection) capabilities. Each block of 256 data bits is accompanied with a set of ECC bits.
 - The ECC algorithm in the PFlash can be switched between a “Safety ECC” which is default and a “Legacy ECC”.

Program Memory Unit (PMU)

- The selection of this algorithm is done by the register field **FCON.NSAFECC**. This is set by the startup software depending on **PROCOND.NSAFECC** but can also be modified by software.
- Data in DFlash uses an ECC code with TEC-QED (Triple Error Correction, Quad Error Detection) capabilities. Each block of 64 data bits is accompanied with a set of ECC bits.

PFlash Safety ECC Details

The PFlash Safety ECC value is calculated over 256 data bits and the address bits 22:5 and the configuration area selection signal. This ECC has the following features:

- Correction of 1-bit and 2-bit errors.
- Detection of 3-bit errors.
- Detection of >99% of all error vectors in the white noise error model.
- Detection of >99% of all-0 and all-1 cases.
- Detection of addressing errors.

As side effect of the all-0 error detection an erased Flash range can't be read without ECC errors. Also over-programming of Flash ranges with all-1 would create entries with ECC errors.

The ECC is automatically generated when programming the PFlash when this is not disabled with **ECCW.PECENCDIS**.

The ECC is automatically evaluated when reading data. Errors are only reported for 256-bit data blocks for which at least one byte is read by a bus master. Errors found in internally pre-fetched data is stored and only reported when this data block is requested by a bus master.

The PMU does not know if the bus master uses the requested data or discards it (e.g. due to speculatively fetching code). Therefore this speculatively fetched data gets the same error response.

Error reporting and ECC disabling:

- Single-bit error:
 - Is noted in **FSR.PFSBER**.
 - Is reported to the SMU (which can trigger an interrupt).
 - Data and ECC value are corrected if this is not disabled with **ECCRp.ECCORDIS¹⁾**.
 - Depending on **CBABCFGp.DIS** and **CBABCFGp.SEL** the affected address is stored uniquely in the **CBAB**.
- Double-bit error:
 - Is noted in **FSR.PFDBER**.

1) Disabling the correction of errors with **ECCORDIS** is a test feature. During application run-time the correction must be enabled.

Program Memory Unit (PMU)

- Is reported to the SMU (which can trigger an interrupt).
- Data and ECC value are corrected if this is not disabled with ECCRPp.ECCORDIS¹⁾.
- Depending on CBABCFGp.DIS and CBABCFGp.SEL the affected address is stored uniquely in the CBAB.
- Triple-bit error (i.e. uncorrectable bit error):
 - Is noted in FSR.PFMBER.
 - Is reported to the SMU (which can trigger an interrupt).
 - Causes a bus error if not disabled by MARP.TRAPDIS.
 - Depending on UBABCFG.DIS and UBABCFG.SEL the affected address is stored in the UBAB.
 - Can additionally report an address error.
- Address error:
 - Is noted in FSR.PFMBER.
 - Is reported to the SMU (which can trigger an interrupt).
 - Causes a bus error if not disabled by MARP.TRAPDIS.
 - Depending on UBABCFG.DIS and UBABCFG.SEL the affected address is stored in the UBAB.
- All-0 and all-1 errors:
 - Are noted in FSR.PFMBER¹⁾.
 - Is reported to the SMU (which can trigger an interrupt).
 - Causes a bus error if not disabled by MARP.TRAPDIS.
 - Depending on UBABCFG.DIS and UBABCFG.SEL the affected address is stored in the UBAB.

PFlash Legacy ECC Details

The “Legacy” PFlash ECC is calculated only over 256 data bits, not over address bits or the configuration selection. Therefore addressing faults (correct data is read from an incorrect address) can not be detected by this algorithm.

This algorithm has the advantage that an erased PFlash range delivers ECC correct 0 data.

Consequently all-0 data and ECC is not reported as an error.

Note: Please note that this algorithm (in contrast to the Auto generation) does not have all-1 ECC bits for all-1 data, i.e. over-programming a page with all-1 will not create an ECC clean data block.

The error detection, reporting and correction capabilities for single-bit, double-bit and triple-bit errors is identical to the Safety ECC.

1) On some addresses additionally FSR.PFDBER is set.

DFlash ECC Details

The DFlash ECC value is calculated over 64 data bits. This ECC has the following features:

- Correction of 1-bit, 2-bit and 3-bit errors.
- Detection of 4-bit errors.
- All-0 (data and ECC) is a correct code vector.
- All-1 (data and ECC) is a valid code vector.
- Address errors are not detected.

An erased data block has an ECC value 0000_H. Therefore an erased sector is free of ECC errors. A data block with all bits '1' has an ECC value of FFFF_H. This is commonly used in EEPROM emulations to over-program data to mark it as "invalid".

Error reporting and ECC disabling:

- Single-bit, double-bit, triple-bit errors:
 - Are noted in FSR.DFSBER, FSR.DFDDBER, FSR.DFTBER respectively.
 - Data and ECC value are corrected if this is not disabled with ECCRD.ECCORDIS¹⁾.
- Quad-bit error:
 - Is noted in FSR.DFMBER.
 - Causes a bus error if not disabled by MARD.TRAPDIS.

10.5.6.3 Margin Checks

The Flash memory offers a "margin check feature": the limit which defines if a Flash cell is read as logic '0' or logic '1' can be shifted. This is controlled by the registers **MARP** and **MARD**.

After changing the read margin at least $t_{FL_MarginDel}$ have to be waited before reading the affected Flash module.

10.5.6.4 PMU and Flash Register Supervision

Safety relevant registers of PMU and Flash are automatically protected against SEUs by the extensive ECC coverage of the complete PFlash read path.

Only the following configuration registers should be supervised by the safety software against accidental modification:

- FCON.STALL: However modification of this affects safety only in case of errors in the application software, when this is reading busy Flash banks.
- MARP.TRAPDIS: Not really safety relevant because all data read errors are anyhow notified to the SMU.

10.5.7 Interrupts and Traps

Generally the PMU reports fatal errors by issuing a bus error which is translated by the CPU into a trap (PMI sets PSE trap and DMI the DSE trap).

This is a list of conditions for reporting a bus error:

- Uncorrectable ECC error (if not disabled by MARP or MARD).
- Write access to read-only register.
- Write access to BootROM.
- Write access to an access controlled register or PFlash address range by a master without allowance by the register access protection ([Chapter 10.5.5.1](#)).
- Not allowed write access to protected register (e.g. SV or ENDINIT).
- Not allowed Flash read access with active read protection.
- Read and write access to all Flash memories when the Flash is in sleep mode.
- Read access to not available Flash memory.
- Read-modify-write access to the Flash memory.
- Read accesses to a busy Flash bank (if not disabled with [FCON.STALL](#)).
- Write access to DF0 address range when normal command interpreter is busy with a command.
- Read or write access to unoccupied register address.

Furthermore selected events can trigger interrupts. The service request nodes SRC_PMUp0/1 are documented in the IR¹⁾.

The following events can trigger the SRC_PMUp0 service request node when enabled by [FCON](#):

- End of busy: any transition of any of the FSR flags P0BUSY, P1BUSY, D0BUSY or D1BUSY from '1' to '0' triggers the interrupt (program and erase sequences, wake-up).
- Operational error: see OPER flag.
- Verify error: see PVER, EVER flags.
- Protection error: see PROER flag.
- Sequence error: see SQER flag.

The event that triggered the interrupt can be determined from the [FSR](#) register.

An interrupt event it also triggered when the event appears again and the corresponding status flag is still set.

The requested read feature ([Chapter 10.5.3.4](#)) signals availability of data with a separate read interrupt via SRC_PMUp1. This can be used to trigger a DMA controller.

1) The parameter "p" in SRC_PMUp0/1 is "0" for PMU0 and "1" for PMU1.

10.5.8 Reset and Startup

The PMU is reset with the application reset with the exception of the following resources:

- Register bits: FSR.PROG, FSR.ERASE, FSR.OPER. These bits are reset with the power-on reset.
- The complete CBAB and UBAB (see [Chapter 10.7.2.11](#) and [Chapter 10.7.2.12](#)) including their status registers. These are also reset with the power-on reset.

The Flash module and the FSI are only reset with the system reset.

When during a Flash operation (program, erase) an application reset occurs the FSI is not reset but the operation is aborted to ensure that the Flash is readable during startup (see [Chapter 10.8.5](#)).

During startup after the application reset the protection setting is installed from the UCBs. If a protection field has an uncorrectable error its copy is used. This is indicated by setting FSR.ORIER. If a copy has also an uncorrectable error the startup software does not enter the user code.

10.5.9 Power Reduction

The Flash module can enter sleep mode to reduce its power consumption. The sleep request can have two sources:

- Global sleep mode requested by the SCU (see “Power Management”). Only executed by the Flash when FCON.ESLDIS = 0.
- Writing ‘1’ to the bit FCON.SLEEP.

After receiving a sleep request the Flash starts the ramp down when the Flash becomes idle, i.e. none of the banks is in command mode and no reads are executed anymore. An ongoing read burst is finished completely. During ramp down to sleep mode all FSR.BUSYx are set.

The sleep mode is indicated in FSR.SLM. The FSR.BUSYx stay set.

The Flash module can be woken up by releasing the sleep request. It enters read mode again after t_{WU} . During the wake-up phase the FSR.BUSYx are set.

Note: It is not recommended to use the SCU controlled sleep mode with FCON.ESLDIS = 0. Because software had to ensure that upon wake-up the Flash is only read after it has returned to read mode. Earlier reads cause bus errors! Therefore the reset value of FCON.ESLDIS is 1.

Note: Requesting sleep mode does not disable automatically an enabled end-of-busy interrupt. When requesting sleep mode during an ongoing Flash operation with enabled end-of-busy interrupt, the operation finishes, the interrupt is issued and then the Flash enters sleep mode. However this end-of-busy interrupt will wake-up the CPU again.

Program Memory Unit (PMU)

An additional power reduction feature is enabled by setting FCON.IDLE. In this case the PFlash read path is switched off when no read access is pending and the read buffers are filled. System performance is slightly reduced because a flash line hit can not be exploited.

10.6 Signaling to the Safety Management Unit (SMU)

The Safety Management Unit “SMU” is described in its own chapter. This section describes the signaling from the PMU to the SMU.

The following error conditions are internally detected and reported to the SMU:

- PMU registers are automatically supervised by the extensive ECC coverage of the PFlash read path (see [Chapter 10.5.6.4](#)). No dedicate alarm is reported to the SMU.
- PFlash detected ECC errors. There are separate signals for corrected single-bit errors, corrected double-bit errors, address errors and uncorrected bit errors.
- CBAB became full.
- SRI ECC errors from address phase, data write phase, data read phase.

The ECC correction logic of each PFlash read path is continuously supervised by the “ECC Monitor”. A detected error is notified to the SMU.

The detection of this error can be checked by disabling the error correction with ECCRPp.ECCORDIS and reading ECC errored data from PFlash. As the correction is disabled the checker reports an error in the correction logic.

The ECC error detection logic of each PFlash read path is also independently continuously supervised by a redundant error detection unit and a comparator “EDC Comparator”. A detected error is notified to the SMU.

The detection of this error can be checked separately for each PFlash “p” error detection logic by setting bit ECCERRPp.EDCERRINJ. An error is injected in the checked detection path.

10.7 Register Set

The register set consists of some general PMU registers which are partly only implemented for PMU0 because they control its specific functionality ([Chapter 10.7.1](#)). The other registers control Flash functionality ([Chapter 10.7.2](#)). Register fields related to DFlash and its sub-sectors are implemented also when these Flash banks or their functionality is not available but they are without function.

The register access conditions use the following abbreviations:

- “U”: Access permitted in User Mode 0 or 1 (applicable to write and read).
- “SV”: Access permitted in Supervisor Mode (applicable to write and read).
- “E”: ENDINIT protected write. “E” means a write access is only allowed before ENDINIT or after disabling this protection with a password as described in the SCU chapter.
- “SE”: Safety ENDINIT protected write.
- “P”: The access is controlled by the master specific register access protection ([Chapter 10.5.5.1](#)).
- “-” or “BE”: Access not permitted.

The PMU and Flash use the following combinations (see [Table 10-16](#) and [Table 10-18](#)):

- U, SV: access always allowed (i.e. in user mode or supervisor mode).
- P, U, SV: access always to masters allowed that are enabled by the register access protection ([Chapter 10.5.5.1](#)).
- P, SV: access only in supervisor mode for masters enabled by the register access protection.
- BE: access never allowed, causing a bus error.
- SV, E: access only in supervisor mode with disabled ENDINIT protection.
- P, SV, E: access only in supervisor mode with disabled ENDINIT protection for masters enabled by the register access protection.
- SV, SE: access only in supervisor mode with disabled Safety ENDINIT protection.

All accesses prevented due to these restrictions fail with a bus error.

Also accesses to unoccupied register addresses fail with a bus error.

Note: It is convention to use short register names (e.g. “FCON”) in the chapter that defines these registers. In all other chapters and in the development tools long register names are used that are a concatenation of the module instance (e.g. “PMU0” or “FLASH0”), an underscore and the short register name, i.e. “FLASH0_FCON”. This document uses for clarification also mostly the long names.

10.7.1 PMU Registers

Non-Flash related registers for the PMU. They have the prefix “PMUx_”.

Table 10-15 Registers Address Space

Module	Base Address	End Address	Note
PMU0	F800 0500 _H	F800 052B _H	

Table 10-16 Registers Overview

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset	Page
			Read	Write		
ID	Module Identification	08 _H	U, SV	BE	Application Reset	10-45

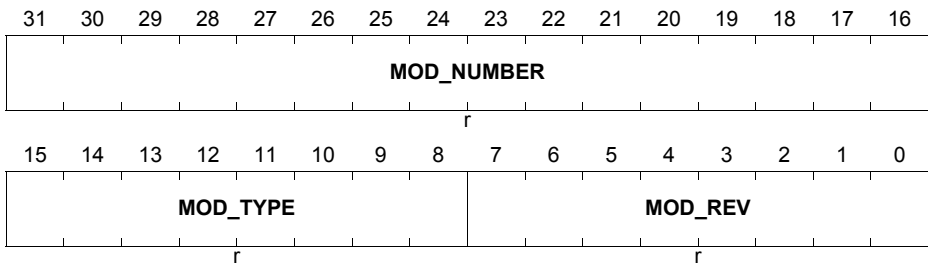
1) The absolute register address is calculated as follows:
 Module Base Address (Table 10-15) + Offset Address (shown in this column)

10.7.1.1 PMU Identification

The PMU_ID register identifies the PMU and its version.

PMU0_ID

PMU0 Identification Register (F800 0508_H) **Reset Value: 00D8 C0XX_H**



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number MOD_REV defines the module revision number. The value of a module revision starts with 01 _H (first rev.).

Program Memory Unit (PMU)

Field	Bits	Type	Description
MOD_TYPE	[15:8]	r	Module Type This bit field is C0 _H . It defines the module as a 32-bit module.
MOD_NUMBER	[31:16]	r	Module Number Value This bit field defines the module identification number for PMU0.

Program Memory Unit (PMU)
10.7.2 Flash Registers

The absolute address of a Flash register is calculated by the base address from [Table 10-17](#) plus the offset address of this register from [Table 10-18](#).

Table 10-17 Registers Address Space

Module	Base Address	End Address	Note
FLASH0	F800 1000 _H	F800 23FF _H	Flash registers of PMU0

Table 10-18 Registers Overview

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset	Page
			Read	Write		
COMM0	FSI Communication 0	0000 _H	U, SV	P, SV	Application Reset	10-95
COMM1	FSI Communication 1	0004 _H	U, SV	P, SV	Application Reset	10-95
COMM2	FSI Communication 2	0008 _H	U, SV	P, SV	Application Reset	10-95
ID	Flash Module Identification Register	1008 _H	U, SV	BE	Application Reset	10-52
FSR	Flash Status Register	1010 _H	U, SV	P, U, SV	Application Reset ²⁾	10-53
FCON	Flash Configuration Register	1014 _H	U, SV	P, SV, E	Application Reset	10-59
FPRO	Flash Protection Control and Status	101C _H	U, SV	P, SV, E	Application Reset	10-63
PROCONP0	PFlash Protection Config. PF0	1020 _H	U, SV	BE	Application Reset	10-67
PROCONP1	PFlash Protection Config. PF1	1024 _H	U, SV	BE	Application Reset	10-67
PROCOND	DFlash Protection Config.	1030 _H	U, SV	BE	Application Reset	10-68
PROCONH SMCOTP	HSM Code Flash OTP Protection Config.	1034 _H	U, SV	BE	Application Reset	10-75
PROCONO TP0	OTP Protection Config. PF0	1038 _H	U, SV	BE	Application Reset	10-71

Program Memory Unit (PMU)
Table 10-18 Registers Overview (cont'd)

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset	Page
			Read	Write		
PROCONO TP1	OTP Protection Config. PF1	103C _H	U, SV	BE	Application Reset	10-71
PROCONW OP0	Write-Once Protection Config. PF0	1048 _H	U, SV	BE	Application Reset	10-73
PROCONW OP1	Write-Once Protection Config. PF1	104C _H	U, SV	BE	Application Reset	10-73
PROCOND BG	Debug Interface Protection Config.	1058 _H	U, SV	BE	Application Reset	10-74
PROCONH SM	HSM Exclusive Config.	105C _H	U, SV	BE	Application Reset	10-78
RDBCFCG00	Read Buffer Port 0 Configuration 0	1060 _H	U, SV	P, SV, E	Application Reset	10-79
RDBCFCG01	Read Buffer Port 0 Configuration 1	1064 _H	U, SV	P, SV, E	Application Reset	10-79
RDBCFCG02	Read Buffer Port 0 Configuration 2	1068 _H	U, SV	P, SV, E	Application Reset	10-79
RDBCFCG10	Read Buffer Port 1 Configuration 0	106C _H	U, SV	P, SV, E	Application Reset	10-79
RDBCFCG11	Read Buffer Port 1 Configuration 1	1070 _H	U, SV	P, SV, E	Application Reset	10-79
RDBCFCG12	Read Buffer Port 1 Configuration 2	1074 _H	U, SV	P, SV, E	Application Reset	10-79
ECCW	ECC Write Register	1090 _H	U, SV	P, SV, E	Application Reset	10-83
ECCRP0	ECC Read Register PF0	1094 _H	U, SV	P, SV, E	Application Reset	10-84
ECCRP1	ECC Read Register PF1	1098 _H	U, SV	P, SV, E	Application Reset	10-84
ECCRD	ECC Read Register DF	10A4 _H	U, SV	P, SV, E	Application Reset	10-84
MARP	Margin Control Register PFlash	10A8 _H	U, SV	P, SV, E	Application Reset	10-85

Program Memory Unit (PMU)
Table 10-18 Registers Overview (cont'd)

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset	Page
			Read	Write		
MARD	Margin Control Register DFlash	10AC _H	U, SV	P, U, SV	Application Reset	10-86
CBABCFG0	Corrected Bits Address Buffer Configuration Port 0	10B4 _H	U, SV	P, SV, E	Power-On Reset	10-88
CBABSTAT 0	Corrected Bits Address Buffer Status Port 0	10B8 _H	U, SV	BE	Power-On Reset	10-89
CBABTOP0	Corrected Bits Address Buffer Top Port 0	10BC _H	U, SV	P, SV	Power-On Reset	10-90
CBABCFG1	Corrected Bits Address Buffer Configuration Port 1	10C0 _H	U, SV	P, SV, E	Power-On Reset	10-88
CBABSTAT 1	Corrected Bits Address Buffer Status Port 1	10C4 _H	U, SV	BE	Power-On Reset	10-89
CBABTOP1	Corrected Bits Address Buffer Top Port 1	10C8 _H	U, SV	P, SV	Power-On Reset	10-90
UBABCFG0	Uncorrectable Bits Address Buffer Configuration Port 0	10E4 _H	U, SV	P, SV, E	Power-On Reset	10-91
UBABSTAT 0	Uncorrectable Bits Address Buffer Status Port 0	10E8 _H	U, SV	BE	Power-On Reset	10-92
UBABTOP0	Uncorrectable Bits Address Buffer Top Port 0	10EC _H	U, SV	P, SV	Power-On Reset	10-93
UBABCFG1	Uncorrectable Bits Address Buffer Configuration Port 1	10F0 _H	U, SV	P, SV, E	Power-On Reset	10-91
UBABSTAT 1	Uncorrectable Bits Address Buffer Status Port 1	10F4 _H	U, SV	BE	Power-On Reset	10-92
UBABTOP1	Uncorrectable Bits Address Buffer Top Port 1	10F8 _H	U, SV	P, SV	Power-On Reset	10-93

Program Memory Unit (PMU)
Table 10-18 Registers Overview (cont'd)

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset	Page
			Read	Write		
RRCT	Requested Read Control Register	1140 _H	U, SV	U, SV	Application Reset	10-80
RRD0	Requested Read Data Register 0	1144 _H	U, SV	BE	Application Reset	10-81
RRD1	Requested Read Data Register 1	1148 _H	U, SV	BE	Application Reset	10-81
RRAD	Requested Read Address Register	114C _H	U, SV	U, SV	Application Reset	10-82
Res. HSM registers	Reserved for HSM command interface registers.	1200 _H – 121B _H	H	H	Application Reset	
ACCEN1	Access Enable Register 1	13F8 _H	U, SV	SV, SE	Application Reset	10-51
ACCEN0	Access Enable Register 0	13FC _H	U, SV	SV, SE	Application Reset	10-50

1) The absolute register address is calculated as follows:

Module Base Address ([Table 10-17](#)) + Offset Address (shown in this column)

2) Some bits are not reset with the application reset but only with the power-on reset.

10.7.2.1 Master Specific Access Control

The master specific access control is described in [Chapter 10.5.5.1](#).

Access Enable Register 0

The Access Enable Register 0 controls write access for transactions with the on-chip bus master TAG IDs 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID to master peripheral mapping). The PMU is prepared for a 6 bit TAG ID. The registers ACCEN0 and ACCEN1 provide one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ..., EN31 -> TAG ID 011111_B.

Program Memory Unit (PMU)

ACCEN0

Access Enable Register 0

(13FC_H)

Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
ENx (x = 0-31)	x	rW	<p>Access Enable for Master TAG ID x</p> <p>This bit enables write access to the protected resources (see Chapter 10.5.5.1) for transactions with the corresponding Master TAG ID</p> <p>0_B Write access will not be executed</p> <p>1_B Write access will be executed</p>

Access Enable Register 1

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID to master peripheral mapping). The PMU is prepared for a 6 bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

ACCEN1

Access Enable Register 1

(13F8_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES															
r															

Field	Bits	Type	Description
RES	[31:0]	r	Reserved Read as 0; should be written with 0.

10.7.2.2 Flash Identification Register

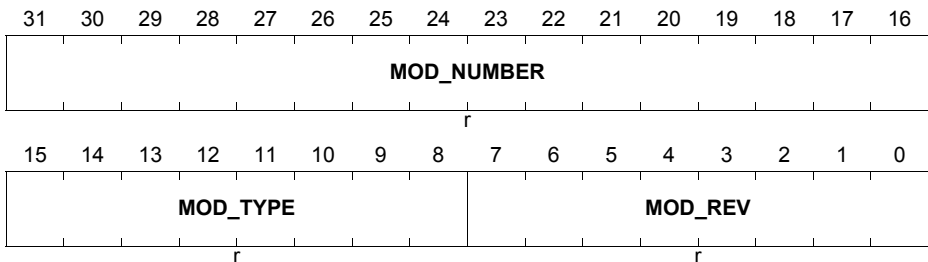
The register identifies the Flash module which can have a different version from the PMU.

Flash Identification

ID

Flash Module Identification Register (1008_H)

Reset Value: 00D9 C0XX_H



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number MOD_REV defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MOD_TYPE	[15:8]	r	Module Type This bit field is C0 _H . It defines the module as a 32-bit module.
MOD_NUMBER	[31:16]	r	Module Number Value This bit field defines a module identification number. For the TC26x B-Step Flash0 this number is 00D9 _H .

10.7.2.3 Flash Status

The Flash Status Register FSR reflects the overall status of the Flash module after Reset and after reception of the different commands.

Program Memory Unit (PMU)

The error flags and the two status flags (PROG, ERASE) are affected by the “Clear Status” command. The error flags are also cleared with the “Reset to Read” command. Some error flags are marked as writable. These flags can be cleared by writing a ‘1’ to this bit.

Flash Status Register
FSR
Flash Status Register
(1010_H)
Reset Value: 0100 0000_H¹⁾

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	ORIER	0	SLM	SPND	EVE R	PVE R	RES	0	SRIADDE RR	DFM BER	DFT BER	DFD BER	DFS BER	RES 17	PFM BER
r	rh	r	rh	rwh	rwh	rwh	rwh	r	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PFD BER	PFS BER	PRO ER	SQ ER	OPE R	DF PAGED	PF PAGED	ERA SE	PRO G	RES 6	RES 5	P1B USY	P0B USY	RES 1	D0 BUS Y	FA BUS Y
rwh	rwh	rwh	rwh	rwh	rh	rh	rwh	rwh	rh	rh	rh	rh	rh	rh	rh

1) The startup process might report errors by setting some of the error flags.

Field	Bits	Type	Description
FABUSY	0	rh	Flash Array Busy¹⁾ Internal busy flag for testing purposes. Must be ignored by application software. This may only use PxBUSY and DxBUSY.
D0BUSY	1	rh	Data Flash Bank 0 Busy¹⁾ HW-controlled status flag. 0 _B DFlash0 ready, not busy; DFlash0 in read mode. 1 _B DFlash0 busy; DFlash0 not in read mode. Indication of busy state of DFlash bank 0 because of active execution of an operation; DFlash0 busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state the DFlash0 is not in read mode.
RES1	2	rh	Reserved for Data Flash Bank 1 Busy¹⁾

Program Memory Unit (PMU)

Field	Bits	Type	Description
P0BUSY	3	rh	Program Flash PF0 Busy¹⁾ HW-controlled status flag. 0 _B PF0 ready, not busy; PF0 in read mode. 1 _B PF0 busy; PF0 not in read mode. Indication of busy state of PF0 because of active execution of an operation; PF0 busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state, the PF0 is not in read mode.
P1BUSY	4	rh	Program Flash PF1 Busy¹⁾ HW-controlled status flag. 0 _B PF1 ready, not busy; PF1 in read mode. 1 _B PF1 busy; PF1 not in read mode. Indication of busy state of PF1 because of active execution of an operation; PF1 busy state is also indicated during Flash startup after reset or in sleep mode; while in busy state, the PF1 is not in read mode.
RES5	5	rh	Reserved for Program Flash PF2 Busy¹⁾
RES6	6	rh	Reserved for Program Flash PF3 Busy¹⁾
PROG	7	rwh	Programming State²⁾³⁾ HW-controlled status flag. 0 _B There is no program operation requested or in progress or just finished. 1 _B Programming operation requested or in action or finished. Set with last cycle of Write Page/Burst command sequence, cleared with Clear Status command (if not busy) or with power-on reset. If one BUSY flag is coincidentally set, PROG indicates the type of busy state. If OPER is coincidentally set, PROG indicates the type of erroneous operation. Otherwise, PROG indicates, that operation is still requested or finished. Can be also cleared by writing '1' to it.

Program Memory Unit (PMU)

Field	Bits	Type	Description
ERASE	8	rwh	<p>Erase State ²⁾³⁾</p> <p>HW-controlled status flag.</p> <p>0_B There is no erase operation requested or in progress or just finished</p> <p>1_B Erase/Verify operation requested or in action or finished.</p> <p>Set with last cycle of Erase/Verify command sequence, cleared with Clear Status command (if not busy) or with power-on reset. Indications are analogous to PROG flag.</p> <p>Can be also cleared by writing '1' to it.</p>
FFPAGE	9	rh	<p>Program Flash in Page Mode ¹⁾⁴⁾</p> <p>HW-controlled status flag.</p> <p>0_B Program Flash not in page mode.</p> <p>1_B Program Flash in page mode.</p> <p>Set with Enter Page Mode for PFlash, cleared with Write Page command</p> <p><i>Note: Concurrent page and read modes are allowed</i></p>
DFFPAGE	10	rh	<p>Data Flash in Page Mode ¹⁾⁴⁾</p> <p>HW-controlled status flag.</p> <p>0_B Data Flash not in page mode</p> <p>1_B Data Flash in page mode</p> <p>Set with Enter Page Mode for DFlash, cleared with Write Page command.</p> <p><i>Note: Concurrent page and read modes are allowed</i></p>
OPER	11	rwh	<p>Flash Operation Error ²⁾³⁾⁴⁾</p> <p>0_B No operation error.</p> <p>1_B Flash array operation aborted, because of a Flash array failure, e.g. an ECC error in microcode.</p> <p>This bit is not cleared with application reset, but with power-on reset.</p> <p>Registered status bit; must be cleared per command or by writing '1'.</p>

Program Memory Unit (PMU)

Field	Bits	Type	Description
SQER	12	rwh	<p>Command Sequence Error¹⁾²⁾⁴⁾</p> <p>0_B No sequence error</p> <p>1_B Command state machine operation unsuccessful because of improper address or command sequence.</p> <p>A sequence error is not indicated if the Reset to Read command aborts a command sequence.</p> <p>Registered status bit; must be cleared per command or by writing '1'.</p>
PROER	13	rwh	<p>Protection Error¹⁾²⁾⁴⁾</p> <p>0_B No protection error</p> <p>1_B Protection error.</p> <p>A Protection Error is reported e.g. because of a not allowed command, for example an Erase or Write Page command addressing a locked sector, or because of wrong password(s) in a protected command sequence such as "Disable Read Protection".</p> <p>A Protection Error is also reported if the safety protection (Chapter 10.5.5.2) prevented a program/erase operation in PFlash.</p> <p>Registered status bit; must be cleared per command or by writing '1'.</p>
PFSBER	14	rwh	<p>PFlash Single-Bit Error and Correction¹⁾²⁾⁴⁾</p> <p>0_B No Single-Bit Error detected during read access to PFlash.</p> <p>1_B Single-Bit Error detected.</p> <p>Registered status bit; must be cleared per command or by writing '1'.</p>
PFDBER	15	rwh	<p>PFlash Double-Bit Error¹⁾²⁾⁴⁾</p> <p>0_B No Double-Bit Error detected during read access to PFlash.</p> <p>1_B Double-Bit Error detected in PFlash.</p> <p>Registered status bit; must be cleared per command or by writing '1'.</p>

Program Memory Unit (PMU)

Field	Bits	Type	Description
PFMBER	16	rwh	PFlash Uncorrectable Error¹⁾²⁾⁴⁾ 0 _B No uncorrectable error (3-bit error or more, address error, all-0/all-1) detected during read access to PFlash. 1 _B Uncorrectable error (3-bit error or more, address error, all-0/all-1) detected in PFlash. Registered status bit; must be cleared per command or by writing '1'.
RES17	17	rwh	Reserved¹⁾²⁾⁴⁾ Registered status bit; must be cleared per command or by writing '1'.
DFSBER	18	rwh	DFlash Single-Bit Error¹⁾²⁾⁴⁾ 0 _B No Single-Bit Error detected during read access to DFlash. 1 _B Single-Bit Error detected. Registered status bit; must be cleared per command or by writing '1'.
DFDBER	19	rwh	DFlash Double-Bit Error¹⁾²⁾⁴⁾ 0 _B No Double-Bit Error detected during read access to DFlash. 1 _B Double-Bit Error detected. Registered status bit; must be cleared per command or by writing '1'.
DFTBER	20	rwh	DFlash Triple-Bit Error¹⁾²⁾⁴⁾ 0 _B No Triple-Bit Error detected during read access to DFlash. 1 _B Triple-Bit Error detected. Registered status bit; must be cleared per command or by writing '1'.
DFMBER	21	rwh	DFlash Uncorrectable Error¹⁾²⁾⁴⁾ 0 _B No uncorrectable error detected during read access to DFlash 1 _B Uncorrectable error detected in DFlash. Registered status bit; must be cleared per command or by writing '1'.

Program Memory Unit (PMU)

Field	Bits	Type	Description
SRIADDERR	22	rwh	SRI Bus Address ECC Error¹⁾²⁾⁴⁾ This flag is set when the PMU detects an ECC error in the address phase bus transaction on the SRI bus. 0 _B No SRI address error detected. 1 _B SRI address error detected. Registered status bit; must be cleared by writing '1'.
RES	24	rwh	Reserved Write zero. Read value unpredictable.
PVER	25	rwh	Program Verify Error¹⁾²⁾⁴⁾ A verify error was reported during a Flash program operation. 0 _B The page is correctly programmed. All bits have full expected quality. 1 _B A program verify error has been detected. Full quality of all bits cannot be guaranteed. Registered status bit; must be cleared per command or writing '1'.
EVER	26	rwh	Erase Verify Error¹⁾²⁾⁴⁾ A verify error was reported during a Flash erase operation. 0 _B The sector is correctly erased. All erased bits have full expected quality. 1 _B An erase verify error has been detected. Full quality erased bits cannot be guaranteed. Registered status bit; must be cleared per command or writing '1'.
SPND	27	rwh	Operation Suspended³⁾ Requested by MARD.SPND a program or erase operation is suspended. 0 _B No Flash operation is suspended. 1 _B Suspended Flash operation. Can be also cleared by writing '1' to it to clean up after a reset that killed a suspended operation context.

Program Memory Unit (PMU)

Field	Bits	Type	Description
SLM	28	rh	Flash Sleep Mode¹⁾ HW-controlled status flag. Indication of Flash sleep mode taken because of global or individual sleep request; additionally indicates when the Flash is in shut down mode. 0 _B Flash not in sleep mode 1 _B Flash is in sleep or shut down mode
ORIER	30	rh	Original Error¹⁾²⁾⁴⁾ 0 _B No original error detected during startup. 1 _B Original data replaced by its copy.
0	31, 29, 23	r	Reserved Read zero, no write

- 1) Cleared with application reset
- 2) Cleared with command "Clear Status"
- 3) Cleared with power-on reset (PORST)
- 4) Cleared with command "Reset to Read"

Note: The xBUSY flags cannot be cleared with the "Clear Status" command or with the "Reset to Read" command. These flags are controlled by HW.

10.7.2.4 Flash Configuration Control

The Flash Configuration Register FCON reflects and controls the general Flash configuration functions.

FCON

Flash Configuration Register
(1014_H)
Reset value: 0091 XXXX_H¹⁾

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EOB M	PR5 V	0			PRO ERM	SQ ERM	VOP ERM	RES23		RES21		STA LL	NSA FEC C	SLE EP	ESL DIS
rw	rw	r			rw	rw	rw	rh		rh		rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDLE		WSECDF			WSDFLASH				WSECPF			WSPFLASH			
rw		rw			rw				rw			rw			

Program Memory Unit (PMU)

- 1) The wait-cycles WSECDF, WSDFLASH, WSECPF and WSPFLASH are changed by the startup after system and power-on resets. **Attention: the configured value is only sufficient for the clock configuration used during startup.** The wait-cycles have to be configured after startup as described in [Chapter 10.5.3.3](#) before changing to higher clock frequencies.

Field	Bits	Type	Description
WSPFLASH	[3:0]	rw	<p>Wait States for read access to PFlash</p> <p>This bitfield defines the number of wait states in number of FSI2 clock cycles, which are used for an initial read access to the Program Flash memory area (see Chapter 10.5.3.3).</p> <p>0_D Read access takes 1 FSI2 clock cycle. 1_D Read access takes 2 FSI2 clock cycles. ..._D ...</p>
WSECPF	[5:4]	rw	<p>Wait States for Error Correction of PFlash</p> <p>This bitfield defines the number of wait-states for the ECC correction in FSI2 clock cycles (see Chapter 10.5.3.3).</p> <p>0_D Error correction takes 1 FSI2 clock cycle. 1_D Error correction takes 2 FSI2 clock cycles. ..._D ...</p>
WSDFLASH	[11:6]	rw	<p>Wait States for read access to DFlash</p> <p>This bitfield defines the number of wait states in number of FSI clock cycles, which are used for a read access to the Data Flash memory area (see Chapter 10.5.3.3).</p> <p>0_D Read access takes 1 FSI clock cycle. 1_D Read access takes 2 FSI clock cycles. ..._D ...</p>
WSECDF	[14:12]	rw	<p>Wait State for Error Correction of DFlash</p> <p>This bitfield defines the number of wait-states for the ECC correction in FSI clock cycles (see Chapter 10.5.3.3).</p> <p>0_D Error correction takes 1 FSI clock cycle. 1_D Error correction takes 2 FSI clock cycles. ..._D ...</p>

Program Memory Unit (PMU)

Field	Bits	Type	Description
IDLE	15	rw	Dynamic Flash Idle 0_B Normal/standard Flash read operation 1_B Dynamic idle of Program Flash enabled for power saving; static prefetching disabled <i>Note: In Data Flash, dynamic idle is always enabled (prefetching not supported).</i> Initial value after startup is 0_B .
ESLDIS	16	rw	External Sleep Request Disable 0_B External sleep request signal input is enabled 1_B Externally requested Flash sleep is disabled The 'external' signal input is connected with a global power-down/sleep request signal from SCU.
SLEEP	17	rw	Flash SLEEP 0_B Normal state or wake-up 1_B Flash sleep mode is requested, Wake-up from sleep is started with clearing of the SLEEP-bit.
NSAFECC	18	rw	Non-Safety PFlash ECC This bit selects if the data in PFlash is written and read with the "Safety" ECC code (calculated over address bits), or the "Legacy" ECC code (see Chapter 10.5.6). 0_B Safety ECC is used. 1_B Legacy ECC is used. Attention: this bit must not be changed while accessing Flash (reading, programming, erasing). Initial value after startup depends on PROCOND.NSAFECC.

Program Memory Unit (PMU)

Field	Bits	Type	Description
STALL	19	rw	<p>Stall SRI</p> <p>This field selects if reading from busy Flash banks causes a bus error or wait-states until busy is cleared again.</p> <p>1_B Stall, Reading busy Flash banks suppresses the SRI bus ready effectively causing wait-states until busy is cleared.</p> <p>0_B Error, Reading busy Flash banks causes a bus error.</p> <p><i>Note: This field must not be changed while any bank is busy. The results are unpredictable. Generally it's strongly recommended to configure this field once and avoid changing it during operation.</i></p> <p><i>Note: Reading Flash in sleep mode causes always a bus error independent of this field (although it reports "busy").</i></p>
RES21	[21:20]	rh	<p>Reserved</p> <p>Write 0; Read value unpredictable.</p>
RES23	[23:22]	rh	<p>Reserved</p> <p>Write 0; Read value unpredictable.</p>
VOPERM	24	rw	<p>Verify and Operation Error Interrupt Mask</p> <p>0_B Interrupt not enabled</p> <p>1_B Flash interrupt because of Verify Error or Operation Error in Flash array (FSI) is enabled</p>
SQERM	25	rw	<p>Command Sequence Error Interrupt Mask</p> <p>0_B Interrupt not enabled</p> <p>1_B Flash interrupt because of Sequence Error is enabled</p>
PROERM	26	rw	<p>Protection Error Interrupt Mask</p> <p>0_B Interrupt not enabled</p> <p>1_B Flash interrupt because of Protection Error is enabled</p>
EOBM	31	rw	<p>End of Busy Interrupt Mask</p> <p>0_B Interrupt disabled.</p> <p>1_B EOB interrupt is enabled.</p>

Program Memory Unit (PMU)

Field	Bits	Type	Description
PR5V	30	rw	Programming Supply 5V Selects the supply for programming. 0 _B P3V , The programming voltage is internally generated. 1 _B P5V , As programming voltage the external 5 V supply is used.
0	[29:27]	r	Reserved Write 0; Read 0.

Note: The default numbers of wait states represent the slow cases. This is a general proceeding and additionally opens the possibility to execute higher frequencies without changing the configuration.

10.7.2.5 Flash Protection

Flash Protection Control and Status Register

This register reports the state of the Flash protection (see [Chapter 10.5.5.5](#)) and contains protection relevant control fields.

FPRO

Flash Protection Control and Status Register (101C_H)

Reset Value: 00XX 00A0_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			0					ENPE		0	DDF D	0	DDF PX	DDF P	DCF P
			r					rw		r	rwh	r	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			0		PROI N H S M	PRO DIS D B G	PROI N D B G	RES 7	PROI N O T P	RES 5	PROI N H S M C O T P	PRO DIS D	PROI N D	PRO DIS P	PROI N P
			r		rh	rh	rh	r	rh	r	rh	rh	rh	rh	rh

Field	Bits	Type	Description
PROINP	0	rh	PFlash Protection This bit reflects the confirmed state of UCB_PFlash.

Program Memory Unit (PMU)

Field	Bits	Type	Description
PRODISP	1	rh	PFlash Protection Disabled¹⁾ The protection configured by UCB_PFlash was successfully disabled by supplying the correct password to “Disable Protection”.
PROIND	2	rh	DFlash Protection This bit reflects the confirmed state of UCB_DFlash.
PRODISD	3	rh	DFlash Protection Disabled¹⁾ The protection configured by UCB_DFlash was successfully disabled by supplying the correct password to “Disable Protection”.
PROINHSMCOT P	4	rh	HSM OTP Protection This bit reflects the confirmed state of UCB_HSMCOTP.
RES5	5	rh	Reserved Read value don't care, updated during startup; should be written with 0.
PROINOTP	6	rh	OTP and Write-Once Protection This bit reflects the confirmed state of UCB_OTP.
RES7	7	rh	Reserved Read value don't care, updated during startup; should be written with 0.
PROINDBG	8	rh	Debug Interface Password Protection This bit reflects the confirmed state of UCB_DBG.
PRODISDBG	9	rh	Debug Interface Password Protection Disabled¹⁾ The password configured by UCB_DBG was correctly received with “Disable Protection”. When PROCONHSMCOTP.DESTDBG is “destructive” then only the SSW can disable this protection.
PROINHSM	10	rh	HSM Configuration This bit reflects the confirmed state of UCB_HSM.

Program Memory Unit (PMU)

Field	Bits	Type	Description
DCFP	16	rwh	<p>Disable Code Fetch from PFlash Memory for CPU0 PMI</p> <p>This bit enables/disables the code fetch from the internal PFlash memory area for CPU0 PMI (see Chapter 10.5.5.3). Once set, this bit can only be cleared when FPRO.PRODISP or not PROCONP0.RPRO.</p> <p>This bit is automatically set with reset and is cleared during startup, if no RP installed, and during startup (BootROM SW) in case of internal start out of Flash.</p> <p>0_B Code fetching by CPU0 from the PFlash memory area is allowed.</p> <p>1_B Code fetching by CPU0 from the PFlash memory area is not allowed.</p>
DDFP	17	rwh	<p>Disable Read from PFlash for CPU0 DMI</p> <p>This bit enables/disables the read access to the PFlash memory area by for CPU0 DMI (see Chapter 10.5.5.3). Once set, this bit can only be cleared when FPRO.PRODISP or not PROCONP0.RPRO.</p> <p>This bit is automatically set with reset and is cleared during startup, if no RP installed, and during startup (BootROM SW) in case of internal start out of Flash.</p> <p>0_B Read access to the PFlash memory area is allowed.</p> <p>1_B Read access to the PFlash memory area is not allowed.</p>
DDFPX	18	rwh	<p>Disable Read from PFlash for Other Masters</p> <p>This bit enables/disables the read access to PFlash for other masters (see Chapter 10.5.5.3). Once set, this bit can only be cleared when FPRO.PRODISP or not PROCONP0.RPRO.</p> <p>This bit is automatically set with reset and is cleared during startup, if no RP installed, and during startup (BootROM SW) in case of internal start out of Flash.</p> <p>0_B The read access by other masters to the PFlash memory area is allowed.</p> <p>1_B The read access to the PFlash memory area is not allowed for other masters.</p>

Program Memory Unit (PMU)

Field	Bits	Type	Description
DDFD	20	rwh	<p>Disable Data Fetch from DFlash Memory</p> <p>This bit enables/disables the data fetch from the internal DFlash memory area (see Chapter 10.5.5.3). Once set, this bit can only be cleared when FPRO.PRODISD or not PROCOND.RPRO.</p> <p>This bit is automatically set with reset and is cleared during startup, if no RP installed, and during startup (BootROM SW) in case of internal start out of Flash.</p> <p>0_B Data fetching from the DFlash memory area is allowed.</p> <p>1_B Data fetching from the DFlash memory area is not allowed.</p>
ENPE	[23:22]	rw	<p>Enable Program/Erase</p> <p>The field prevents any Flash program or erase directly in the FSI. It is set to Enabled by the SSW upon finishing the startup.</p> <p>01_B Enabled, Program/Erase are enabled in the FSI.</p> <p>10_B Disabled, Program/Erase are disabled in the FSI.</p> <p>others: Illegal values, disabled also Program/Erase. Once in “Enabled” state this field can’t be changed anymore.</p>
0	[31:24], 21, 19, [15:11]	r	<p>Reserved</p> <p>Always read as 0; should be written with 0.</p>

1) Cleared with command “Resume Protection”.

Notes

After reset and execution of BootROM startup SW, the read protection control bits are set to the following values:

- Start not in internal Flash:
 - DCFP set to PROCONP0.RPRO.
 - DDFP set to PROCONP0.RPRO.
 - DDFPX set to PROCONP0.RPRO.
 - DDFD set to PROCOND.RPRO.

Program Memory Unit (PMU)

- Start in internal Flash:
 - DCFP, DDFP, DDFPX, DDFD set to 0.
- Independent of the start configuration: the PROIN fields are set depending on the content of their assigned UCB sectors.

Attention! Before disabling read access by setting any of the DCF* and DDF* flags all pending read accesses to the affected Flash ranges should have finished.

10.7.2.6 Protection Configuration

The configuration of read/write/OTP protection is indicated with registers PROCONP, PROCOND, PROCONHSMCOTP, PROCONOTP, PROCONWOP, PROCONHSM and PROCONDBG.

PFlash Protection Configuration

When UCB_PFlash is in errored state the initial value is “FFFF FFFF_H”, else the initial value after Flash startup reflects the UCB_PFlash content ([Table 10-8](#)).

PROCONPp (p=0-1)
PFlash Protection Configuration PFp
 $(1020_H + p \cdot 4_H)$
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RPR			RES		S26L	S25L	S24L	S23L	S22L	S21L	S20L	S19L	S18L	S17L	S16L
rh			rh		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15L	S14L	S13L	S12L	S11L	S10L	S9L	S8L	S7L	S6L	S5L	S4L	S3L	S2L	S1L	S0L
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Program Memory Unit (PMU)

Field	Bits	Type	Description
SxL (x=0-26)	x	rh	PFlash p Sector x Locked for Write Protection These bits indicate whether PFLASH sector x is write-protected. 0 _B No write protection is configured for sector x. 1 _B Write protection is configured for sector x. In PFlash banks with size < 2 MByte not all bits are functional. In PMU0 PROCONP0 the S6L, S16L and S17L are only effective if these HSM code sectors are not HSM_exclusive.
RPRO	31	rh	Read Protection Configuration This bit indicates whether read protection is configured for PFLASH. This bit is only used in PROCONP0. In other PROCONPs it is reserved. 0 _B No read protection configured 1 _B Read protection and global write protection is configured.
RES	[30:27]	rh	Reserved Deliver the corresponding content of UCB_PFlash.

DFlash Protection Configuration

Only existing in PMUs with DFlash.

When UCB_DFlash is in errored state the initial value is "8000 0000_H", else the initial value after Flash startup reflects the UCB_DFlash content ([Table 10-9](#)).

PROCOND

DFlash Protection Configuration (1030_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RPR	RES	RES29			ESR0CNT										
rh	rh	rh			rh										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP	CAP	CAP	CAP	APR	MODE	OSC	RAMINSEL				RAMIN	NSA	FEC	L	
3EN	2EN	1EN	0EN	EN		CFG						C			
rh	rh	rh	rh	rh	rh	rh	rh				rh	rh	rh	rh	

Program Memory Unit (PMU)

Field	Bits	Type	Description
L	0	rh	DF_EEPROM Locked for Write Protection This bit indicates whether the DFlash sectors EEPROMx are write protected. 0 _B No write protection is configured. 1 _B Write protection is configured.
NSAFECC	1	rh	Non-Safety PFlash ECC This bit indicates whether the SSW selects the “Safety” PFlash ECC algorithm or the “Legacy” ECC. 0 _B Safety ECC is configured. 1 _B Legacy ECC is configured.
RAMIN	[3:2]	rh	RAM Initialization by SSW Control These bits defined whether the RAMs selected by the field RAMINSEL are initialized. 00 _B Init_All , RAM initialization is performed after cold power-on resets and warm power-on resets. 01 _B Init_Warm , RAM initialization is performed after warm power-on resets but not after cold power-on resets (not recommended). 10 _B Init_Cold , RAM initialization is performed after cold power-on resets. 11 _B No_Init , No RAM initialization is performed. This field determines also if a RAM is initialized before MBIST access is granted. In all “Init_*” cases a RAM is initialized before MBIST access is enabled, in the “No_Init” case a RAM is not erased. This is independent of the memory selection with RAMINSEL for initialization during startup (see MTU chapter).

Program Memory Unit (PMU)

Field	Bits	Type	Description
RAMINSEL	[7:4]	rh	RAM Initialization Selection These bits select which memories are initialized when the RAM initialization is configured with RAMIN. Each bit with value '0' selects the corresponding memory for initialization. xxx0 _B RAMs (PSPR, DSPR, PCACHE and DCACHE) in CPU0 are selected for initialization. xx0x _B RAMs (PSPR, DSPR, PCACHE and DCACHE) in CPU1 are selected for initialization. x0xx _B Reserved for CPU2. 0xxx _B LMU RAMs are selected for initialization.
OSCCFG	8	rh	OSC Configuration by SSW This bit indicates whether the oscillator configuration (fields: CAPxEN, APREN, MODE) are installed by the SSW in SCU_OSCCON. 0 _B SSW copy this data into SCU_OSCCON (bits 17 to 23 of PROCOND are ignored). 1 _B SSW configures oscillator with this data.
MODE	[10:9]	rh	OSC Mode When enabled by OSCCFG this field is copied to SCU_OSCCON.MODE.
APREN	11	rh	OSC Amplitude Regulation Enable When enabled by OSCCFG this field is copied to SCU_OSCCON.APREN.
CAPxEN (x=0-3)	12+x	rh	OSC Capacitance x Enable When enabled by OSCCFG these fields are copied to SCU_OSCCON.CAPxEN.
ESR0CNT	[27:16]	rh	ESR0 Prolongation Counter Used to configure the ESR0 delay. Evaluation by SSW.
RES29	[29:28]	rh	Reserved Deliver the corresponding content of UCB_DFlash.
RES30	30	rh	Reserved Deliver the corresponding content of UCB_DFlash.

Program Memory Unit (PMU)

Field	Bits	Type	Description
RPRO	31	rh	Read Protection Configuration This bit indicates whether read protection is configured for DFlash sectors EEPROMx. 0_B No read protection configured 1_B Read protection and write protection is configured.

OTP Protection Configuration

After Flash startup this register represents the or-combination of all PROCONOTP entries of all confirmed configuration sets in UCB_OTP ([Table 10-10](#)).

When any OTP set is in the errored state the initial content is forced to “3FFF FFFF_H”.

PROCONOTPp (p=0-1)
OTP Protection Configuration PFp
 $(1038_H + p * 4_H)$
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TP	BML		RES		S26 ROM	S25 ROM	S24 ROM	S23 ROM	S22 ROM	S21 ROM	S20 ROM	S19 ROM	S18 ROM	S17 ROM	S16 ROM
rh	rh		rh		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15 ROM	S14 ROM	S13 ROM	S12 ROM	S11 ROM	S10 ROM	S9 OM	S8 OM	S7R OM	S6R OM	S5R OM	S4R OM	S3R OM	S2R OM	S1R OM	S0R OM
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxROM (x=0-26)	x	rh	PFlash p Sector x Locked Forever These bits indicate whether PFlash p sector x is an OTP protected sector with read-only functionality. 0_B No OTP protection is configured for sector x. 1_B OTP protection is configured for sector x. In PFlash banks with size < 2 MByte not all bits are functional. In PMU0 PROCONOTP0 the S6ROM, S16ROM and S17ROM are not effective. These HSM code sectors are OTP protected by PROCONHSMCOTP.

Program Memory Unit (PMU)

Field	Bits	Type	Description
TP	31	rh	Tuning Protection This bit indicates whether tuning protection is installed or not. This bit is only evaluated in PMU0 PROCONOTP0. In other PMUs and PROCONOTPs it is reserved. 0 _B Tuning protection is not configured. 1 _B Tuning protection is configured and installed, if correctly confirmed.
RES	[28:27]	rh	Reserved Deliver the corresponding content of UCB_OTP.
BML	[30:29]	rh	Boot Mode Lock Only used in PMU0 PROCONOTP0, in other PROCONOTPs these bits are reserved. Used by the SSW to restrict the boot mode selection. 00 _B Boot flow with standard evaluation of boot headers. 01 _B Restricted boot flow, never evaluating HWCFG pins and without fallback to boot loader. 10 _B Restricted boot flow, never evaluating HWCFG pins and without fallback to boot loader. 11 _B Restricted boot flow, never evaluating HWCFG pins and without fallback to boot loader.

Write-Once Protection Configuration

After Flash startup this register represents the or-combination of all PROCONWOP entries of all confirmed configuration sets in UCB_OTP ([Table 10-10](#)).

When any OTP set is in the errored state the initial content is forced to “FFFF FFFF_H”.

Program Memory Unit (PMU)
PROCONWOPp (p=0-1)
Write-Once Protection Configuration PFp
(1048_H+p*4_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DAT M			RES		S26 WOP	S25 WOP	S24 WOP	S23 WOP	S22 WOP	S21 WOP	S20 WOP	S19 WOP	S18 WOP	S17 WOP	S16 WOP
rh			rh		rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15 WOP	S14 WOP	S13 WOP	S12 WOP	S11 WOP	S10 WOP	S9W OP	S8W OP	S7W OP	S6W OP	S5W OP	S4W OP	S3W OP	S2W OP	S1W OP	S0W OP
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
SxWOP (x=0-26)	x	rh	PFlash p Sector x Configured for Write-Once Protection These bits indicate whether PFlash p sector x is an WOP protected sector. 0 _B No WOP protection is configured for sector x. 1 _B WOP protection is configured for sector x. In PFlash banks with size < 2 MByte not all bits are functional. In PMU0 PROCONWOP0 the S6WOP, S16WOP and S17WOP are not effective. These HSM code sectors are OTP protected by PROCONHSMCOTP.
DATM	31	rh	Disable ATM This bit indicates if the ATM “Application Test Mode” is disabled or not. 0 _B ATM is enabled. 1 _B ATM is disabled. This bit is only used in the PMU0 PROCONWOP0. In other PMUs and PROCONWOPs it is reserved.
RES	[30:27]	rh	Reserved Deliver the corresponding content of UCB_OTP.

Debug Interface Protection Configuration

The debug interface and OCDS enable have a dedicated password protection logic.

When UCB_DBG is in errored state the initial value is “0000 FFFF_H”, else the initial value after Flash startup reflects the UCB_DBG content ([Table 10-12](#)).

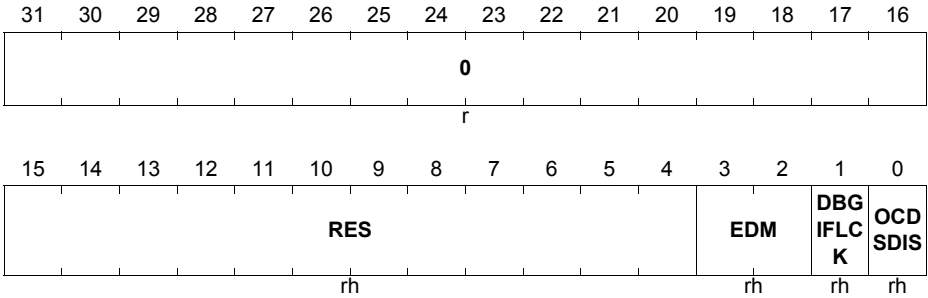
Program Memory Unit (PMU)

PROCONDBG

Debug Interface Protection Configuration

(1058_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
OCDSDIS	0	rh	<p>OCDS Disabled</p> <p>This bit indicates whether the OCDS is configured as locked.</p> <p>0_B No OCDS lock configured in UCB_DBG.</p> <p>1_B OCDS lock configured in UCB_DBG.</p>
DBGIFLCK	1	rh	<p>Debug Interface Locked</p> <p>This bit indicates whether the debug interface is configured as locked.</p> <p>0_B No debug interface lock configured in UCB_DBG.</p> <p>1_B Debug interface lock configured in UCB_DBG.</p>

Program Memory Unit (PMU)

Field	Bits	Type	Description
EDM	[3:2]	rh	Entered Debug Mode This bit indicates whether the debug interface has been opened via the “ Destructive Debug Entry ” (Page 10-37). Consequently the CAN and Flexray operation is made impossible! 00 _B “Debug not entered”, device operation not affected. 01 _B “Debug not entered”, device operation not affected. 10 _B “Debug not entered”, device operation not affected. 11 _B “Debug entered”, CAN and Flexray operation affected.
RES	[15:4]	rh	Reserved Deliver the corresponding content of UCB_DBG.
0	[31:16]	r	Reserved Always read as 0; should be written with 0.

HSM Code Flash OTP Protection Configuration

The HSM code sectors have also their own OTP protection configuration. This register represents after Flash startup the or-combination of all PROCONHSMCOTP entries of all confirmed configuration sets in UCB_HSMCOTP (see [Table 10-13](#)).

When any OTP set is in the errored state the initial content is forced to “0003 007F_H”.

PROCONHSMCOTP
HSM Code Flash OTP Protection Configuration
(1034_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0														S17 ROM	S16 ROM	
														r	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RES	BLK FLA N	DESTDBG	HSMENRE S	HSMENPI NS	S6R OM	HSM 17X	HSM 16X	HSM 6X	HSM DX	SSW WAI T	HSM BOO TEN					
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	

Program Memory Unit (PMU)

Field	Bits	Type	Description
HSMBOOTEN	0	rh	HSM Boot Enable 0 _B HSM Boot is not enabled. 1 _B HSM Boot is enabled. HSMBOOTEN must not be set to 1 in devices without HSM.
SSWWAIT	1	rh	SSW Wait Defines if the SSW waits for the HSM to release the jump of CPU0 to user code. 0 _B The SSW does not wait for HSM. 1 _B SSW wait for acknowledge of HSM.
HSM DX	2	rh	HSM Data Sectors Exclusive This bit indicates whether the HSM data sectors HSMx are configured as "HSM_exclusive". 0 _B HSMx are not HSM_exclusive. 1 _B HSMx are HSM_exclusive.
HSM6X	3	rh	HSM Code Sector 6 Exclusive This bit indicates whether the HSM code sector S6 is configured as "HSM_exclusive". 0 _B S6 is not HSM_exclusive. 1 _B S6 is HSM_exclusive.
HSM16X	4	rh	HSM Code Sector 16 Exclusive This bit indicates whether the HSM code sector S16 is configured as "HSM_exclusive". 0 _B S16 is not HSM_exclusive. 1 _B S16 is HSM_exclusive.
HSM17X	5	rh	HSM Code Sector 17 Exclusive This bit indicates whether the HSM code sector S17 is configured as "HSM_exclusive". 0 _B S17 is not HSM_exclusive. 1 _B S17 is HSM_exclusive.
S6ROM	6	rh	HSM Code Sector 6 Locked Forever This bit indicates whether PFlash sector 6 is an OTP protected sector with read-only functionality. 0 _B No OTP protection is configured. 1 _B OTP protection is configured.

Program Memory Unit (PMU)

Field	Bits	Type	Description
HSMENPINS	[8:7]	rh	Enable HSM Forcing of Pins HSM1/2 This bit indicates whether HSM may force the value of the pins HSM1/2 (i.e. overrule the value driven by the application). 00 _B HSM can't force pins. 01 _B HSM can't force pins. 10 _B HSM can't force pins. 11 _B HSM can force pins.
HSMENRES	[10:9]	rh	Enable HSM Triggering Resets This bit indicates whether HSM may trigger application or system resets. 00 _B HSM can't trigger resets. 01 _B HSM can't trigger resets. 10 _B HSM can't trigger resets. 11 _B HSM can trigger resets.
DESTDBG	[12:11]	rh	Destructive Debug Entry This field configures the destructive debug entry (see Destructive Debug Entry on Page 10-37). 00 _B Debug entry is non-destructive. 01 _B Debug entry is non-destructive. 10 _B Debug entry is non-destructive. 11 _B Debug entry is destructive.
BLKFLAN	13	rh	Block Flash Analysis This bit allows to block the command “Verify Erased Logical Sector Range” on certain HSM related Flash ranges as described on Page 10-21 . 0 _B Functions allowed on all Flash ranges. 1 _B Functions blocked on HSM_exclusive Flash ranges.
RES	[15:14]	rh	Reserved Deliver the corresponding content of UCB_HSMCOTP.
S16ROM	16	rh	HSM Code Sector 16 Locked Forever This bit indicates whether PFlash sector 16 is an OTP protected sector with read-only functionality. 0 _B No OTP protection is configured. 1 _B OTP protection is configured.

Program Memory Unit (PMU)

Field	Bits	Type	Description
S17ROM	17	rh	HSM Code Sector 17 Locked Forever This bit indicates whether PFlash sector 17 is an OTP protected sector with read-only functionality. 0 _B No OTP protection is configured. 1 _B OTP protection is configured.
0	[31:18]	r	Reserved Always read as 0; should be written with 0.

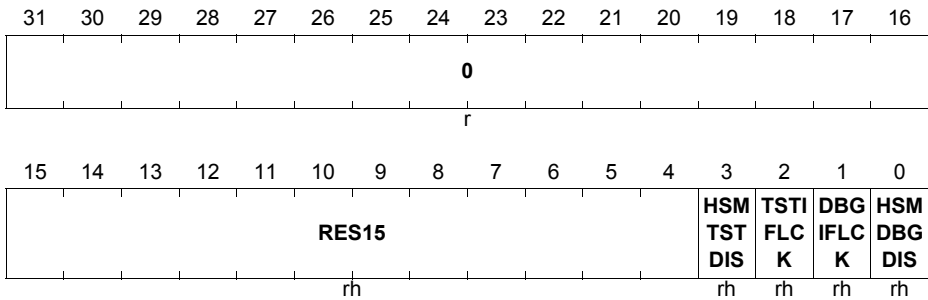
HSM Interface Configuration

Only existing in PMU0. Content only evaluated in devices with activated HSM.

When UCB_HSM is in errored state the initial value is "0000 FFFF_H", else the initial value after Flash startup reflects the UCB_HSM content ([Table 10-14](#)).

PROCONHSM

HSM Interface Configuration (105C_H) **Reset Value: 0000 0000_H**



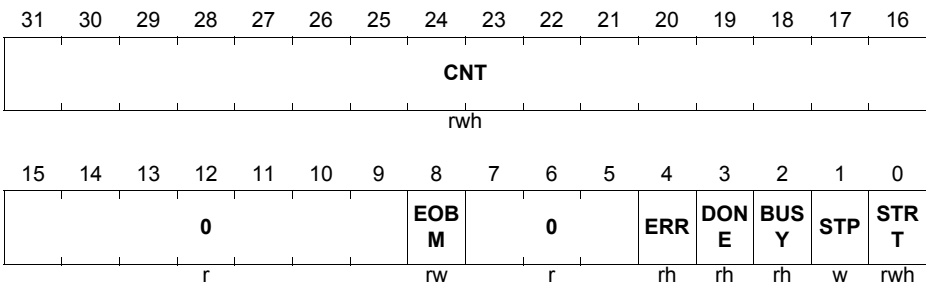
Field	Bits	Type	Description
HSMDBGDIS	0	rh	HSM Debug Disable This bit indicates whether HSM debug is configured as "disabled". 0 _B HSM debug is enabled. 1 _B HSM debug is disabled.
DBGIFLCK	1	rh	Debug Interface Locked This bit indicates whether the chip debug interface is configured as "locked". 0 _B Debug is unlocked. 1 _B Debug is locked.

Program Memory Unit (PMU)

Field	Bits	Type	Description
TAG	[5:0]	rw	Master Tag This read buffer is assigned to the master with SRI tag = TAG.
0	[31:6]	r	Reserved Write 0, read 0.

10.7.2.8 Requested Read Interface

The following registers can be used by any CPU to control the requested read interface and perform data transfers (see [Chapter 10.5.3.4](#)) from the EEPROMx sectors.

Requested Read Control
RRCT
Requested Read Control Register
(1140_H)
Reset Value: 0000 0000_H


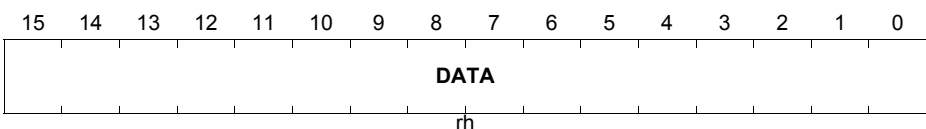
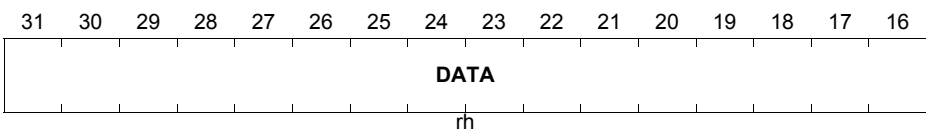
Field	Bits	Type	Description
STRT	0	rwh	Start Request Writing '1' to this bit starts the next read process as configured by RRAD and RRCT.CNT. This bit is cleared by the PMU as soon the reading process starts executing.
STP	1	w	Stop Stops the read process.

Program Memory Unit (PMU)

Field	Bits	Type	Description
BUSY	2	rh	Flash Read Busy 0_B The RRD registers contain the data addressed by RRAD (data ready). 1_B The Flash is busy reading the data addressed by RRAD.
DONE	3	rh	Flash Read Done 0_B The Flash read has not finished. 1_B The Flash read has finished, data is available.
ERR	4	rh	Error Cleared when starting a sequence with STRT, set when an error is detected. 0_B No error occurred. 1_B Error occurred.
EOBM	8	rw	End of Busy Interrupt Mask Interrupt for RRCT.BUSY. 0_B Interrupt disabled. 1_B EOB interrupt is enabled.
CNT	[31:16]	rwh	Count Defines number of remaining data transfers.
0	[15:9], [7:5]	r	Reserved Always read as 0; should be written with 0.

Requested Read Data Registers
RRD_x (x=0-1)
Requested Read Data Register x

$$(1144_H + x \cdot 4_H)$$

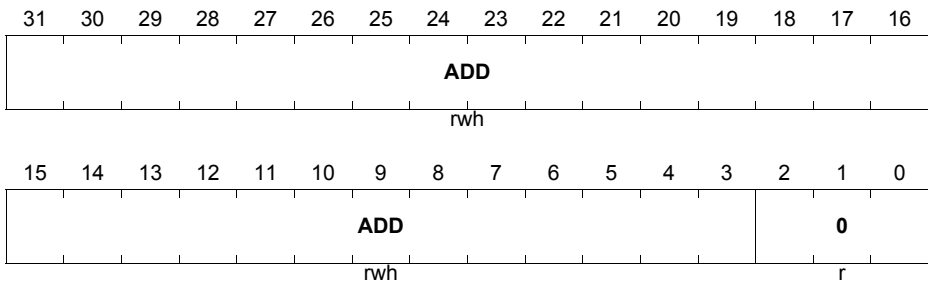
Reset Value: 0000 0000_H


Program Memory Unit (PMU)

Field	Bits	Type	Description
DATA	[31:0]	rh	Read Data Read data.

Requested Read Address Register
RRAD
Requested Read Address Register

 (114C_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
ADD	[31:3]	rwh	Address Start address/current address Can be written by software to define the next read address. Is automatically incremented by the requested read hardware when finishing the last read access. In case both writes happen concurrently the hardware update is performed, the software write is ignored.
0	[2:0]	r	Reserved Always read as 0; should be written with 0.

10.7.2.9 Flash ECC Access
ECC Write Register

The Error Correction Code Write register ECCW contains bits for disabling the ECC encoding separately for PFlash and DFlash.

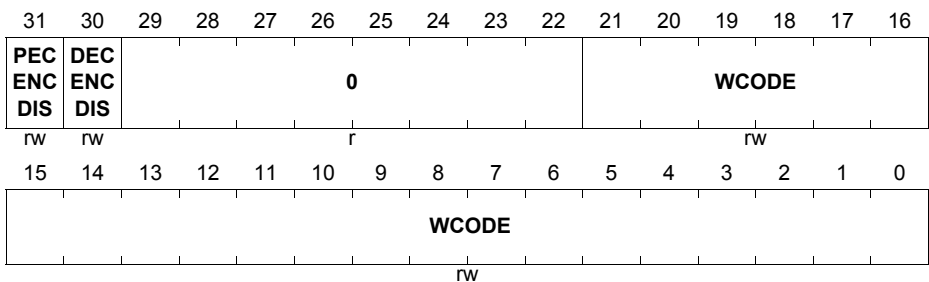
Program Memory Unit (PMU)

When disabling the ECC encoding for PFlash with PECENCDIS = '1' the ECC code for the next 256-bit data block transferred from PMU to the Flash assembly buffer is taken from ECCW.WCODE.

When disabling the ECC encoding for DF_EEPROM with DECENCDIS = '1' the ECC code for the next 64-bit data block transferred from PMU to the Flash assembly buffer is taken from ECCW.WCODE.

Because of internal dependencies only one of PECENCDIS or DECENCDIS may be set to '1'.

When any of PECENCDIS or DECENCDIS is set to '1' the "Write Burst" command sequence is not allowed. Its use would cause unpredictable results.

ECCW
ECC Write Register
(1090_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
WCODE	[21:0]	rw	Error Correction Write Code 22-bit ECC code for the current 64-bit (for DFlash) or 256-bit (for PFlash) write buffer to be written into the assembly buffer instead of a generated ECC.
DECENCDIS	30	rw	DF_EEPROM ECC Encoding Disable 0 _B The ECC code is automatically calculated. 1 _B The ECC code is taken from WCODE.
PECENCDIS	31	rw	PFlash ECC Encoding Disable 0 _B The ECC code is automatically calculated. 1 _B The ECC code is taken from WCODE.
0	[29:22]	r	Reserved Always read as 0.

Program Memory Unit (PMU)
ECC Read Registers

For each Flash read path there is a separate Error Correction Code Read register. These allow disabling the ECC correction. The ECC decoding (i.e. the error detection) is not influenced. After reset ECC correction is enabled. If necessary the trap generation has to be separately disabled by MARP.TRAPDIS and MARD.TRAPDIS. Further on these registers allows to read the ECC code.

The ECC code of the last read access is stored in the RCODE field.

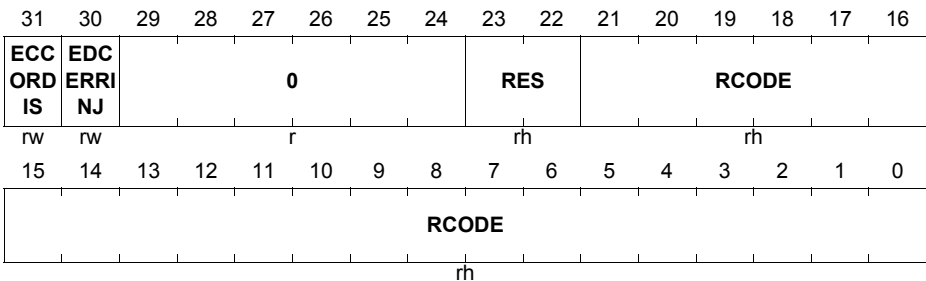
Note: After reading data with disabled ECC correction data buffers throughout the system (e.g. pre-fetch buffers in PMU and CPUs) may contain uncorrected data values. Therefore it is recommended to perform a reset to resume normal operation with ECC correction.

ECCRpp (p=0-1)

ECC Read Register PFp (1094_H+p*4_H) **Reset value: 0000 0000_H**

ECCRD

ECC Read Register DF (10A4_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RCODE	[21:0]	rh	Error Correction Read Code ECC code, read from the Flash read buffer with last data read operation.
RES	[23:22]	rh	Reserved Internal data.

Program Memory Unit (PMU)

Field	Bits	Type	Description
SELPi(i=0-1)	i	rw	PFLASH Bank PFi Selection Bank PFi is selected for reading with MARD.HMARGIN. 0 _B Bank PFi is read with the standard (default) margin independent of HMARGIN. 1 _B Bank PFi is read with the margin selected by HMARGIN.
RES2	2	rw	Reserved Reserved for PFLASH Bank Selection SELP2.
RES3	3	rw	Reserved Reserved for PFLASH Bank Selection SELP3.
TRAPDIS	15	rw	PFLASH Uncorrectable Bit Error Trap Disable 0 _B If an uncorrectable error occurs in PFLASH (setting FSR.PFMBER), a bus error trap is generated ¹⁾ . 1 _B The uncorrectable error trap is disabled.
0	[14:4], [31:16]	r	Reserved Always read as 0; should be written with 0.

1) After Boot ROM exit, uncorrectable bit error traps are enabled (TRAPDIS = 0_B).

Margin Check Control DFlash and Suspend
MARD
Margin Control Register DFlash (10AC_H) **Reset Value: 0000 8000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRAPDIS						0					SPNDER	SPND	0	SELDO	HMARGIN
rw						r					rwh	rwh	r	rw	rw

Program Memory Unit (PMU)

Field	Bits	Type	Description
HMARGIN	0	rw	Hard Margin Selection Banks selected by MARD.SELi = 1 or MARP.SELi = 1 are read with: 0 _B Tight0 , Tight margin for 0 (low) level. Suboptimal 0-bits are read as 1s. 1 _B Tight1 , Tight margin for 1 (high) level. Suboptimal 1-bits are read as 0s. The concrete margin values are restored from the configuration sector and are determined by Infineon.
SELDO	1	rw	DFLASH Bank Selection Bank DF0 is selected for reading with HMARGIN. 0 _B Bank DF0 is read with the standard (default) margin independent of HMARGIN. 1 _B Bank DF0 is read with the margin selected by HMARGIN.
SPND	3	rwh	Suspend 0 _B No suspend requested or pending. 1 _B Suspension of the ongoing program/erase process is requested or pending.
SPNDERR	4	rwh	Suspend Error 0 _B No suspend error. 1 _B Last suspend request via MARD.SPND failed (see Chapter 10.5.4.4). Can be cleared by writing '1'.
TRAPDIS	15	rw	DFLASH Uncorrectable Bit Error Trap Disable 0 _B If an uncorrectable error occurs in DFLASH (setting FSR.DFMBER), a bus error trap is generated ¹⁾ . 1 _B The uncorrectable error trap is disabled.
0	2, [14:5], [31:16]	r	Reserved Always read as 0; should be written with 0.

1) After Boot ROM exit, uncorrectable bit error traps are enabled (TRAPDIS = 0_B).

Program Memory Unit (PMU)

10.7.2.11 Corrected Bits Address Buffer (CBAB)

When data is read from PFlash and the ECC decoder detects correctable bit errors those addresses are stored in the “corrected bits address buffer”. Each address is only entered once. Depending on its configuration 1-bit and 2-bit errors or only 2-bit errors are entered.

When CBAB becomes full the SMU is informed which can trigger an interrupt.

When the CBAB is full no further addresses are added. With CBABCFGp.CLR or CBABTOPp.CLR entries can be emptied.

There exists one CBAB per PFlash “p” SRI port.

Each CBAB instance implements ten entries.

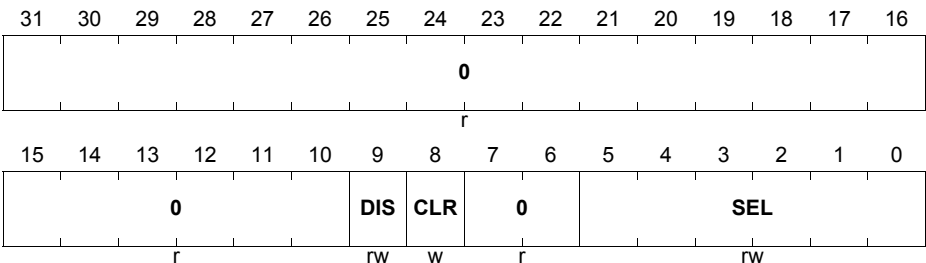
The CBAB is called in safety documentation “PFLASH Monitor”.

CBAB Configuration

Reset: power-on reset.

CBABCFGp (p=0-1)

CBAB Configuration Port p (10B4_H+p*C_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
SEL	[5:0]	rw	<p>Select Bit-Errors</p> <p>This bit controls which type of errors are entered into the CBAB.</p> <p>000000_BNo errors enter the CBAB.</p> <p>000001_BCorrected single-bit errors enter the CBAB.</p> <p>000010_BCorrected double-bit errors enter the CBAB.</p> <p>000011_BCorrected single-bit and double-bit errors enter the CBAB.</p> <p>others: Reserved values.</p>

Program Memory Unit (PMU)

Field	Bits	Type	Description
CLR	8	w	Clear Clears the complete CBABp.
DIS	9	rw	Disable This bit allows to disable this CBAB. This reduces the power consumption. The content of the CBAB stays unchanged. <i>Note: Due to race conditions switching this bit concurrently to Flash read accesses can cause an erroneous address to become registered twice.</i>
0	[7:6], [31:10]	r	Reserved Always read as 0; should be written with 0.

CBAB Status

Reset: power-on reset.

CBABSTATp (p=0-1)
CBAB Status Port p

 (10B8_H+p*C_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						VLD	VLD	VLD	VLD	VLD	VLD	VLD	VLD	VLD	VLD
r						9	8	7	6	5	4	3	2	1	0
r						rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
VLDx (x=0-9)	x	rh	Filling Level Each VLDx indicates a valid entry. The complete set of VLD flags operates as thermometer code. Entry 0 is the top entry which is read in CBABTOP. The entry with the highest index number was entered last.
0	[31:10]	r	Reserved Always read as 0; should be written with 0.

Program Memory Unit (PMU)

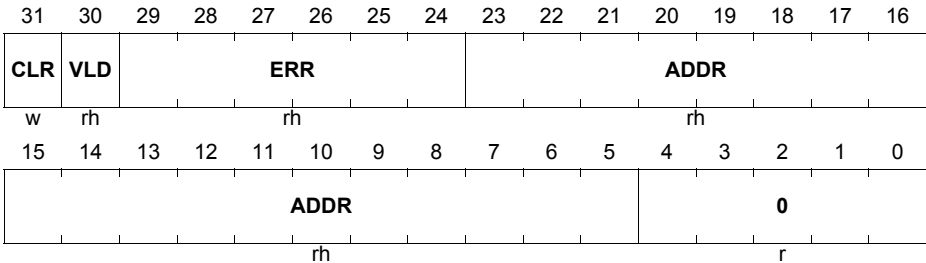
CBAB FIFO Top Entry

This register displays the information of the top-most entry of the CBAB FIFO.

Reset: power-on reset (for the complete FIFO content).

CBABTOPp (p=0-1)

CBAB FIFO TOP Entry Port p ($10BC_H + p * C_H$) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
CLR	31	w	Clear Write-only trigger bit. Reading returns 0. 1 _B Clears valid flag "VLD" of this entry and removes it from the FIFO. 0 _B No operation.
VLD	30	rh	Valid Entry is valid.
ERR	[29:24]	rh	Error Type This field contains the error type found at ADDR. The bits correspond to CBABCFG.SEL. 000000 _B No error (reserved). 000001 _B Single-bit error. 000010 _B Double-bit error. others: Reserved.
ADDR	[23:5]	rh	Address Captured address (bits 23:5 of the SRI bus address) with detected error.
0	[4:0]	r	Reserved Always read as 0; should be written with 0.

Program Memory Unit (PMU)

10.7.2.12 Uncorrectable Bits Address Buffer (UBAB)

When data is read from PFlash and the ECC decoder detects uncorrectable bit errors those addresses are stored in the “uncorrected bits address buffer”. Each address is only entered once. Depending on its configuration selected failures can be entered.

When the UBAB is full no further addresses are added. With UBABCFGp.CLR or UBABTOPp.CLR entries can be emptied.

There exists one UBAB per PFlash “p” SRI port.

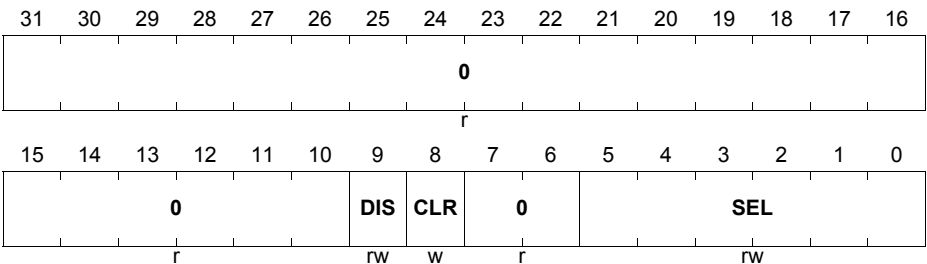
Each UBAB instance implements one entry.

UBAB Configuration

Reset: power-on reset.

UBABCFGp (p=0-1)

UBAB Configuration Port p (10E4_H+p*C_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
SEL	[5:0]	rw	Select Bit-Errors This bit controls which type of errors are entered into the UBAB. 000000 _b No errors enter the UBAB. 000100 _b Detected uncorrectable bit-errors (3-bit or more, address errors, all-0, all-1 enter the UBAB. others: Reserved values.
CLR	8	w	Clear Clears the complete UBABp.

Program Memory Unit (PMU)

Field	Bits	Type	Description
DIS	9	rw	Disable This bit allows to disable this UBAB. This reduces the power consumption. The content of the UBAB stays unchanged.
0	[31:10] , [7:6]	r	Reserved Always read as 0; should be written with 0.

UBAB Status

Reset: power-on reset.

UBABSTATp (p=0-1)
UBAB Status Port p (10E8_H+p*C_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															VLD
r															0
r															rh

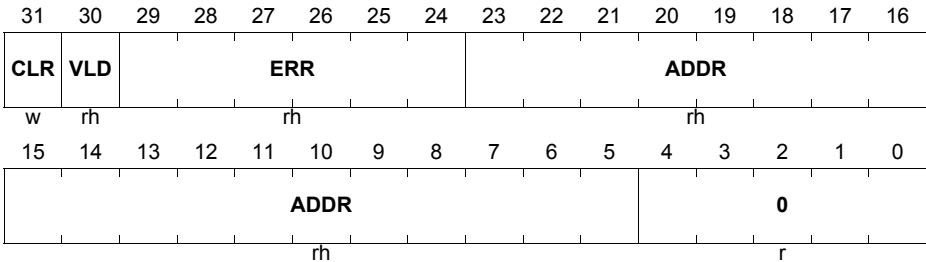
Field	Bits	Type	Description
VLD0	0	rh	Filling Level Each VLDi indicates a valid entry. The complete set of VLD flags operates as thermometer code. Entry 0 is the top entry which is read in UBABTOP. The entry with the highest index number was entered last.
0	[31:1]	r	Reserved Always read as 0; should be written with 0.

UBAB FIFO Top Entry

This register displays the information of the top-most entry of the UBAB FIFO.

Program Memory Unit (PMU)

Reset: power-on reset (for the complete FIFO content).

UBABTOPp (p=0-1)
UBAB FIFO TOP Entry Port p (10EC_H+p*C_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
CLR	31	w	Clear Write-only trigger bit. Reading returns 0. 1 _B Clears valid flag "VLD" of this entry and removes it from the FIFO. 0 _B No operation.
VLD	30	rh	Valid Entry is valid.
ERR	[29:24]	rh	Error Type This field contains the error type found at ADDR. The bits correspond to UBABCFG.SEL. 000000 _B No error (reserved). XXX100 _B Uncorrectable error (3-bit or more, addressing error, all-0, all-1). others: Reserved.
ADDR	[23:5]	rh	Address Captured address (bits 23:5 of the SRI bus address) with detected error.
0	[4:0]	r	Reserved Always read as 0; should be written with 0.

10.7.2.13 Direct Flash Communication

 These registers enable direct communication of software (e.g. a Flash driver running on TriCore) with the FSI specifically with its μ Code.

Program Memory Unit (PMU)

Because of design particularities byte wide write accesses are not supported and cause unpredictable results. Only half-word (16-bit) wide writes shall be performed.

Read and write accesses during ongoing Flash operations (FSR shows BUSY flags) is forbidden. The PMU returns a bus error.

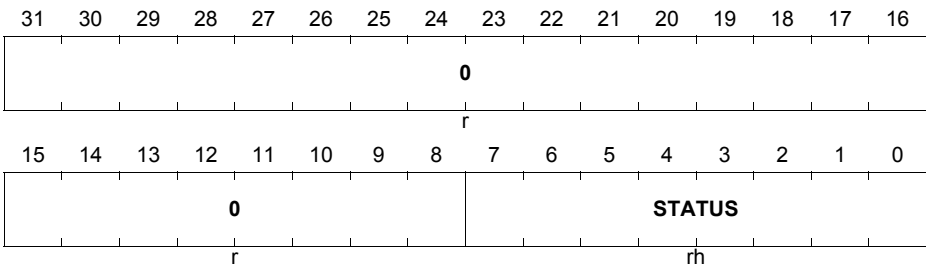
These registers shall be read or written in exceptional cases only when demanded by IFX for extended functions.

The behavior of these registers in specific Flash modes is special:

- During all FSI operations including Flash startup: the STATUS fields can deliver unpredictable data.
- When the Flash is in sleep mode: write and read accesses fail with a bus error.

Flash Communication Register 0
COMM0

FSI Communication Register 0 (0000_H) **Reset Value: 0000 0000_H**



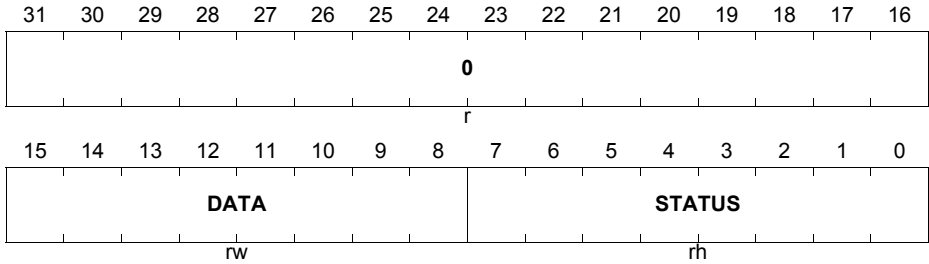
Field	Bits	Type	Description
STATUS	[7:0]	rh	Status This field can be only written by the FSI.
0	[31:8]	r	Reserved Always read as 0.

Program Memory Unit (PMU)

Flash Communication Register

COMMx (x=1-2)

FSI Communication Register x (0000_H+4*x) Reset Value: 0000 0000_H



Field	Bits	Type	Description
STATUS	[7:0]	rh	Status This field can be only written by the FSI.
DATA	[15:8]	rwh	Data This field can be only written by software not by the FSI.
0	[31:16]	r	Reserved Always read as 0.

10.8 Application Hints

The following application hints should support the user in using the PMU and Flash optimally.

10.8.1 Changes with Respect to Audo Families Audo-NG/F/S/Max

The following changes of the PMU with respect to Audo-NG/F/S/Max products are highlighted. With “Audo-Max” are changes noted with respect to SRI based Audo-Max products. The tag “All” indicates changes with respect to all Audo product generations:

- All: Flash cell concept is changed enabling many significant improvements:
 - Performance of programming significantly enhanced.
 - Burst programming feature with even higher programming throughput.
 - No over-erased state anymore.
 - Sector structure of PFlash enhanced with more small Flash sectors.
 - EEPROM emulation in DFlash improved by smaller pages.
 - Enhanced retention and endurance figures.
 - All logical sectors of PFlash (see [Table 10-2](#)) have the same endurance. “Second class” logical sectors with only 100 cycle endurance are not existing anymore.
- All: Changed PMU and FSI architecture (enabled by the increased programming performance). Now one FSI and PMU can serve several PFlash banks in addition to the DFlash banks.
- All: Due to this one PMU and FSI has to support several independent read paths with independent bus interfaces to the assigned PFlash banks.
- All: Changed command sequences. In Audo the SA and PA arguments were transmitted in the address phase, thus these command cycles had to be written to the target Flash range. Now SA and PA are transmitted in the data phase. Only writes into the DFlash address range are interpreted as command sequences.
- Audo-NG/F/S: Enhanced Flash read buffer concept with master specific read buffers.
- Audo-NG/F/S: Changed bus interface from LMB to SRI.
- All: Changed supply concept supporting EVR. Dedicated programming modes for 3.3V only supply or 5V supply.
- Audo-NG/F/S: The wait-state settings in FCON don't relate by default to the bus clock as in Audo-NG/F/S (i.e. the LMB clock) but to the FSI and FSI2 clocks. These clocks have their own divider from the PLL clock which need to be configured in SCU_CCUCON0.FSIDIV and SCU_CCUCON0.FSI2DIV. They have to be configured to have a relation $f_{SRI}/f_{FSI(2)}$ of $n/1$.
- All: Explicit suspend to read feature replacing automatic “program while erase” of the Audo generation.
- Audo-NG/F/S: The OVRAM and the EMEM were moved to a different module “LMU”.
- All: The register field FCON.STALL configures the behavior when reading busy Flash banks. Either “ready” is suppressed or a bus error is generated.
- Audo-NG/F/S: TP: changed addresses RTPWT and RPSSW.

Program Memory Unit (PMU)

- Audo-NG/F/S: Programming and reading data with disabled ECC generation and correction allows customers to program data with invalid ECC.
- Audo-NG/F/S: Error flags in FSR can be selectively cleared to enable easier flag handling for concurrent operations.
- Audo-Max: Dropped extended status register XSFR.
- All: Protection fields assembled in new register FPRO.
- All: Changed protection concept:
 - Only one protection “user” in PFlash.
 - Separate protection with password for DFlash.
 - Password width increased from 64 bits to 256 bits.
 - More flexible OTP configuration that allows adding OTP.
 - New write-once protection enabling “automatic” OTP protection.
 - All UCBs realized as part of DFlash instead of PFlash.
 - Confirmation concept for UCB blocks changed. Now an erased confirmation code activates complete protection and locks the UCB.
 - New password protected debug lock.
- All: New erase counters.
- All: Changed base addresses of all Flash modules.
- All: Changed base address of BootROM.
- All: TEC-QED ECC algorithm in DFlash.
- All: DEC-TED ECC algorithm in PFlash with specific safety features for 99% white noise coverage, detection of all-0/all-1 inducing faults and addressing errors.
 - As in Audo-Max the ECC algorithm can be switched between safety and non-safety mode.
 - Attention: switching the algorithm with FCON.NSAFECC may only be done while no Flash accesses are performed. This feature may conflict with Audo-F Flash driver software.
- All: New “Erase Verify” functionality.
- All: New multiple sector erase command sequence “Erase Logical Sector Range”.
- All: Extended safety protection of registers with ACCEN and “Safety ENDINIT”.
- All: FIFOs that store uniquely addresses in PFlash read with corrected and uncorrectable bit errors.

10.8.2 Performing Flash Operations

This section offers advice for programming command sequences.

General Advice

- Please remember to disable the Safety ENDINIT protection before executing program and erase commands for the PFlash (see [Chapter 10.5.5.2](#)).
- Code that performs PFlash programming or erasing should not be executed from the same PFlash.

Program Memory Unit (PMU)

- The command cycles shall address the non-cached address range of the Flash (otherwise the data may stay in the cache or could be received by the PMU in an incorrect order).
- The non-cached Flash range is not regarded as peripheral address space by the TriCore. This means that a read (notably reading the FSR for checking the flags) placed behind the last command cycle write in program order might be executed before this last write.
The FSR flags xFPAGE, PROG, ERASE can be used to ensure that polling for a cleared BUSY starts only after the command was accepted by the PMU. Additionally it is recommended to place a “DSYNC” instruction between a command sequence and read accesses to depending data including affected registers.
- The caches (data cache in DMI “DCache” and program cache in PMI “PCACHE”) as well as the data line buffer in the DMI “DLB” are not automatically invalidated nor updated after changing Flash content by erasing or programming. It is therefore recommended to either invalidate them actively or read the Flash content via the non-cached address range. Otherwise old data might be delivered from these buffers. The DMI cache and line buffer can be invalidated by writing OVCCON.DCINVAL to ‘1’. The PCACHE can be invalidated by writing PCON1.PCINV to ‘1’. All buffers are invalidated by a reset.
- The PMU and Flash work with the SRI, FSI and FSI2 clocks. When changing the divider values of these clocks in SCU_CCUCON0 the following rules apply:
 - The only allowed PMU/Flash “operation” when switching is reading from Flash memory. Programming or erasing the Flash is forbidden.
 - It must be ensured that before, during and after the divider change the configured number of Flash wait-cycles is sufficient for the selected clock frequency. Remember that the wait-cycles are counted with f_{FSI2} for PFlash and f_{FSI} for DFlash.
 - After System Resets and Power-On Resets the wait cycle values are configured to a value only sufficient for the clock frequencies used during startup (i.e. 100 MHz). So basically always changes to the clock configuration must be preceded by changes to the wait cycles.

Sequence for Programming

The following sequence is the most defensive one for programming a page. It is however acceptable to skip some checks when the programmed data is later verified:

- “Clear Status” to clear flags.
- “Enter Page Mode”.
- DSYNC.
- Wait until FSR.xFPAGE = ‘1’ or fail if FSR.SQER = ‘1’ or FSR.PROER = ‘1’.
- Repeat “Load Page” until the page is filled.
- “Write Page”.
- DSYNC.

Program Memory Unit (PMU)

- Wait until FSR.PROG = '1' or fail if FSR.SQER = '1' or FSR.PROER = '1'.
- Wait until FSR.xBUSY = '0' or enable the interrupt.
 - While FSR.xBUSY is '1' the flags FSR.OPER can be checked for '1' as abort criterion to protect against hardware failures causing BUSY to stay '1'.
- Check for FSR.PVER flag.
- Fail if FSR.OPER = '1'.
- Recommended: check programmed content, evaluate FSR.xMBER and possibly count correctable errors.
- Clear error flags either with "Clear Status" or by directly writing to FSR.

Sequence for Erasing

The following sequence is the most defensive one for erasing a range of sectors. It is however acceptable to skip some checks when the programmed data is verified after programming (or the flag FSR.PVER is checked as described above):

- "Clear Status" to clear flags.
- "Erase Logical Sector Range".
- DSYNC.
- Wait until FSR.ERASE = '1' or fail if FSR.SQER = '1' or FSR.PROER = '1'.
- Wait until FSR.xBUSY = '0' or enable the interrupt.
 - While FSR.xBUSY is '1' the flags FSR.OPER can be checked for '1' as abort criterion to protect against hardware failures causing BUSY to stay '1'.
- Check for FSR.EVER/PVER flags.
- Fail if FSR.OPER = '1'.
- Clear error flags either with "Clear Status" or by directly writing to FSR.

An analog sequence can be used for "Verify Erased Logical Sector Range".

10.8.3 EEPROM Emulation With DFlash

The term "EEPROM emulation" designates an algorithm with the following features:

- It increases the effective endurance by spreading the EEPROM write accesses over a larger range of Flash memory.
- It ensures that all Flash cells incur a similar number of cycles independent of the update frequency of the EEPROM data ("wear levelling").
- It manages the allocation of EEPROM data to Flash ranges so that stale data can be erased.

A popular emulation algorithm is the "two sector" emulation, which splits the complete DFlash in two equally sized ranges:

- The EEPROM writes are performed in one range, the "active" one. The other range stays erased.
- The data is stored so that several writes of the same EEPROM address can be performed in one DFlash range. The position of the latest data version and the

Program Memory Unit (PMU)

original EEPROM address can be determined from the address in Flash and additional administrative data (e.g. all data is pushed to a stack and the pushed data contains the EEPROM address).

- At a certain range fill level a range switch is performed. This moves all active data to the erased range. After that the active range is erased and the other range becomes the active one.

With the 6 erasable DFlash ranges of the TC26x B-Step other more efficient algorithms are possible. However the algorithm must ensure that all sectors are used equally, e.g. it is not allowed to keep static data in certain sectors and often updated data in other sectors. The sectors with static data would incur too many disturbs.

One specific requirement for an EEPROM emulation algorithm is the resistance against aborts during its operation. In any case the algorithm must be able after device startup to recover without data loss, determine the active data and resume operation.

In many applications even resistance against aborted Flash processes (program, erase) is needed. See details in [Chapter 10.8.5.2](#).

10.8.3.1 Robust EEPROM Emulation

A key requirement for an EEPROM emulation algorithm is the reliability of the stored data. The DFlash with its TEC-QED ECC algorithm protects perfectly against bit or bit-line oriented failures.

However, ECC mechanisms do not protect against word-line oriented failures. Word-line failure modes may result in a complete word-line which can no longer be programmed or erased, and the data within the failed word-line may not be able to be read correctly. Word-line failures are most likely to occur during the high voltage conditions present during programming or erase operations. Therefore, a robust EEPROM emulation algorithm must tolerate word-line failures, and must protect against data loss in case the word-line fails during an erase or programming operation.

The following steps shall be followed to achieve the necessary robustness:

- Before programming a page save the content of all other pages on the same word-line that contain active data to SRAM.
- Program the new page and compare the content of this page and of the saved pages with their reference data. This can be done with normal read margins. Ignore correctable bit-errors.
- If the data comparison fails or the programming returned with PVER error program this page and the saved content of the other pages to a different word-line.
- This procedure can be repeated if the data comparison fails again. The number of repetitions should be limited (e.g. to 3) in case the programming fails because of out-of-spec operating conditions.
- Word-line oriented fails can also have the effect that the affected word-lines can not be erased anymore, yet the contents of the word-line data could still be read without

Program Memory Unit (PMU)

any error indication. Sequence counters for the data blocks or other means must be used to identify old data blocks in word-lines with this type of failure mode.

For the TC26x B-Step this robust EEPROM algorithm is required for the usage of the DFlash.

Due to the specificity of each application the appropriate usage and implementation of these measures must be chosen according to the context of the application.

10.8.4 Handling Errors

The earlier sections described shortly the functionality of “error indicating” bits in the flash status register **FSR**. This section elaborates on this with more in-depth explanation of the error conditions and recommendations how these should be handled by customer software.

10.8.4.1 Handling Errors During Operation

This first part handles error conditions occurring during operation (i.e. after issuing command sequences) and the second part ([Section 10.8.4.2](#)) error conditions detected during startup.

SQER “Sequence Error”

Fault conditions:

- Improper command cycle address or data, i.e. incorrect command sequence.
- New “Enter Page” in Page Mode.
- “Load Page” and not in Page Mode.
- “Load Page” results in buffer overflow.
- “Load Page” with mixed 32/64-bit transfers.
- First “Load Page” addresses 2. word.
- “Write Page” with buffer underflow.
- “Write Page” and not in Page Mode.
- “Write Page” to unavailable Flash range.
- Command sequence with address not pointing to a legal start address (e.g. page, UCB or sector).
- “Write Page Once” targeting DFlash.
- Byte transfer to password or data.
- “Erase Logical Sector Range” or “Verify Erased Logical Sector Range” with range leaving a physical sector.
- “Resume Prog/Erase” with arguments not matching the suspended command.
- “Resume Prog/Erase” when there is no suspended operation.
- Any programming or erase command when there is a suspended programming command.

Program Memory Unit (PMU)

- Any erase command when there is a suspended erase command, including “Verify Erased Logical Sector Range”.
- Programming to the target range of a suspended erase command.

New state:

Read mode is entered with following exceptions:

- “Enter Page” in Page Mode re-enters Page Mode.
- “Write Page” with buffer underflow is executed.
- After “Load Page” causing a buffer overflow the Page Mode is not left, a following “Write Page” is executed.

Proposed handling by software:

Usually this bit is only set due to a bug in the software. Therefore in development code the responsible error tracer should be notified. In production code this error will not occur. It is however possible to clear this flag with “Clear Status” or “Reset to Read” and simply issue the corrected command sequence again.

OPER “Operation Error”

Fault conditions:

ECC double-bit error detected in Flash microcode SRAM during an operation or the microcode has detected another significant failure. This can be a transient event due to alpha-particles or illegal operating conditions or it is a permanent error due to a hardware defect. This situation will practically not occur.

Attention: these bits can also be set during startup (see [Chapter 10.8.4.2](#)).

New state:

The Flash operation is aborted, the BUSY flag is cleared and read mode is entered.

Proposed handling by software:

The last operation can be determined from the PROG and ERASE flags. In case of an erase operation the affected physical sector must be assumed to be in an invalid state, in case of a program operation only the affected page. Other sectors can still be read. New Flash commands (Write*, Erase*, Verify*, Resume*) must not be issued before the next system or power-on reset. “Clear Status” can be used to clear the OPER flag, however this doesn’t resolve the corrupted state of the FSI.

Attention: New FSI operations in this state can cause corruption of Flash content, which might even prevent the device from booting.

The next system or power-on reset performs a new Flash startup with initialization of the microcode SRAM. A power-on reset clears additionally the OPER flag. The application must determine from the context which operation failed and react accordingly. Mostly erasing the addressed sector and re-programming its data is most appropriate. If a “Program Page” command was affected and the sector can not be erased (e.g. in Flash

Program Memory Unit (PMU)

EEPROM emulation) the word-line could be invalidated if needed by marking it with all-one data and the data could be programmed to another empty word-line.

Only in case of a defective microcode SRAM the next program or erase operation will incur again this error.

PROER “Protection Error”

Fault conditions:

- Password failure.
- Erase/Write to protected sector.
- Erase/Write of protected UCB.
- “Verify Erased Logical Sector Range” to blocked addresses.
- Program or erase targeting PFlash with active Safety ENDINIT protection.

New state:

Read mode is entered. The protection violating command is not executed.

Proposed handling by software:

Usually this bit is only set during runtime due to a bug in the software. In case of a password failure a reset must be performed in the other cases the flag can be cleared with “Clear Status” or “Reset to Read”. After that the corrected sequence can be executed.

EVER “Erase Verify Error”

Fault conditions:

This flag is set by the erase commands when they don’t achieve an optimum result.

New state:

No state change. Just the bit is set.

Proposed handling by software:

This bit should be cleared with “Clear Status” or “Reset to Read”.

The operating conditions should be checked. The following advice assumes correct operating conditions and a correctly configured device.

In the PFlash an EVER can be regarded as device failure.

In the DFlash an EVER is a warning which doesn’t require immediate action. The robust EEPROM emulation will detect during programming the corresponding word-lines and “jump” over them.

Note: Even when this flag is ignored it is recommended to clear it. Otherwise all following operations — including “sleep” — could trigger an interrupt even when they are successful (see [Chapter 10.5.7](#), interrupt because of verify error).

Program Memory Unit (PMU)**PVER “Program Verify Error”**Fault conditions:

This flag is set by the program commands when they don't achieve an optimum result.

New state:

No state change. Just the bit is set.

Proposed handling by software:

This bit should be cleared with “Clear Status” or “Reset to Read”.

The operating conditions should be checked. The following advice assumes correct operating conditions and a correctly configured device.

In the PFlash a PVER can be regarded as device failure. However due to the high correction capability of the ECC customers may choose to check the amount of corrected and uncorrectable errors and decide on this basis if the device can stay in operation.

In the DFlash a PVER is a signal for the robust EEPROM emulation that programming on the current word-line failed. The algorithm has to “jump” over this word-line and programs its data to the next word-line.

Note: Even when this flag is ignored it is recommended to clear it. Otherwise all following operations — including “sleep” — could trigger an interrupt even when they are successful (see [Chapter 10.5.7](#), interrupt because of verify error).

PFSBER, PFDBER, DFSBER, DFDBER, DFTBER “Single/Double/Triple-Bit Error”Fault conditions:

When reading data or fetching code from PFlash or DFlash the ECC evaluation detected an error which was corrected.

This flag is a warning indication and not an error. A certain amount of correctable errors must be expected because of known physical effects.

New state:

No state change. Just the bit is set.

Proposed handling by software:

This flag can be used to analyze the state of the Flash memory. During normal operation it should be ignored. High counts of correctable errors can indicate that the Flash is operated outside of the defined operating conditions (e.g. temperature, voltage, endured p/e cycles, configured wait-states).

When programming the PFlash (end-of-line programming or SW updates) customers can count the number of correctable errors using the CBAB or rely on the PVER indication.

Program Memory Unit (PMU)

In case of EEPROM emulation using DFlash the verification of programmed data should be done with the normal read level and correctable errors should be ignored. Further advice can be found in [Chapter 10.8.3](#).

10.8.4.2 Handling Errors During Startup

The FSR flags are not only used to inform about the success of Flash command sequences but they are also used to inform (1) the startup software and (2) the user software about special situations incurred during startup. In order to react on this information these flags must be evaluated after reset before performing any flag clearing sequence as “Clear Status” or “Reset to Read”.

The following two levels of situations are separated:

- Fatal level: the user software is not started. A WDT reset is performed.
- Warning level: the user software is started but a warning is issued.

Fatal Level (WDT Reset)

These error conditions are evaluated by the startup software which decides that the Flash is not operable and thus waits for a WDT reset. The application sees only a longer startup time followed by a WDT reset.

The reason for a failed Flash startup can be a hardware error or damaged configuration data.

Warning Level

These conditions inform the user software about an internally corrected or past error condition.

Leftover OPER:

FSR bits set: OPER.

The OPER flag are only cleared by the command sequence “Clear Status” or with a power-on reset. After any other reset a OPER flag can still be set when the user software is started.

Correctable error in UCB or configuration sector:

FSR bits set: PFSBER/PFDBER, DFSBER, DFDBER, DFTBER or ORIER.

An correctable ECC error was detected during installation of the protection or of the configuration sector content.

10.8.5 Resets During Flash Operation

A reset or power failure during an ongoing Flash operation (i.e. program or erase) must be considered as violation of stable operating conditions. However the Flash was designed to prevent damage to non-addressed Flash ranges when the reset is applied

Program Memory Unit (PMU)

as defined in the data sheet. The addressed Flash range is left in an undefined state. Additional means are implemented that help to prevent aborting programming processes in the DFlash.

10.8.5.1 General Advice

When an erase operation is aborted the previously programmed bits ('1') in the addressed Flash range can be in any state between '0' and '1'. When reading this range all-0 can be returned, the old data, or something in between. The result can be instable. Due to the ECC correction there may even appear '1' bits at positions which contained '0' bits before erase start.

When a page programming operation is aborted the page can still appear as erased (but contain slightly programmed bits), it can appear as being correctly programmed (but the data has a lowered retention) or the page contains garbage data. It is also possible that the read data is instable so that depending on the operating conditions different data is read.

For the detection of an aborted Flash process the flags FSR.PROG and FSR.ERASE could be used as indicator but only when the reset was an application reset. When Flash processes are aborted by power-on resets this is not indicated by any flags. It is not possible to detect an aborted operation simply by reading the Flash range (please note that the ECC is not a reliable means to detect an aborted Flash operation). Even the margin reads don't offer a reliable indication.

When erasing or programming the PFlash usually an external instance can notice the reset and restart the operation by erasing the Flash range and programming it again.

10.8.5.2 Advice for EEPROM Emulation

However for the case of EEPROM emulation in the DFlash this external instance is not existing. A common solution is detecting an abort by performing two operations in sequence and determine after reset from the correctness of the second the completeness of the first operation.

E.g. after erasing a DFlash sector a page is programmed. After reset the existence of correct data in this page proves that the erase process was performed completely.

The detection of aborted programming processes can be handled similarly. After programming a block of data an additional page is programmed as marker. When after reset the block of data is readable and the marker is existent it is ensured that the block of data was programmed without interruption.

In very specific cases it is allowed to repair data left from an aborted programming operation: if the algorithm can detect that an abort occurred and the algorithm knows which data must be present in the page it is possible to simply redo the programming by programming the same data again.

Program Memory Unit (PMU)

In the AURIX family the probability of aborting a programming process can be minimized by the following means:

- In case of a reset with stable power supply (“warm resets”) the Flash gets automatically a request to enter safe state before the reset is applied. Due to their short duration single “Write Page” operations are finished correctly by this process. “Write Burst” operations however are interrupted after the current page programming has finished.
- The voltage monitoring (see SCU chapter) can be used to get an under voltage warning early enough that an ongoing programming process can be finished and no new one is started.
- By selecting an internally generated programming voltage (see [Chapter 10.5.4.5](#)) the needed voltage at V_{EXT} is reduced giving more headroom for an early warning by the voltage monitors.

10.8.6 ECC

For special applications the ECC features need to be considered.

Using the PFlash Safety ECC

The PFlash Safety ECC is a vital part of the safe fetch path needed for safety applications. The difference between Legacy ECC and Safety ECC is explained in [Chapter 10.5.6.2](#).

The calculation of the Safety ECC is done over 256 data bits, the address bits 22:5 and the configuration area selection signal. As side effect erased PFlash ranges or PFlash ranges programmed with the Legacy ECC will be read with ECC errors.

Thus the installation of an application using the Safety ECC should be done as follows:

- By default (delivery state) the device starts with the Safety ECC enabled.
- The Flash loader can execute from RAM. Erased PFlash ranges shall not be read.
- A sector should be programmed completely to ensure that it can be read without ECC errors.
- When changing the UCB_DFlash it is important to keep the ECC configuration flag `PROCOND.NSAFECC = 0`.

The installation of an application using the Legacy ECC should be done as follows:

- By default (delivery state) the device starts with the Safety ECC enabled.
- The Flash loader can execute from RAM.
- By writing `FCON.NSAFECC` to 1 the ECC algorithm is switched to Legacy ECC.
- Afterwards the PFlash programming is performed with the Legacy ECC.
- Erased PFlash ranges can be read without error.
- In UCB_DFlash the ECC configuration flag `PROCOND.NSAFECC` shall be programmed to 1 to ensure that the device starts after next reset with the Legacy ECC. It must be ensured that is kept when changing later the UCB_DFlash.

Program Memory Unit (PMU)

Attention: Changing `FCON.NSAFECC` while accessing the Flash (reading, programming, erasing) is forbidden.

Attention: Reading PFlash ranges with the incorrect ECC (e.g. Flash was programmed with Legacy ECC but reading with setting of `NSAFECC=0`) must not be done although it might seem to work when the uncorrectable error traps are disabled.

On certain address/data combinations mis-corrections will happen and single and double-bit errors in PFlash will not be corrected because for the wrong ECC algorithm nearly all addresses will have uncorrectable errors resulting in the delivery of the raw data.

Creating Incorrect ECC

In safety applications error detection features are usually checked at each startup. In order to check the detection logic for ECC errors addresses in PFlash need to be programmed with erroneous data.

The automatic ECC generation can be disabled with `ECCW.PECENCDIS`. With this feature any combination of data and ECC bits can be programmed.

The ECC code causing the required error type (e.g. triple-bit error) can be determined by first programming the data with automatic ECC generation on. When reading this data the corresponding ECC code can be found in `ECCRPp.RCODE`. After erasing this range of Flash a modified data/ECC combination can be programmed with automatic ECC generation switched off (`ECCW.PECENCDIS = 1`).

It is also possible to create ECC incorrect patterns by programming twice with enabled automatic ECC generation.

Ready-to-use programming routines for the creation of ECC errors are part of the Infineon safety software "SafeTlib".

10.8.7 Startup Tests of ECC Logic

As described in [Chapter 10.6](#) the ECC logic of the PFlash is protected by special hardware means, the "ECC Monitor" and the "EDC Comparator". Depending on the targeted ASIL level these units, their error signals and generally the ECC error signaling to the SMU has to be tested during each startup after reset.

The Infineon safety software "SafeTlib" contains routines to perform these tests.

10.8.7.1 Testing ECC Alarms and Error Flags

In order to test if each type of ECC error is correctly reported to the SMU and PMU status registers a pattern with this error needs to be stored in PFlash and read during startup after reset.

Such patterns can be created as described in [Chapter 10.8.6](#).

Test Pattern

As the patterns stored in PFlash might change their error signature over life-time (e.g. a single-bit error becomes a double-bit error due to a single bit toggling) each pattern needs to be stored redundantly (we propose 4 times).

As the ECC logic of each PFlash bank has separate paths the patterns have to be stored in each PFlash bank and tested in each PFlash bank.

Patterns for the following types of errors should be stored and read:

- No error (valid data). This data could be shared with the one for [Chapter 10.8.7.2](#).
- 1-bit error.
- 2-bit error.
- 3-bit error to trigger an uncorrectable error.
- Address error (can be created by generating an ECC for an address that is different by 1 or 2 address bits). This test pattern is optional. It is not needed for sufficient test coverage.
- All-0 in data and ECC. This test pattern is optional. It is not needed for sufficient test coverage.
- All-1 in data and ECC. This test pattern is optional. It is not needed for sufficient test coverage.

In sum these 7 pattern (each covering one 32-byte page) are stored 4 times per PFlash bank.

For best possible robustness 2 of the 4 pattern sets shall be inverse to the other 2.

Test Flow

For each error type its 4 patterns are read. The corresponding error flag and SMU alarm should occur at least once.

10.8.7.2 Testing the “ECC Monitor”

As the ECC Monitor is always in use its logic is usually tested sufficiently during startup of the application, simply by performing read accesses to the PFlash banks.

However in order to achieve a defined error coverage a dedicated test procedure recommended.

Test Pattern

In each PFlash bank a set of 32 valid pages (i.e. containing no intentional ECC error) is stored. These patterns are chosen by IFX to ensure >90% stuck-at coverage of the ECC Monitor logic.

Test Flow

The 32 pages of each PFlash bank are read by any master e.g. the CPU.

Program Memory Unit (PMU)

The ECC monitor logic is working correctly if during this test the alarm “PFLASH ECC monitor error” at the SMU is not activated.

10.8.7.3 Testing the SMU Alarm of the “ECC Monitor”

In order to test that the “PFLASH ECC monitor error” alarm to the SMU can be activated the ECC monitor has to receive a pattern with incorrect ECC for which the ECC correction logic states that a correction was performed.

Test Pattern

In each PFlash bank a pattern with a correctable error has to be read, e.g. the ones used in [Chapter 10.8.7.1](#).

Test Flow

The ECC correction has to be disabled for the tested PFlash bank “p” with ECCRPp.ECCORDIS. After that read 4 patterns from this bank with a correctable error. This shall trigger at least once the PFLASH ECC Error alarm in the SMU. Clear the alarm, clear ECCRPp.ECCORDIS for this bank and repeat the test for the other PFlash banks.

10.8.7.4 Testing the “EDC Comparator”

The EDC comparator compares the output of two error checking units. In case of a difference the SMU alarm “PFLASH EDC comparator error” is activated. This error can be enforced by setting for a PFlash bank “p” by setting ECCRPp.EDCERRINJ.

Test Pattern

The test patterns of [Chapter 10.8.7.1](#) can be used.

Test Flow

The test patterns are read with ECCRPp.EDCERRINJ = 0. This is already part of the test in [Chapter 10.8.7.1](#). The “PFLASH EDC comparator error” alarm must not be activated.

The test patterns are read with ECCRPp.EDCERRINJ = 1. This shall trigger the “PFLASH EDC comparator error” alarm in the SMU.

Repeat this test for all PFlash banks.

10.8.7.5 General Advice for Startup Tests

The previously described tests need to be executed under controlled conditions:

Program Memory Unit (PMU)

- Only a single master with blocked interrupts should execute these tests. This is mandatory for tests with disabled ECC correction (**Chapter 10.8.7.3**) because other masters might receive uncorrected Flash data during that time.
- For the same reason the Flash test code should execute from RAM. Especially it must not execute from a PFlash bank with disabled ECC correction (**Chapter 10.8.7.3**).
- It has to be taken into account that internal pre-fetching of the PMU can trigger the “PFLASH ECC monitor error” and the “PFLASH EDC comparator error” before the corresponding Flash read access is issued by the CPU. For the ECC alarms and error flags of **Chapter 10.8.7.1** this issue is not existing as the PMU activates these errors only when the respective data is read from the bus.
- The alarms activation in the SMU has a latency of several clock cycles.
- The software executing these tests has to cope with all consequent actions like activated error flags, entries in error buffers, bus-errors, SMU alarms. Before normal execution continues these mechanisms have to be cleared.
- The existence of Flash addresses with correctable and uncorrectable errors has to be respected by the application software. Interpretation as real error has to be prevented.

11 Local Memory Unit (LMU)

The Local Memory Unit is an SRI peripheral providing access to volatile memory resources. It's primary purpose is to provide access to the separate block of emulation and debug memory (EMEM) provided in the Emulation Devices.

The LMU also provides OnLine Data Acquisition (OLDA) region support. This provides an address space where writes complete without error but no memory is addressed. This allows production code to be written which writes data to memory which is only present in the Emulation Device. When running in the Emulation Device, the code maps EMEM to the OLDA region address space using the memory overlay feature and the writes are stored in the EMEM. when running in the Production Device, the LMU terminates writes to the OLDA address space without error, even though no memory exists at the target address, but the write data is discarded.

11.1 Feature List

An overview of the features implemented in the LMU follows:

- Interface to the EMEM of the Emulation Device.
- OLDA region support.

11.2 Local Memory (LMU SRAM)

This version of the LMU has no local memory.

11.3 Emulation Memory (EMEM)

In the Emulation Device, an area of Emulation Memory (EMEM) is provided, which can be used for either calibration via overlay of non-volatile memory or overlay of the OLDA address space (see [Chapter 1.8](#) below). The address range allocated for the memory is $9F000000_H$ to $9F3FFFFFF_H$ which allows the maximum EMEM size fitted in any of the derivative products to be addressed.

As well as the cached addresses (segment 9_H), noncached address (segment B_H) accesses can be used for EMEM accesses via the LMU.

The Emulation Memory interface controls the CPU-accesses to the Emulation Memory in the Emulation Device. All widths of write accesses are supported (byte, halfword, word, double-word).

CPU-controlled Load-Modify-Store accesses (with LDMST instruction) are supported as separate read and write instructions not as an atomic operation.

In the production device, the EMEM interface is always disabled. A CPU read access from the Emulation Memory region causes a DSE trap by returning an SRI bus error. If the Emulation Memory region read access is initiated by a SPB master (e.g. PCP), additionally a SPB error interrupt can be generated.

Local Memory Unit (LMU)

By default, write accesses to the Emulation Memory by any master will cause an SRI bus error trap in the production device.

In the Emulation Device, a SRI bus error is returned by the LMU if a read access can't be handled by the EMEM, for example, when the CPU accesses a trace memory tile in EMEM. In this case, the EMEM access is aborted by the LMU.

Write accesses which cannot be handled by the EMEM will fail silently as the write access is completed on the SRI bus before being passed to the EMEM. Therefore any error condition encountered by the EMEM will occur after the SRI access has completed. A status bit **LMU_MEMCON.EWERR** will be set if an error occurs on an EMEM write. This bit can be cleared by writing 0_B.

Wait states can be manually added to EMEM memory accesses using the **LMU_MEMCON.WSTATES** field. This can be programmed with a counter preload value. A counter is loaded with this value every time the LMU starts a read access to the EMEM. Data returned by the EMEM will be delayed until the counter reaches a count of zero. The counter is decremented by one every SRI clock cycle.

11.3.1 EMEM Memory Read Buffers

Accesses to the EMEM address space can be stored to read buffers. One read buffer will be instantiated for each CPU in the system. Each of these buffers will store 256 bits of data. Buffering of read data is controlled by the **LMU_BUFCONx (x=0-2).EREN** bit and will be enabled when **EREN** is 1_B. Speculative prefetch is available only if buffering of read data is enabled and is controlled by the **LMU_BUFCONx (x=0-2).EPEN** bit. Setting **EPEN** to 1_B will allow prefetch if **EREN** is also 1_B. Two Unique Master Tag IDs can be manually associated with each CPU using the **LMU_BUFCONx (x=0-2).TAG1** and **LMU_BUFCONx (x=0-2).TAG2**. Each tag field can be individually enabled using the **LMU_BUFCONx (x=0-2).EN1** and **LMU_BUFCONx (x=0-2).EN2** fields respectively.¹⁾

If read buffering is enabled, then all reads to the EMEM not using the BTR4 opcode will be mapped to BTR4 equivalent read accesses to the EMEM and the data returned stored in the related buffer while the requested data is returned to the SRI interface. Any further reads to the address range of the data stored in the buffer will be returned directly from the buffer.

If prefetch is enabled, then two conditions will trigger a prefetch:

- any SDTW read to the EMEM will cause a speculative read of the next BTR4 address range to be triggered if the address of the read corresponds to the highest 32 bit address of a BTR4²⁾ (even if the data for the read is already stored in the buffer).

1) Multiple buffers should not be configured to respond to the same Master Tag ID. LMU behaviour in this case is undefined.

2) A BTR4 access requires 256 bits of data or 8, 32 bit words. These words always have A_{SRI}[4:2] = 000_B, 001_B, 010_B, 011_B, 100_B, 101_B, 110_B, 111_B. An 32 bit access with A_{SRI}[4:2] = 111_B will trigger a prefetch

Local Memory Unit (LMU)

- Any BTR4 read to the cached segment address of the EMEM (9_H) will trigger a prefetch (even if the data for the read is already stored in the buffer).

The prefetched data will be stored in the related buffer and any reads to the address range of the prefetch will be returned directly from the buffer. A prefetch will not occur if another read to EMEM is already required. Write accesses will be scheduled after the prefetch has completed.

Prefetch takes priority over read buffering.

Any write access to EMEM which collides with the address range stored in a buffer will invalidate that buffer. Read accesses to the Emulation Device register space will not be buffered.

11.3.2 Access to Emulation Device Register Space

In addition to accessing EMEM, the EMEM interface in the LMU is also used to access configuration registers in the Emulation Device. To allow this accesses to the address range $F9000000_H$ to $F90FFFFF_H$ are directed to the ED register address space via the EMEM interface.

The interface to the ED register space only supports 32 bit accesses. Any other size of access will be errored.

11.4 Error Detection and Signalling

The LMU will detect several different classes of error which cannot be signalled on the SRI during the associated transaction. These will cause a flag to be set in the **LMU_MEMCON** register and a trigger will be sent to either the SMU or the Interrupt system for processing. In general ECC errors will be processed by the SMU and other errors will be passed to the Interrupt Router. The following list details the detected error conditions:

11.4.1 EMEM Read Error

An error signalled by the EMEM on the second data phase of a 256 bit access cannot be signalled on the SRI as this would violate the protocol or, in the case of a 32 bit read, occur after the SRI access has completed. In this event, the **LMU_MEMCON.ERERR** bit will set and a trigger will be sent to the Interrupt Router.

11.4.2 EMEM Write Error

All errors signalled by the EMEM during a write access will, by definition in the SRI protocol, occur after the access has been accepted from the SRI. These errors will set the **LMU_MEMCON.EWERR** bit and send a trigger to the Interrupt router

11.4.3 SRI write access data phase error

If an ECC error occurs on the data phase of an SRI write access then the LMU_MEMCON.DATAERR bit will be set and an error will be signalled to the SMU

11.4.4 SRI access address phase error

If an ECC error occurs on the address phase of an SRI access then the LMU_MEMCON.ADDERR bit will be set and an error will be signalled to the SMU. The SRI access will terminate with an error.

11.5 Online Data Acquisition (OLDA) and its Overlay

Calibration is additionally supported by an OLDA memory range of up to 32 Kbyte, which is a virtual memory and physically only available if it is redirected (by the overlay feature of the processor) to internal or external physical memory or to the EMEM in the Emulation Device.

If OLDA support is enabled in the LMU, direct write accesses (without redirection) to the OLDA range are not really executed, and they do not generate a bus error trap¹⁾. If OLDA support is not enabled, write accesses will generate a bus error trap. OLDA support is enabled by setting LMU_MEMCON.OLDAEN to 1_B.

Note that switching on or off the OLDA function takes a finite amount of time after the register write completes. It is therefore possible for an immediately following access pipelined into the LMU SRI interface to be acknowledged with either the "on" or "off" behaviour.

Read accesses to the OLDA range generate a bus error trap, if not redirected to a physically available overlay block. Successful accesses to the OLDA memory range will only take place when the accesses are redirected to real, physical memory.

The base address of the virtual OLDA memory range is A/8FE7 0000_H, the end address is A/8FE7 7FFF_H. Accesses to the OLDA range are also supported in cached address space.

Note: In OTARx registers, any target address can be selected for redirection, thus also addresses in the OLDA range. However, the handling of direct accesses to the OLDA range is completely controlled in the LMU.

11.6 Clock Control

The LMU contains a clock control register, **LMU_CLC**, which allows the LMU to be put into a power saving mode.

1) Write accesses to a cached memory address will trigger a read to fill the cache line before the data is written to the cache. This read will trigger a bus error.

Local Memory Unit (LMU)

If LMU_CLC.DISR is set then the LMU will be disabled and all accesses will be errored unless they are addressed to a register.

11.7 LMU Register Protection

The LMU implements the standard memory protection scheme for peripheral registers using the **LMU_ACCEN0** and **LMU_ACCEN1** registers. This allows the LMU control registers to be protected from corruption by untrusted masters. Masters are identified using the SRI tag of the access and, if the appropriate bit is not set in the access enable registers, write accesses will be disconnected with error acknowledge. See the On Chip Bus chapter for the product's master Tag ID to master peripheral mapping. This protection scheme does not apply to the **LMU_ACCEN0** and **LMU_ACCEN1** themselves.

LMU_ACCEN0 and **LMU_ACCEN1** are protected by Safe Endinit while all other registers are Endinit protected. The Endinit and Safe Endinit system status is defined by different Watchdog Units in the System Control Unit (SCU).

11.8 LMU Registers

The LMU registers are mapped into a 256 byte address space which allows for 64 registers. Accesses to unused register space will cause an SRI bus error.

Table 11-1 Registers Address Space

Module	Base Address	End Address	Note
LMU	F870 0800 _H	F870 08FF _H	All registers are endinit or safe endinit protected and are accessible in Supervisor mode only

Table 11-2 Registers Overview

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Page Number
			Read	Write		
LMU_CLC	LMU Clock Control	0 _H	SV,32	SV,E,P,32	3	4-33
RESERVED	reserved address space	4 _H	BE	BE	n.a.	
LMU_MODID	LMU Module ID	8 _H	SV,32	R	3	4-34
RESERVED	reserved address space	C _H	BE	BE	n.a.	
LMU_ACCEN0	LMU Access Enable 0	10 _H	SV,32	SV,SE,32	3	4-35
LMU_ACCEN1	LMU Access Enable 1	14 _H	SV,32	SV,SE,32	3	4-36
RESERVED	reserved address space	18 _H to 1C _H	BE	BE	n.a.	
LMU_MEMCON	LMU Memory Control	20 _H	SV,32	SV,E,P,32	3	4-37
RESERVED	reserved address space	24 _H to 2C _H	BE	BE	n.a.	
LMU_BUFCON0	LMU Buffer Control 0	30 _H	SV,32	SV,E,P,32	3	4-41
LMU_BUFCON1	LMU Buffer Control 1	34 _H	SV,32	SV,E,P,32	3	4-41

Local Memory Unit (LMU)
Table 11-2 Registers Overview (cont'd)

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Page Number
			Read	Write		
LMU_BUFCON 2	LMU Buffer Control	38 _H	SV,32	SV,E,P, 32	3	4-41
RESERVED	reserved address space	3C _H to FF _H	BE	BE	n.a.	

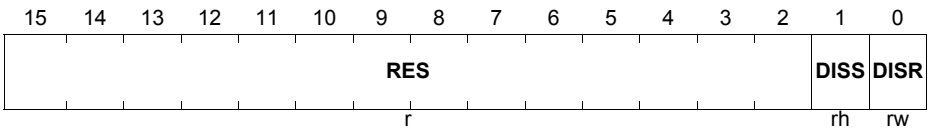
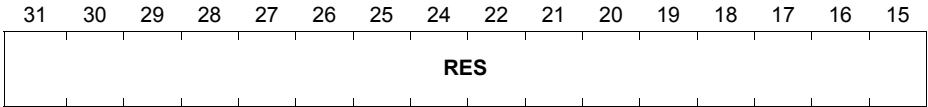
1) The absolute register address is calculated as follows:
Module Base Address ([Table 1-5](#)) + Offset Address (shown in this column)

Local Memory Unit (LMU)

LMU Clock Control Register

LMU_CLC

LMU Clock Control Register (000_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
DISR	0	rw	LMU Disable Request Bit This bit is used for enable/disable control of the LMU. 0 _B LMU disable is not requested 1 _B LMU disable is requested
DISS	1	rh	LMU Disable Status Bit Current state of LMU. 0 _B LMU is enabled (default after reset) 1 _B LMU is disabled
RES	31:2	r	Reserved

Local Memory Unit (LMU)

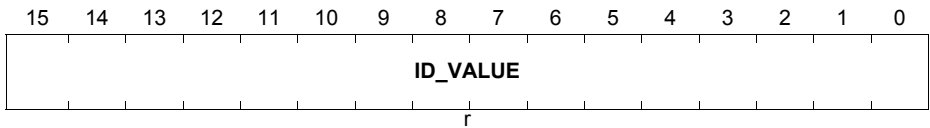
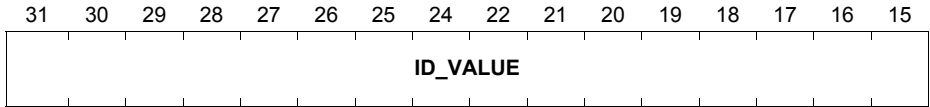
LMU Module ID Register

LMU_MODID

LMU Module ID Register

(008_H)

Reset Value: 0088 C002_H



Field	Bits	Type	Description
ID_VALUE	31:0	r	Module Identification Value

Local Memory Unit (LMU)

LMU Access Enable Register 0

The Access Enable Register 0 controls write access for transactions to registers with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The LMU is prepared for an 6 bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ... ,EN31 -> TAG ID 011111_B.

LMU_ACCEN0

LMU Access Enable Register 0

(10_H)

Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Field	Bits	Type	Description
ENn (n = 0-31)	n	r/w	<p>Access Enable for Master TAG ID n</p> <p>This bit enables write access to the LMU register addresses for transactions with the Master TAG ID n</p> <p>0_B Write access will not be executed. Read accesses will be executed.</p> <p>1_B Write and read accesses will be executed</p>

Local Memory Unit (LMU)

LMU Access Enable Register 1

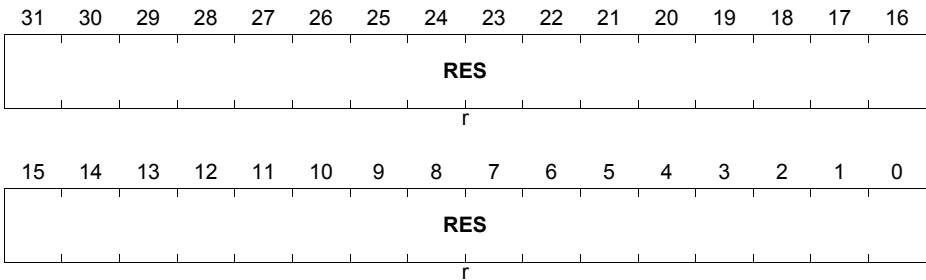
The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). These tags are not used in this system and so no programmable bits are provided.

LMU_ACCEN1

LMU Access Enable Register 1

(14_H)

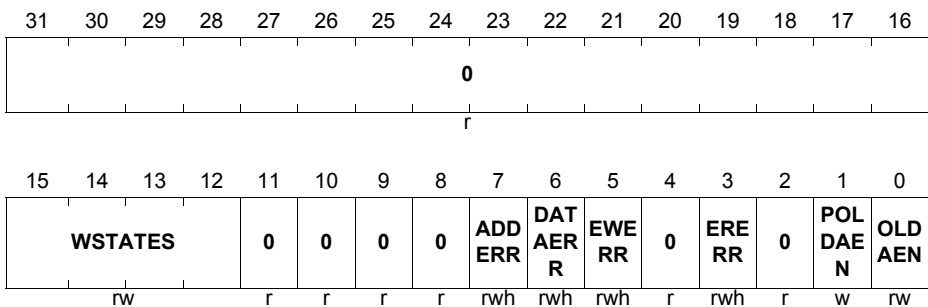
Reset Value: 0000 0000_H



Field	Bits	Type	Description
RES	31:0	r	Reserved Read as 0; should be written with 0.

Local Memory Unit (LMU)
LMU Memory Control Register

Provides Control of the memory integrity error checking, error signalling to the SMU and error injection for ECC logic test. Also control of the OLDA function. The register is cleared by an application reset.

LMU_MEMCON
LMU Memory Control Register (020_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
OLDAEN	0	rw	Online Data Acquisition Enabled This bit is used to control trap generation for write accesses to the OLDA address range. 0 _B Trap generation on write access to OLDA memory range is enabled 1 _B No trap generated on write access to OLDA memory range.
POLDAEN	1	w	Protection Bit for OLDAEN This bit always returns 0 _B when read. 0 _B Bit Protection: Bit OLDAEN remains unchanged after LMU_MEMCON write. 1 _B OLDAEN can be changed by current write to LMU_MEMCON
0	2	r	Reserved Read as 0 _B must be written with 0 _B

Local Memory Unit (LMU)

Field	Bits	Type	Description
ERERR	3	rwh	EMEM Read Error Flag set by hardware when the LMU logic detects an error on an EMEM read transaction which cannot be reported on the SRI bus. This bit is cleared by writing 0 _B but cannot be set by software. 0 _B No error has occurred 1 _B An unreportable error has been observed on an EMEM read transaction.
0	4	r	Reserved Read as 0 _B must be written with 0 _B
EWERR	5	rwh	EMEM Write Error Flag set by hardware when the LMU logic detects an error on an EMEM write transaction. This bit is cleared by writing 0 _B but cannot be set by software. 0 _B No error has occurred 1 _B An error has been observed on an EMEM write transaction.
DATAERR	6	rwh	SRI Data Phase ECC Error Flag set by hardware when the SRI interface detects an ECC error in the data phase of an incoming write transaction. This bit is cleared by writing 0 _B but cannot be set by software. 0 _B No error has occurred 1 _B An ECC error has been observed on an SRI transaction addressed to the LMU
ADDERR	7	rwh	SRI Address Phase ECC Error Flag set by hardware when the SRI interface detects an ECC error in the address phase of an incoming transaction. This bit is cleared by writing 0 _B but cannot be set by software. 0 _B No error has occurred 1 _B An ECC error has been observed on an SRI transaction addressed to the LMU
0	9:8	r	Reserved Read as 0 _H , must be written as 0 _H
0	10	r	Reserved Read as 0 _H , must be written as 0 _H
0	11	r	Reserved Read as 0 _H , must be written as 0 _H

Local Memory Unit (LMU)

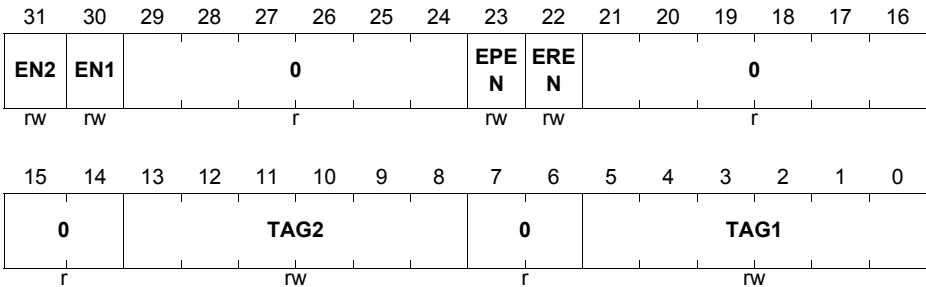
Field	Bits	Type	Description
WSTATES	15:12	rw	EMEM Wait States Wait State Counter for EMEM accesses. See “Emulation Memory (EMEM)” on Page 4-6 for operational details.
0	31:16	r	Reserved Read as 0 _H , must be written as 0 _H

Local Memory Unit (LMU)
LMU Buffer Control Register

Enables control of a read buffer. The register is cleared by an application reset.

LMU_BUFCONx (x=0-2)

LMU Buffer Control Register (030_H+x*04_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
TAG1	5:0	rw	Master Tag ID 1 This field should store the Master Tag ID of accesses which are required to trigger an update of the read and prefetch buffers.
0	7:6	r	Reserved Read as 0 _B , should be written with 0 _B .
TAG2	13:8	rw	Master Tag ID 2 This field should store the Master Tag ID of accesses which are required to trigger an update of the read and prefetch buffers.
0	21:14	r	Reserved Read as 0 _B , should be written with 0 _B .
EREN	22	rw	EMEM Read Buffer Enable Enable bit for the read buffer function on EMEM reads 0 _B Do not buffer reads from the EMEM 1 _B Read buffer is enabled.
EPEN	23	rw	EMEM Prefetch Enable Enable bit for the prefetch function on EMEM reads 0 _B Do not prefetch from the EMEM 1 _B Prefetch is enabled.

Local Memory Unit (LMU)

Field	Bits	Type	Description
0	29:24	r	Reserved Read as 0 _B , should be written with 0 _B .
EN1	30	rw	TAG1 Field Enable Enable bit for the TAG1 field 0 _B TAG1 does not contain a valid Unique Master Tag ID value and should be ignored. 1 _B TAG1 does contain a valid Unique Master Tag ID value and should be used when triggering an update for the read buffer.
EN2	31	rw	TAG2 Field Enable Enable bit for the TAG1 field 0 _B TAG2 does not contain a valid Unique Master Tag ID value and should be ignored. 1 _B TAG2 does contain a valid Unique Master Tag ID value and should be used when triggering an update for the read buffer.

12 Data Access Overlay (OVC)

The data overlay provides the capability to redirect selected data accesses to the Overlay memory. Data accesses made by the TriCore to Program Flash, Online Data Acquisition space, or EBU space can be redirected. Overlay memory may be located in the Local Memory (if present), in the Emulation Memory (Emulation Device only), or in the DPSR/PSPR memory. Overlay functionality makes it possible, for example, to modify the application's test and calibration parameters (which are typically stored in Flash memory) during run time of a program. Note that only read and write data accesses are redirected. The access redirection is executed without performance penalty.

Attention: As the address translation is implemented in the DMI it is only effective for data accesses by the TriCore. Instruction fetches by the TriCore or accesses by any other master (including the debug interface) are not effected!

Summary of Features and Functions

- Redirecting data accesses addressed to Program Flash, OLDA or External EBU Space.
- Support redirection to Overlay memory located in:
 - Local Memory (LMU) (if present)
 - Emulation Memory (Emulation Device only)
 - DPSR or PSPR memory
- Support of up to 4 MB overlay memory address range;
- Up to 32 overlay ranges ("blocks") available in each TriCore instance;
- Overlay block size from 32 byte to 128 Kbyte;
- Up to 4 MB of redirected space (32 ranges x 128 Kbyte);
- Overlay memory location and block size selected individually for every overlay block;
- Multiple overlay blocks can be enabled or disabled with one register write access;
- Programmable flush (invalidate) control for data cache in DMI.
- Overlay start/stop synchronised against data load.
- Individual Overlay system per processor core.

12.1 Data Access Redirection

The principle of redirecting data access from the original target memory (“Target Address”) to overlay memory (“Redirected Address”) is shown below.

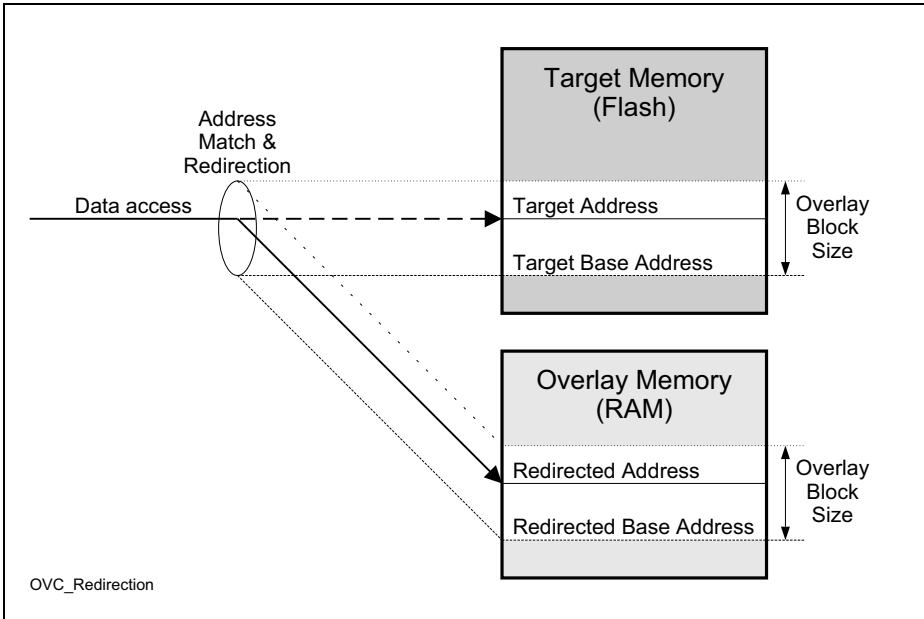


Figure 12-1 Data Access Redirection

Data access overlay is defined using overlay ranges (“overlay blocks”). Each overlay block defines one continuous range of address space for which the accesses are redirected. Each overlay block is configured with the following parameters:

- Overlay Block Target Base Address - the start address for the range of the target addresses to be redirected;
- Overlay Block Size - the size of the range of the addresses to be redirected;
- Overlay Block Redirection Base Address - the start address for redirection.

In TC26x B-Step up to 32 overlay ranges can be used in each TriCore instance.

Each overlay block has 3 associated registers for independent configuration of these parameters. The overlay parameters are configured as follows:

- Target Base Address is configured with OTARx register (“[OTARx \(x=0-31\)](#)” on [Page 12-13](#)),
- Overlay Block Size is configured with OMASKx register (“[OMASKx \(x=0-31\)](#)” on [Page 12-14](#)),

Data Access Overlay (OVC)

- Redirection Base Address is configured with RABRx register (“RABRx (x=0-31)” on Page 12-11).

The size of the overlay memory blocks can be $2^n \times 32$ bytes, with $n = 0$ to 12. This gives the block size range from 32 bytes to 128 Kbytes. The start address of the block can only be an integer multiple of the programmed block size (natural alignment boundary). If OTAR register value, or RABR register value is not aligned with the block size, the least significant bit values are ignored and treated as zero.

The Redirection Base Address is determined by two fields in RABRx register:

- RABRx.OMEM selecting the Overlay Memory, and
- RABRx.OBASE selecting the base address within this memory.

Each overlay block can be activated or deactivated with its RABRx.OVEN bit. Overlay blocks can be activated and deactivated independently, by directly accessing RABRx register, or in groups, where multiple configured blocks are activated or deactivated concurrently. For information on concurrent block activation see Chapter 12.4.1.

The address redirection process is shown in the following figure.

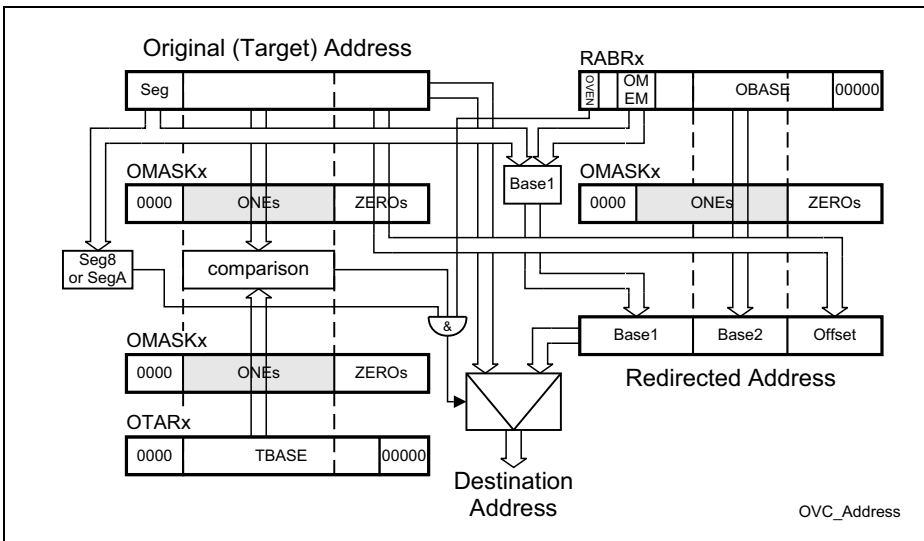


Figure 12-2 Address Redirection Process

Any data access to segment 8_H or segment A_H is checked against all the activated overlay blocks. For each activated overlay block, address bits 27..5 are compared with the target base address (OTARx), and this bit-wise comparison is qualified by the content of OMASKx register. Address bits participate in the comparison if the

Data Access Overlay (OVC)

corresponding OMASKx bits are set to one. The access is redirected, if all the address bits selected by OMASKx equal to the corresponding bits in OTARx register.

The address for redirection is constructed as follows:

- Address bits 31..22 are set according to the overlay memory selection (RABRx.OMEM) and the cache-ability of the original address.
- For address bits 21..5:
 - If the corresponding OMASKx bit is set, the address bit value is taken from RABRx.OBASE field;
 - If the corresponding OMASKx bit is cleared, the address bit value is taken from the original address.
- Address bits 4..0 are always taken directly from the original address.

If there is no redirection, the original address is used to perform the access.

Target address ranges for activated overlay blocks should not overlap or an exception may occur, see [Chapter 12.6](#).

12.2 Target Memories

Data access to any memory within the segment 8_H or segment A_H may be redirected to overlay memory. In particular, access to the following memories may be redirected:

- Program Flash;
- Data Flash;
- OLDA space;
- External EBU space.

12.2.1 Online Data Acquisition (OLDA) Space

Calibration is additionally supported by virtual OLDA memory range. The base address of the virtual OLDA memory range is $A/8FE7\ 0000_H$.

12.3 Overlay Memories

In the following, all the supported overlay memories are described. Note, that depending on the device type only a subset of the listed overlay memories is available. Overlay memory is selected independently for each block by RABRx.OMEM field value.

12.3.1 Local Memory

If present, the Local Memory (LMU) can be selected for overlay. The Local Memory is selected for overlay block x redirection, if RABRx.OMEM value is 6. The base address of the LMU is $B/9000\ 0000_H$. During address translation, the upper 10 address bits are set to $B0_H00_B$ (for target segment A_H) or to 90_H00_B (for target segment 8_H).

12.3.2 Emulation Memory

If present, the Emulation Memory (EMEM) can be selected for overlay. The Emulation Memory is selected for overlay block x redirection, if RABRx.OMEM value is 7. The base address of the Emulation Memory is B/9F00 0000_H. During address translation, the upper 10 address bits are set to BF_H00_B (for target segment A_H) or to 9F_H00_B (for target segment 8_H).

12.3.3 DSPR & PSPR Memory

Data Scratch Memory (DSPR) or Program Scratch Memory (PSPR) from Core 0, 1 or 2 can be selected for overlay. The DSPR or PSPR is selected for overlay block x redirection, if RABRx.OMEM value is 0, 1, or 2. The base address is 7000 0000_H, 6000 0000_H, or 5000 0000_H. During address translation, the upper 10 address bits are set to 70_H00_B, 60_H00_B, or 50_H00_B. Depending on the value set in RABRx.OBASE either DSPR memory (starting at offset 0_H) or PSPR memory (starting at offset 10 0000_H) can be used.

12.4 Global Overlay Control

Overlay can be disabled or enabled individually for each core with OVCENABLE register. If OVCENABLE.OVEN_x bit is cleared no address redirection is permitted on Core x regardless of the remaining register settings. A write to OVCENABLE register does not change any of the remaining register values.

While each overlay block can be activated and deactivated individually by writing its RABRx.OVEN bit, a dedicated functionality is provided for concurrently activating and deactivating multiple blocks. This can be useful in maintaining data consistency across several memory regions. For the purpose of concurrent activation and deactivation overlay blocks are selected in two stages:

- The individual blocks for activation and deactivation are selected with OVC_x_OSEL registers, for each core x independently;
- The set of cores is selected with OVCCON.CSEL field.

Multiple overlay blocks can be simultaneously activated or deactivated with OVCCON.OVSTRT bit. When OVCCON.OVSTRT bit is written with one:

- If OVCCON.CSEL_x bit is written with one, and OVC_x_OSEL.SHOVEN_y bit value is one, overlay block y in core x is activated, and OVC_x_RABR_y.OVEN bit is set;
- If OVCCON.CSEL_x bit is written with one, and OVC_x_OSEL.SHOVEN_y bit value is zero, overlay block y in core x is deactivated, and OVC_x_RABR_y.OVEN bit is cleared;
- If OVCCON.CSEL_x bit is written with zero, the overlay configuration in core x is not effected.

The actions listed above are executed concurrently. The overlay configuration is not changed otherwise. With this function it is possible to switch directly from one set of overlay blocks to another set of overlay blocks.

Data Access Overlay (OVC)

Multiple overlay blocks can be simultaneously deactivated with OVCCON.OVSTP bit. When OVCCON.OVSTP is written with one:

- If OVCCON.CSELx bit is written with one, all the overlay blocks in core x are deactivated, and all OVCx_RABRy.OVEN bits are cleared;
- If OVCCON.CSELx bit is written with zero, the overlay configuration in core x is not effected.

The actions listed above are executed concurrently. The overlay configuration is not changed otherwise.

Note: Overlay should not be enabled or disabled using global OVCENABLE register if any of the blocks are enabled with RABRx.OVEN. Instead, OVSTRT or OVSTP should be used if concurrent block enabling or disabling is required.

When OVCCON.DCINVAL is written with one, all the unmodified (clean) data cache lines in the selected cores are invalidated. The data cache lines containing modified (dirty) data are not effected. The cores not selected with OVCCON.CSEL field are not effected. Data Cache invalidation can be combined with OVSTRT or OVSTP action. This function helps to assure that the CPU can access the new data after the overlay blocks have been activated or deactivated.

Note: OVCCON.CSEL field is written together with OVSTRT, OVSTP and DCINVAL bits in the same register, and only impacts any action triggered by the same write. CSEL, OVSTRT, OVSTP and DCINVAL do not retain the written value and always read as zero.

The OVCCON.OVCONF user control flag is provided, together with its protection bit OVCCON.POVCONF. These bits do not impact the overlay functionality.

When OVCCON register is written with CSELx set and either OVSTRT or OVSTP set, and at the same time OVCx_RABRy register is written, the resulting value of OVCx_RABRy.OVEN bit is not defined. Possibility of such simultaneous access should be avoided.

OVSTRT, OVSTP and DCINVAL actions are not performed when TriCore is in IDLE state.

12.4.1 Global Overlay Control Synchronisation

When OVSTRT, OVSTP or DCINVAL action is requested its execution may be delayed to prevent changing Overlay configuration during an ongoing data load.

Sufficient time should be allowed after an action request, before another action is requested. If a new action is requested while previous action is still pending (due to synchronisation with CPU loads) some actions may be lost.

12.5 Overlay Configuration Change

Overlay block should be disabled, by clearing its RABRx.OVEN bit, before any changes are made to OTARx, OMASKx or RABRx registers. Otherwise, unintended access redirections may occur. Overlay block should only be enabled, if the target address, the overlay memory selection, the redirection address and the mask have all been configured with intended values.

Note: The Overlay Control does not prevent configuring the translation logic incorrectly. In particular, redirection to not implemented or forbidden address range is not prevented.

Special care needs to be taken to synchronise Overlay redirection change to the executed instruction stream if data consistency is required.

External accesses may be buffered in the CPU. External accesses that have not been completed may still be affected by overlay configuration changes. Therefore, it is advised to ensure completion of all pending accesses (for example, by executing DSYNC instruction) before any overlay range is activated or deactivated.

When overlay block is enabled and the same memory location is written through the target address space and read through the redirected address space, or vice-versa, the access synchronisation need to be enforced (with DSYNC and data cache writeback if applicable).

12.6 Access Protection, Attributes, Concurrent Matches

When data access is redirected by Overlay, access protection is applied as follows:

- Target address is subject to CPU Memory Protection (MPU) validation;
- Redirected address is subject to Safety Protection validation.

Physical Memory Attributes for the redirected access are determined basing on the Target Address segment (see CPU Physical Memory Attributes chapter).

Concurrent matches in more than one enabled overlay block are not supported. When an address matches two, or more, of the enabled overlay blocks, an exception is raised and the memory access is not performed. A load operation with multiple matches on overlay ranges, raises a Data Access Synchronous Error (DSE) trap, and a store operation raises Data Access Asynchronous Error (DAE) trap. In such case, relevant trap information registers: Data Synchronous Trap Register (DSTR), Data Asynchronous Trap Register (DATR), and Data Error Address Register (DEADD) are updated, see DMI Registers chapter for more information.

12.7 Overlay Control Registers

OVC block control registers are located in each module that supports data access overlay. OVC global control registers are located in SCU. OVC register access can be restricted by Safety Register Protection.

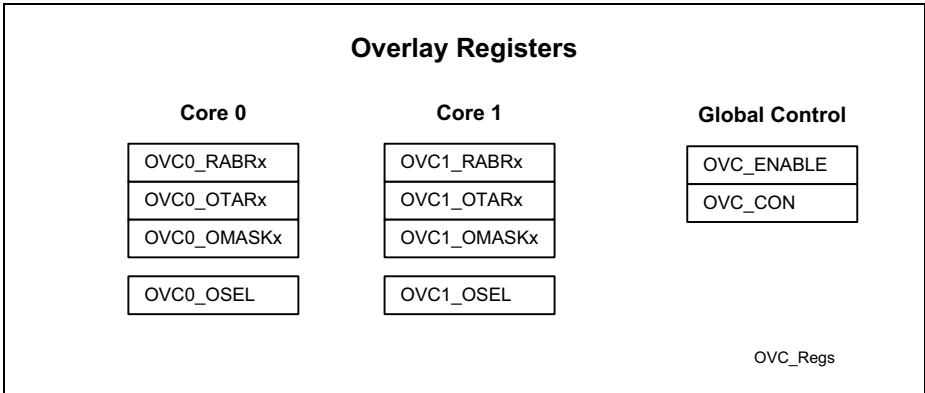


Figure 12-3 Overlay Register Overview

Registers OVCENABLE and OVCCON are described in SCU Chapter.

Table 12-1 Registers Address Space of OVC Registers

Module	Base Address	End Address	Note
OVC0	F880 FB00 _H	F880 FCFF _H	Core0 Overlay
OVC1	F882 FB00 _H	F882 FCFF _H	Core1 Overlay

Table 12-2 Registers Overview, Overlay Block Control

Register ¹⁾ Short Name	Register Long Name	Offset Address	Access Mode ²⁾		Description see
			Read	Write	
OSEL	Overlay Range Select Register	0000 _H	U, SV, 32	P, SV, 32	Page 12-10
RABRx	Redirected Address Base Register x (x = 0-31)	0010 _H + x * C _H	U, SV, 32	P, SV, 32	Page 12-11
OTARx	Overlay Target Address Register x (x = 0-31)	0014 _H + x * C _H	U, SV, 32	P, SV, 32	Page 12-13
OMASKx	Overlay Mask Register x (x = 0-31)	0018 _H + x * C _H	U, SV, 32	P, SV, 32	Page 12-14

1) The OVC register short names are extended with the module name prefix "OVCI_", where i is core number.

- 2) Symbol P: Write access protected with Safety Protection
 Symbol SV: Access permitted in Supervisor Mode
 Symbol U: Access permitted in User Mode 0 or 1
 Symbol 32: Only 32-bit word access is permitted

12.7.1 Block control registers

For each of the 32 overlay memory blocks (indicated by index x), three registers control the overlay operation and the memory selection:

- Redirected Address Base Register RABRx, which selects the overlay memory, holds the block base address within this memory, and contains block enable bit.
- Overlay Target Address Register OTARx, which holds the base address of the memory block being overlaid.
- Overlay Mask Register OMASKx, which determines which bits (from RABRx) are used for the base address (of overlay memory and block) and which bits (of original data address) are directly used as offset within the block.

Additionally, Overlay Range Select Registers OSEL determines which blocks are to be enabled and which blocks are to be disabled when OVCCON.OVSTRT bit is set.

All overlay block control registers are reset to their default values with the application reset. A special debug reset is not considered.

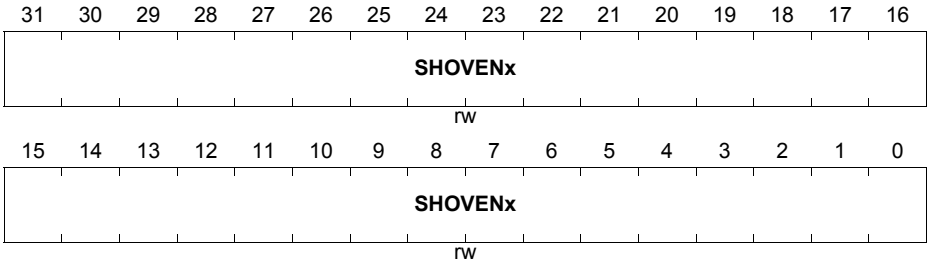
Data Access Overlay (OVC)

OSEL

Overlay Range Select Register

(0000_H)

Reset Value: 0000 0000_H

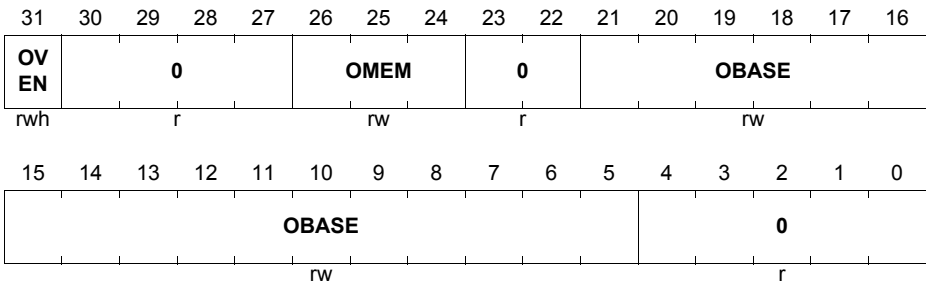


Field	Bits	Type	Description
SHOVENx (x=0-31)	x	rw	<p>Shadow Overlay Enable x</p> <p>0_B Overlay block x is disabled when OVCCON.OVSTRT is set.</p> <p>1_B Overlay block x is enabled when OVCCON.OVSTRT is set.</p> <p>One enable bit is provided for each of the 32 overlay blocks (indicated by index x).</p>

Note: See [Chapter 12.4.1](#) for more information on using OSEL register.

RABRx (x=0-31)
Redirected Address Base Register x

$$(10_H + x \cdot C_H)$$

Reset Value: 0000 0000_H


Field	Bits	Type	Description
OVEN	31	rwh	Overlay Enabled This bit controls whether the overlay function of overlay block x is enabled. 0 _B Overlay function of block x is disabled. 1 _B Overlay function of block x is enabled. This bit can also be changed when OVCCON.OVSTP or OVCCON.OVSTRT is set. See OVCCON register description.
OMEM	[26:24]	rw	Overlay Memory Select Selects overlay memory used for redirection. 0 _H Redirection to Core 0 DSPR/PSPR memory 1 _H Redirection to Core 1 DSPR/PSPR memory 2..5 _H Reserved, do not use 6 _H Redirection to LMU 7 _H Redirection to EMEM
OBASE	[21:5]	rw	Overlay Block Base Address Bits 21..5 of the base address the overlay memory block in the overlay memory. If the corresponding bit in OMASK register is set to one, OBASE bit value is used in the redirection address. If the corresponding bit in OMASK register is set to zero, OBASE bit value is ignored.

Data Access Overlay (OVC)

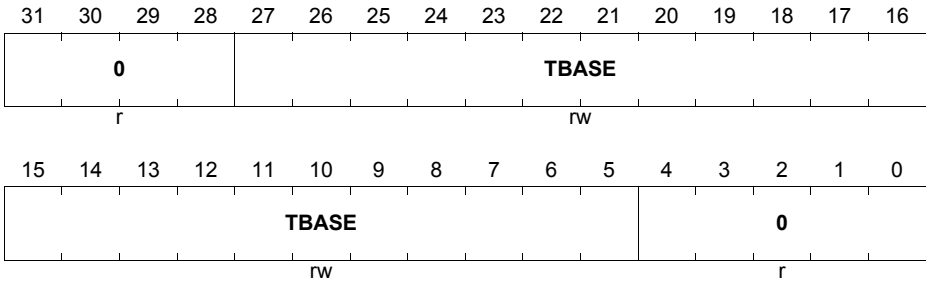
Field	Bits	Type	Description
0	[30:27] [23:22] [4:0]	r	Fixed Value Read as 0; should be written with 0.

OTARx (x=0-31)

Overlay Target Address Register x

$$(14_H + x * C_H)$$

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TBASE	[27:5]	rw	Target Base This field holds the base address of the overlay memory block in the target memory. If the corresponding bit in OMASK register is set to one TBASE bit value is used in the address match. If the corresponding bit in OMASK register is set to zero TBASE bit value is ignored.
0	[31:28] [4:0]	r	Reserved Reads as 0; should be written with 0.

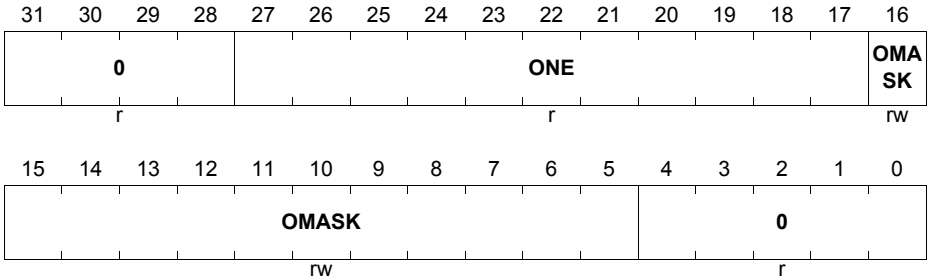
Data Access Overlay (OVC)

OMASKx (x=0-31)

Overlay Mask Register x

(18_H+x*C_H)

Reset Value: 0FFF FFE0_H



Field	Bits	Type	Description
OMASK	[16:5]	rw	<p>Overlay Address Mask</p> <p>This bitfield determines the overlay block size and the bits used for address comparison and translation.</p> <p>000000000000_B, 128 Kbyte block size</p> <p>100000000000_B, 64 Kbyte block size</p> <p>110000000000_B, 32 Kbyte block size</p> <p>[...]</p> <p>111111111110_B, 64 byte block size</p> <p>111111111111_B, 32 byte block size</p> <p>“Zero” bits determine the corresponding address bits which are not used in the address comparison and thus determine the block size; corresponding final address bits are derived from the original data address.</p> <p>“One” bits determine the corresponding address bits which are used for the address comparison; corresponding final address bits are derived from RABRx register in case of address match.</p>
ONE	[27:17]	r	<p>Fixed “1” Values</p> <p>Corresponding address bits are participating in the address comparison. Corresponding final address bits are taken from RABRx.</p>

Data Access Overlay (OVC)

Field	Bits	Type	Description
0	[4:0] [31:28]	r	Fixed “0” Values Corresponding address bits are not used in the address comparison. Corresponding final address bits are taken from the original data address.

12.8 Global overlay control registers

Two registers globally control the overlay operation for all the cores:

- Overlay Enable Register OVCENABLE can be used to disable or enable data access overlay individually for each core;
- Overlay Control Register OVCCON can be used to perform the following action on selected set of cores:
 - concurrently enable / disable selected overlay blocks,
 - concurrently disable overlay blocks,
 - invalidate data cache.

8-Bit Standby Controller (TC2x_SCR)

13 8-Bit Standby Controller (TC2x_SCR)

TC2x_SCR is an 8-bit microcontroller that can continue to run during the standby mode. It is based on the XC800 Core that is compatible with the industry standard 8051 processor. There is an embedded 8 KB XRAM for program code and data.

Key features include a 16-bit general purpose timer with a capture/compare unit (T2CCU) for digital signal generation like pulse generation, pulse width modulation and pulse width measuring. It also includes a Real Time Clock (RTC) to support periodic wake-up in standby mode and an On-Chip Debug Support (OCDS) unit for software development and debugging of XC800-based systems. Local Interconnect Network (LIN) applications are also supported through extended UART features and the provision of LIN low level drivers for TC26x B-Step. For low power applications, various power saving techniques are available for selection by the user. Control of the numerous on-chip peripheral functionalities is achieved by extending the Special Function Register (SFR) address range with an intelligent paging mechanism optimized for interrupt handling.

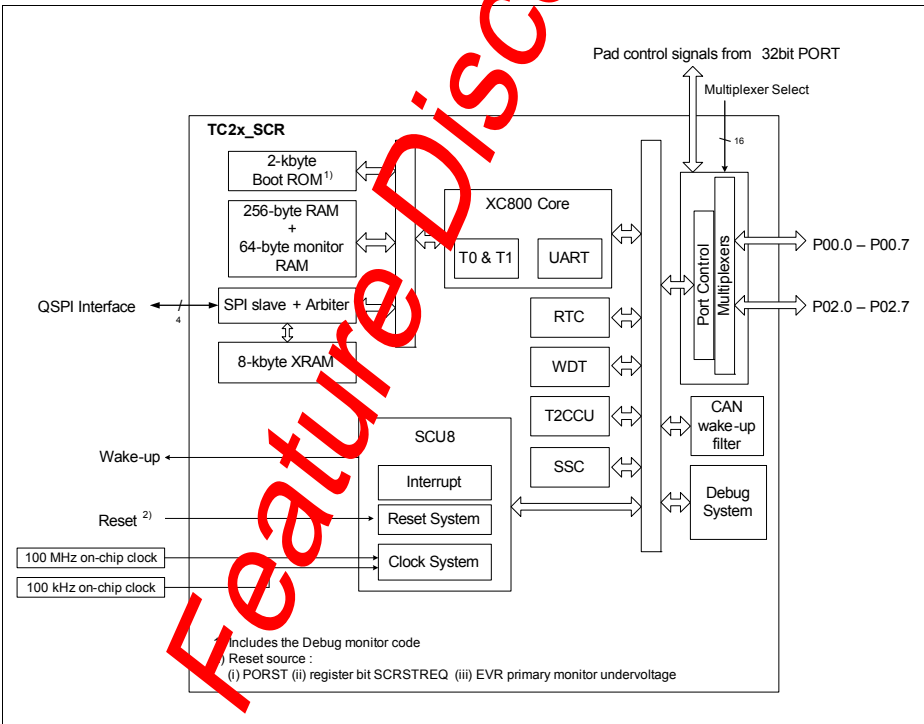


Figure 13-1 TC2x_SCR Block Diagram

8-Bit Standby Controller (TC2x_SCR)
13.1 Features

The following list summarizes the main features of the TC2x_SCR.

- High performance XC800 Core
 - compatible to standard 8051 Core
 - two clocks per machine cycle architecture (for memory access without wait state)
 - two data pointers
 - maximum 20 MHz of core frequency
- On-chip Memory
 - 2-KByte Boot ROM for startup firmware
 - 256-byte RAM; plus 64-byte Monitor RAM
 - 8-Kbyte XRAM for program code and data; accessible by CPUx via SPI interface
- Power saving modes
 - idle mode
 - clock gating control to each peripheral
- Watchdog Timer (WDT) with programmable window feature for refresh operation and warning prior to overflow
- Two general purpose I/O ports
 - able to control the shared pins in P00 and P02
- Three 16-bit timers - Timer 0, Timer 1, Timer 2 of T2CCU
- Capture/compare unit of T2CCU for digital signal generation
- Real-time clock with on-chip oscillator to support periodic wake-up in standby mode
- Full duplex serial interface (UAR)
- Synchronous serial channel (SSC)
- Interrupt supported from TC2x_SCR to TriCore and vice versa.
- On-chip Debug Support via single pin DAP interface (SPD)
 - 1.5-KByte monitor ROM (part of the Boot ROM)
 - 64-byte monitor RAM

Feature Discontinued

13.2 XC800 Core

This chapter describes the XC800 Core.

13.2.1 SFRs of the CPU

The XC800 Core registers occupy direct Internal Data Memory space locations in the range 80_H to FF_H.

13.2.1.1 Stack Pointer (SP, 81_H)

The SP register contains the Stack Pointer. The Stack Pointer is used to load the program counter into internal data memory during LCALL and ACALL instructions and to retrieve the program counter from memory during RET and RETI instructions. Data may also be saved on or retrieved from the stack using PUSH and POP instructions. Instructions that use the stack automatically pre-increment or post-decrement the stack pointer so that the stack pointer always points to the last byte written to the stack, i.e. the top of the stack. On reset, the Stack Pointer is reset to 07_H. This causes the stack to begin at a location = 08_H above register bank zero. The SP can be read or written under software control. The programmer must ensure that the location and size of the stack in internal data memory do not overlap with other application data.

13.2.1.2 Data Pointer (DPTR, 80_H)

The Data Pointer (DPTR) is stored in registers DPL (Data Pointer Low byte) and DPH (Data Pointer High byte) to form 16-bit addresses for External Data Memory accesses (MOVX A,@DPTR and MOVX @DPTR,A), for program byte moves (MOVC A,@A+DPTR) and for indirect program jumps (JMP @A+DPTR).

Two true 16-bit operations are allowed on the Data Pointer: load immediate (MOV DPTR,#data) and increment (INC DPTR).

13.2.1.3 Accumulator (ACC, E0_H)

This register provides one of the operands for most ALU operations. ACC is the symbol for the accumulator register. The mnemonics for accumulator-specific instructions, however, refer to the accumulator simply as "A".

13.2.1.4 B Register (F0_H)

The B register is used during multiply and divide operations to provide the second operand. For other instructions, it can be treated as another scratch pad register.

13.2.1.5 Program Status Word (PSW, D0_H)

The PSW contains several status bits that reflect the current state of the core.

PSW
Program Status Word Register (D0_H) **Reset Value: 00_H**
RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
CY	AC	F0	RS	OV	F1	P	
rwh	rwh	rw	rw	rwh	rw	rh	

Field	Bits	Type	Description
P	0	rh	Parity Flag Set/cleared by hardware after each instruction to indicate an odd/even number of “one” bits in the accumulator, i.e. even parity.
F1	1	rw	General Purpose Flag
OV	2	rwh	Overflow Flag Used by arithmetic instructions.
RS	[4:3]	rw	Register Bank Select These bits are used to select one of the four register banks. 00 _B Bank 0 selected, data address 00 _H - 07 _H 01 _B Bank 1 selected, data address 08 _H - 0F _H 10 _B Bank 2 selected, data address 10 _H - 17 _H 11 _B Bank 3 selected, data address 18 _H - 1F _H
F0	5	rw	General Purpose Flag
AC	6	rwh	Auxiliary Carry Flag Used by instructions which execute BCD operations.
CY	7	rw	Carry Flag Used by arithmetic instructions.

Feature Discontinued

13.2.1.6 Extended Operation Register (EO, A2_H)

The instruction set includes an additional instruction `MOVC @(DPTR++),A`, which writes to program memory implemented as RAM. This instruction may be used both to download code into the program memory when the CPU is initialized and subsequently, to provide software updates. The instruction copies the contents of the accumulator to the code memory at the location pointed to by the current data pointer, then increments the data pointer.

The instruction uses the opcode A5_H, which is the same as the software break instruction TRAP (see [Table 13-1](#)). Bit TRAP_EN in the Extended Operation (EO) register is used to select the instruction executed by the opcode A5_H. When bit TRAP_EN is 0 (default), the A5_H opcode executes the `MOVC` instruction. When bit TRAP_EN is 1, the A5_H opcode executes the software break instruction TRAP, which switches the CPU to debug mode for breakpoint processing.

Register EO is also used to select the current data pointer.

EO
Extended Operation Register (A2_H) **Reset Value: 00_H**
RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
	0		TRAP_EN	0	DPSEL2	DPSEL1	DPSEL0
	r		rw	r	rw	rw	rw

Field	Bits	Type	Description
DPSEL0, DPSEL1, DPSEL2	0, 1, 2	rw	Data Pointer Select These bits are used to select the current data pointer. 000 _B DPTR0 selected 001 _B DPTR1 selected Others: Reserved
TRAP_EN	4	rw	TRAP Enable 0 _B Select <code>MOVC @(DPTR++),A</code> 1 _B Select software TRAP instruction
0	[7:5]	r	Reserved Returns 0 if read; should be written with 0.

13.2.1.7 Power Control Register (PCON, 87_H)

The PCON register provides control for entering idle mode, baud rate control for UART in mode 2, as well as two general purpose flags.

The XC800 Core has two power-saving modes: idle mode and power-down mode. The idle mode can be entered via the PCON register. In idle mode, the clock to the core is disabled while the timers, serial port and interrupt controller continue to run. In power-down mode 1, the clock to the entire core is stopped.

Note: :Power down mode is not available in TC2x_SCF

PCON

Power Control Register

(87_H)
Reset Value: 00_H
RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
SMOD		0		GF1	GF0	0	IDLE
rw		r		rw	rw	r	rw

Field	Bits	Type	Description
IDLE	0	rw	Idle Mode Enable 0 _B Do not enter idle mode 1 _B Enter idle mode
GF0	2	rw	General Purpose Flag Bit 0
GF1	3	rw	General Purpose Flag Bit 1
SMOD	7	rw	Enable Baud Rate Enable This bit controls the baud rate generation for UART in mode 2. 0 _B Do not double the baud rate 1 _B Double the baud rate
0, 0	1, [6:4]	r	Reserved Returns 0 if read; should be written with 0.

13.2.1.8 Interrupt Registers

One non-maskable and fourteen maskable interrupt nodes are available.

Refer to Interrupt chapter for details of the interrupt registers.

13.2.2 SFRs of The Core Peripherals

13.2.2.1 Timer Registers

Two 16-bit timers are provided - Timer 0 (T0) and Timer 1 (T1).
Refer to Timer 0 and Timer 1 chapter for details of the timer registers.

13.2.2.2 UART Registers

The UART uses three SFRs - PCON, SCON and SBUF.
Refer to [Section 13.2.1.7](#) and UART chapter for details of the UART registers.

Feature Discontinued

13.2.3 Instruction Timing

A CPU machine cycle comprises two input clock periods, referred to as Phase 1 (P1) and Phase 2 (P2), that correspond to two different CPU states. Every CPU state within an instruction is referenced by the machine cycle and state number (e.g., C2P1 means the first clock period within machine cycle 2). Memory access takes place during one or both phases of the machine cycle. SFR writes occur only at the end of P2. Instructions are 1, 2, or 3 bytes long and can take 1, 2 or 4 machine cycles to execute. Registers are generally updated and the next opcode pre-fetched at the end of P2 of the last machine cycle for the current instruction.

The XC800 Core supports access to slow memory by using wait cycle(s). Each wait cycle lasts one machine cycle, i.e. two clock periods. For example, in case of a memory requiring one/two wait state(s), the access time is increased by one machine cycle for every byte of opcode/operand fetched.

Figure 13-2 shows the fetch/execute timing relative to the internal states and phases. Execution of an instruction occurs at C1P1. For a 2-byte instruction, the second reading starts at C1P1.

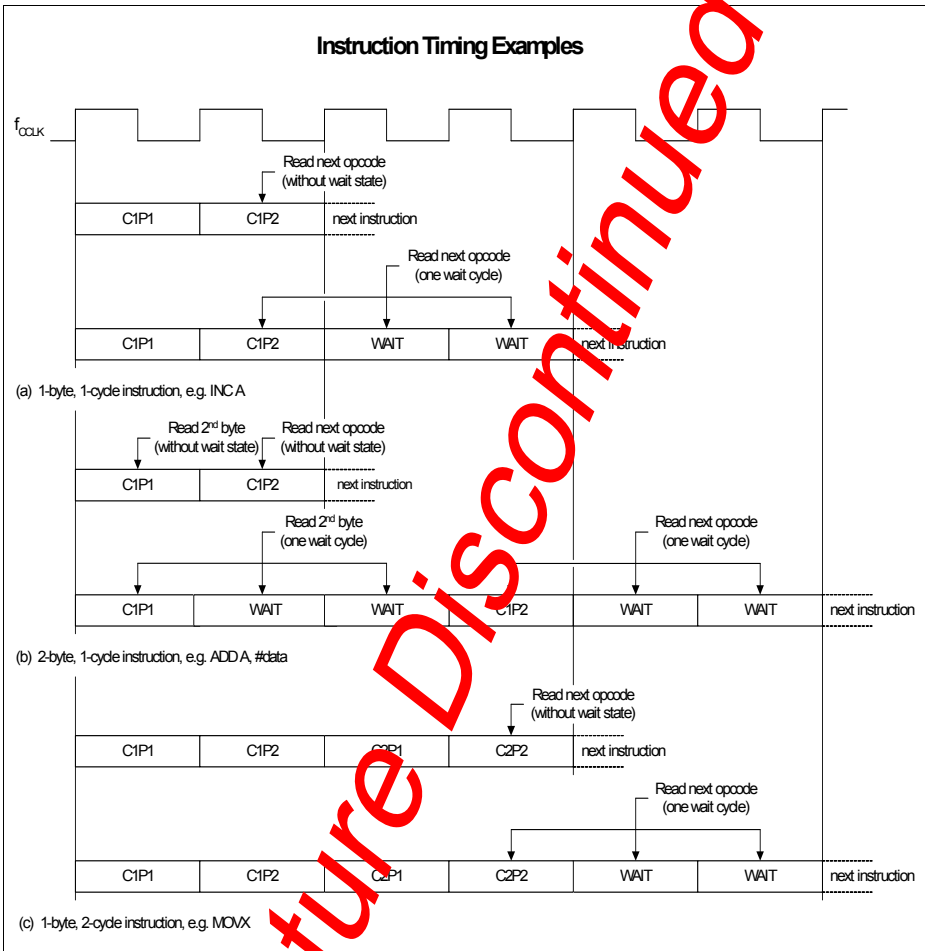
Figure 13-2 (a) shows two timing diagrams for a 1-byte, 1-cycle ($1 \times$ machine cycle) instruction. The first diagram shows the instruction being executed within one machine cycle since the opcode (C1P2) is fetched from a memory without wait state. The second diagram shows the corresponding states of the same instruction being executed over two machine cycles (instruction time extended), with one wait cycle inserted for opcode fetching from the flash memory.

Figure 13-2 (b) shows two timing diagrams for a 2-byte, 1-cycle ($1 \times$ machine cycle) instruction. The first diagram shows the instruction being executed within one machine cycle since the second byte (C1P1) and the opcode (C1P2) are fetched from a memory without wait state. The second diagram shows the corresponding states of the same instruction being executed over three machine cycles (instruction time extended), with one wait cycle inserted for each access to the flash memory. In this case, two wait cycles are inserted in total.

Figure 13-2 (c) shows two timing diagrams of a 1-byte, 2-cycle ($2 \times$ machine cycle) instruction. The first diagram shows the instruction being executed over two machine cycles with the opcode (C2P2) fetched from a memory without wait state. The second diagram shows the corresponding states of the same instruction being executed over three machine cycles (instruction time extended), with one wait cycle inserted for opcode fetching from the slow memory requiring one/two wait state(s).

Note: For instructions that are executed over two or more machine cycles, execution cycle may or may not be extended in case of access to slow memory with one/two wait states. The execution cycle is, nonetheless, guaranteed consistent for each instruction when accessed from slow memory with defined wait state(s).

Reference: [Table 13-1](#).


Figure 13-2 CPU Instruction Timing

The time taken for each instruction includes:

- decoding/executing the fetched opcode
- fetching the operands (for instructions > 1 byte)
- fetching the first byte (opcode) of the next instruction (due to CPU pipeline)

Note: The XC800 Core fetches the opcode of the next instruction while executing the current instruction.

Table 13-1 lists all the instructions supported by the XC800 Core. Instructions are 1, 2 or 3 bytes long as indicated in the 'Bytes' column. Each instruction takes 1, 2 or 4 machine cycles to execute (with no wait cycle). The table gives two values for the number of machine cycles required by each instruction. The first value applies to fetching operand/s and opcode from fast memory (e.g. Boot ROM and XRAM) without wait state. The second value applies to fetching operand/s and opcode (and in some cases accessing data) from slow memory (e.g. Flash) with wait cycles inserted due to memory requiring one/two wait state(s). One machine cycle comprises two ϕ CLK clock cycles.

Table 13-1 Instruction Table

Mnemonic	Hex Code	Bytes	Machine Cycles (no wait state)	Machine Cycles (one wait state ¹⁾)
ARITHMETIC				
ADD A,Rn	28-2F	1	1	2
ADD A,dir	25	2	1	3
ADD A,@Ri	26-27	1	1	2
ADD A,#data	24	2	1	3
ADDC A,Rn	38-3F	1	1	2
ADDC A,dir	35	2	1	3
ADDC A,@Ri	36-37	1	1	2
ADDC A,#data	34	2	1	3
SUBB A,Rn	98-9F	1	1	2
SUBB A,dir	95	2	1	3
SUBB A,@Ri	96-97	1	1	2
SUBB A,#data	94	2	1	3
INC A	04	1	1	2
INC Rn	08-0F	1	1	2
INC dir	05	2	1	3
INC @Ri	06-07	1	1	2
DEC A	14	1	1	2
DEC Rn	18-1F	1	1	2
DEC dir	15	2	1	3
DEC @Ri	16-17	1	1	2
INC DPTR	A3	1	2	2

Table 13-1 Instruction Table (cont'd)

Mnemonic	Hex Code	Bytes	Machine Cycles (no wait state)	Machine Cycles (one wait state ¹⁾)
MUL AB	A4	1	4	4
DIV AB	84	1	4	4
DA A	D4	1	1	2
LOGICAL				
ANL A,Rn	58-5F	1	1	2
ANL A,dir	55	2	1	3
ANL A,@Ri	56-57	1	1	2
ANL A,#data	54	2	1	3
ANL dir,A	52	2	1	3
ANL dir,#data	53	3	2	5
ORL A,Rn	48-4F	1	1	2
ORL A,dir	45	2	1	3
ORL A,@Ri	46-47	1	1	2
ORL A,#data	44	2	1	3
ORL dir,A	42	2	1	3
ORL dir,#data	43	3	2	5
XRL A,Rn	68-6F	1	1	2
XRL A,dir	65	2	1	3
XRL A,@Ri	66-67	1	1	2
XRL A,#data	64	2	1	3
XRL dir,A	62	2	1	3
XRL dir,#data	63	3	2	5
CLR A	14	1	1	2
CPL A	14	1	1	2
SWAP A	C4	1	1	2
RL A	23	1	1	2
RLC A	33	1	1	2
RR A	03	1	1	2
RRC A	13	1	1	2

Table 13-1 Instruction Table (cont'd)

Mnemonic	Hex Code	Bytes	Machine Cycles (no wait state)	Machine Cycles (one wait state ¹⁾)
DATA TRANSFER				
MOV A,Rn	E8-EF	1	1	2
MOV A,dir	E5	2	1	3
MOV A,@Ri	E6-E7	1	1	2
MOV A,#data	74	2	1	3
MOV Rn,A	F8-FF	1	1	2
MOV Rn,dir	A8-AF	2	2	4
MOV Rn,#data	78-7F	2	1	3
MOV dir,A	F5	2	1	3
MOV dir,Rn	88-8F	2	2	4
MOV dir,dir	85	3	2	5
MOV dir,@Ri	86-87	2	2	4
MOV dir,#data	75	3	2	5
MOV @Ri,A	F6-F7	1	1	2
MOV @Ri,dir	A6-A7	2	2	4
MOV @Ri,#data	76-77	2	1	3
MOV DPTR,#data	90	3	2	5
MOVC A,@A+DPTR	93	1	2	3 or 4 ²⁾
MOVC A,@A+PC	83	1	2	3 or 4 ²⁾
MOVX A,@Ri	E2-E3	1	2	3
MOVX A,@DPTR	E1	1	2	3
MOVX @Ri,A	F2-F3	1	2	3
MOVX @DPTR,A	F1	1	2	3
PUSH dir	C0	2	2	4
POP dir	D0	2	2	4
XCH A,Rn	C8-CF	1	1	2
XCH A,dir	C5	2	1	3
XCH A,@Ri	C6-C7	1	1	2
XCHD A,@Ri	D6-D7	1	1	2

Table 13-1 Instruction Table (cont'd)

Mnemonic	Hex Code	Bytes	Machine Cycles (no wait state)	Machine Cycles (one wait state ¹⁾)
BOOLEAN				
CLR C	C3	1	1	2
CLR bit	C2	2	1	3
SETB C	D3	1	1	2
SETB bit	D2	2	1	3
CPL C	B3	1	1	2
CPL bit	B2	2	1	3
ANL C,bit	82	2	2	4
ANL C,/bit	B0	2	2	4
ORL C,bit	72	2	2	4
ORL C,/bit	A0	2	2	4
MOV C,bit	A2	2	1	3
MOV bit,C	92	2	2	4
BRANCHING³⁾				
ACALL addr11	11->F1	2	2	4
LCALL addr16	12	3	2	5
RET	22	1	2	2 or 3 ²⁾
RETI	32	1	2	2 or 3 ²⁾
AJMP addr 11	01->E1	2	2	4
LJMP addr 16	02	3	2	5
SJMP rel	80	2	2	4
JC rel	40	2	2	4
JNC rel	50	2	2	4
JB bit,rel	70	3	2	5
JNB bit,rel	30	3	2	5
JBC bit,rel	10	3	2	5
JMP @A+DPTR	73	1	2	2 or 3 ²⁾
JZ rel	60	2	2	4
JNZ rel	70	2	2	4

Table 13-1 Instruction Table (cont'd)

Mnemonic	Hex Code	Bytes	Machine Cycles (no wait state)	Machine Cycles (one wait state ¹⁾)
CJNE A,dir,rel	B5	3	2	5
CJNE A,#d,rel	B4	3	2	5
CJNE Rn,#d,rel	B8-BF	3	2	5
CJNE @Ri,#d,rel	B6-B7	3	2	5
DJNZ Rn,rel	D8-DF	2	2	4
DJNZ dir,rel	D5	3	2	5
MISCELLANEOUS				
NOP	00	1	1	2
ADDITIONAL INSTRUCTIONS				
MOVC @(DPTR++),A	A5	1	1	2 or 3 ²⁾
TRAP	A5	1	1	–

1) In case of fetch from slow memory requiring only 1 wait state, no wait cycle may be required per the instruction fetched (normally for opcodes that do not require fetch in the next cycle).

2) Depending on whether the operation is accessing memory with zero or one/two wait state.

3) For branch instructions, the instruction time may vary depending on jump destination.

Feature Discontinued

13.3 Memory Organization

The TC2x_SCR CPU operates in the following four address spaces:

- 2 Kbytes of Boot ROM program memory
- 256 bytes of internal RAM data memory
- 8 Kbytes of XRAM memory
 - ECC protected
 - XRAM can be read/written as program memory or external data memory
- a 128-byte Special Function Register area

Figure 13-3 illustrates the memory address spaces of the TC2x_SCR.

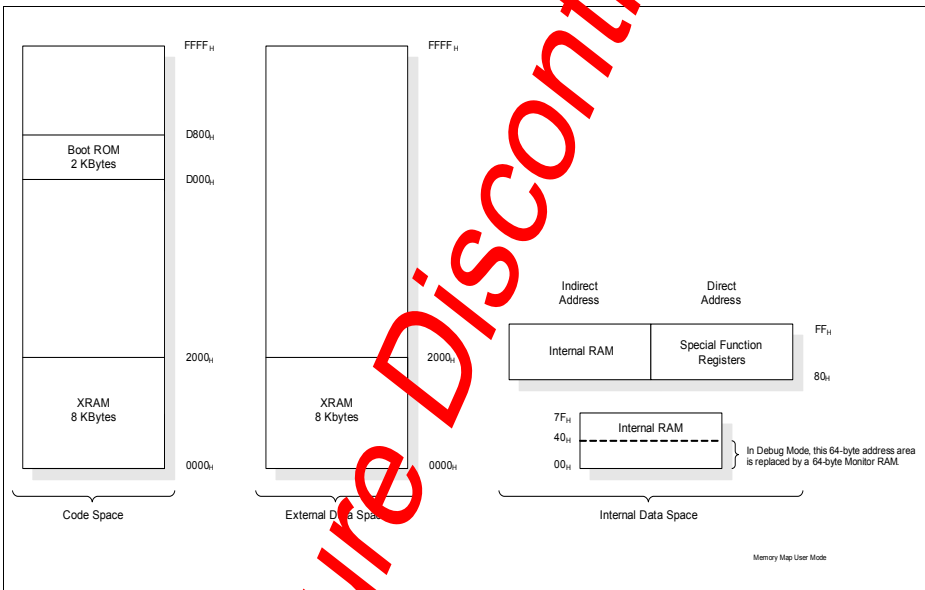


Figure 13-3 Memory Map of the TC2x_SCR

13.3.1 Program Memory

The code space is theoretically 64 KBytes. However, only access to defined program memory (as shown in memory map figure) is supported. For TC2x_SCR, defined code space is occupied by on-chip memories.

13.3.2 Data Memory

The data space consists of an internal and external data space. Access to internal and external data space are distinguished by different sets of instruction opcodes. In TC2x_SCR, on-chip XRAM is located in external data space and accessed by MOVX instructions. TC2x_SCR does not support access to external (off-chip) memory. Internal data space is occupied by Internal RAM (IRAM) and Special Function Registers (SFRs), distinguished by direct or indirect addressing.

13.3.2.1 Internal Data Memory

The internal data memory is divided into two physically separate and distinct blocks: the 256-byte RAM and the 128-byte Special Function Register (SFR) area. While the upper 128 bytes of RAM and the SFR area share the same address locations, they are accessed through different addressing modes. The lower 128 bytes of RAM can be accessed through either direct or register indirect addressing, while the upper 128 bytes of RAM can be accessed through register indirect addressing only. The SFRs are accessible through direct addressing.

The 16 bytes of RAM that occupy addresses from 20_H to 2F_H are bitaddressable. RAM occupying direct addresses from 30_H to 7F_H can be used as scratch pad registers or used for the stack.

Feature Discontinued

13.3.2.2 External Data Memory

The 8-Kbyte XRAM is mapped to both the external data memory area and the program memory area. In TC2x_SCR, the XRAM can be accessed by both the TC2x_SCR core and CPU0. The arbiter module determines which CPU has control of the XRAM at any one time. CPU0 accesses the XRAM using the SPI processor. A SPI interface is established between the main TC2x system and the SPI slave interface (inside the Arbiter module) of the TC2x_SCR for this purpose.

TC2x_SCR core accesses the XRAM using either the 'MOVX' or 'MOVC' instructions. The 'MOVX' instructions for XRAM access use either 8-bit or 16-bit indirect addresses, as shown in [Table 13-2](#).

Table 13-2 Instructions for External Data Memory Access

Type of Access	8-bit Addressing Mode	16-bit Addressing Mode
Read	MOVX A, @DPTR, A	MOVX A, @Ri
Write	MOVX @DPTR, A	MOVX @Ri, A

While the DPTR register is used for 16-bit addressing, either register R0 or R1 is used to form the 8-bit address. The upper byte of the XRAM address during execution of the 8-bit accesses is defined by the value stored in register XADDRH. Hence, the write instruction for setting the higher order of XRAM address in register XADDRH must precede the 'MOVX' instruction.

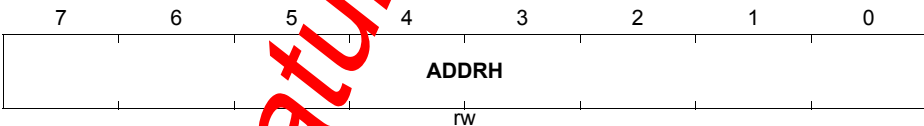
A write access to a undefined external data memory address has no effect while a read access returns the last external data on the bus.

XADDRH

On-Chip XRAM Address Higher Order (F_{2H})

Reset Value: 00_H

RMAP: 0, PAGE: X



Field	Bits	Type	Description
ADDRH	[7:0]	rw	Higher Order of On-chip XRAM Address The value for TC2x_SCR is 00 _H .

13.3.3 Special Function Registers

The Special Function Registers (SFRs) occupy direct internal data memory space in the range 80_H to FF_H . All registers, except the program counter, reside in the SFR area. The SFRs include pointers and registers that provide an interface between the CPU and the on-chip peripherals. As the 128-SFR range is less than the total number of registers required, address extension mechanisms are required to increase the number of addressable SFRs. The address extension mechanisms include:

- Mapping
- Paging

13.3.3.1 Address Extension by Mapping

Address extension is performed at the system level by mapping. The SFR area is extended into two portions: the standard (non-mapped) SFR area and the mapped SFR area. Each portion supports the same address range 80_H to FF_H , bringing the number of addressable SFRs to 256. The extended address range is not directly controlled by the CPU instruction itself, but is derived from bit RMAP in the system control register SYSCON0 at address $8F_H$. To access SFRs in the mapped area, bit RMAP in SFR SYSCON0 must be set. However, the SFRs in the standard area can be accessed by clearing bit RMAP. **Figure 13-4** shows how the SFR area can be selected.

As long as bit RMAP is set, the mapped SFR area can be accessed. This bit is not cleared automatically by hardware. Thus, before standard/mapped registers are accessed, bit RMAP must be cleared/set, respectively, by software.

Feature Discontinued

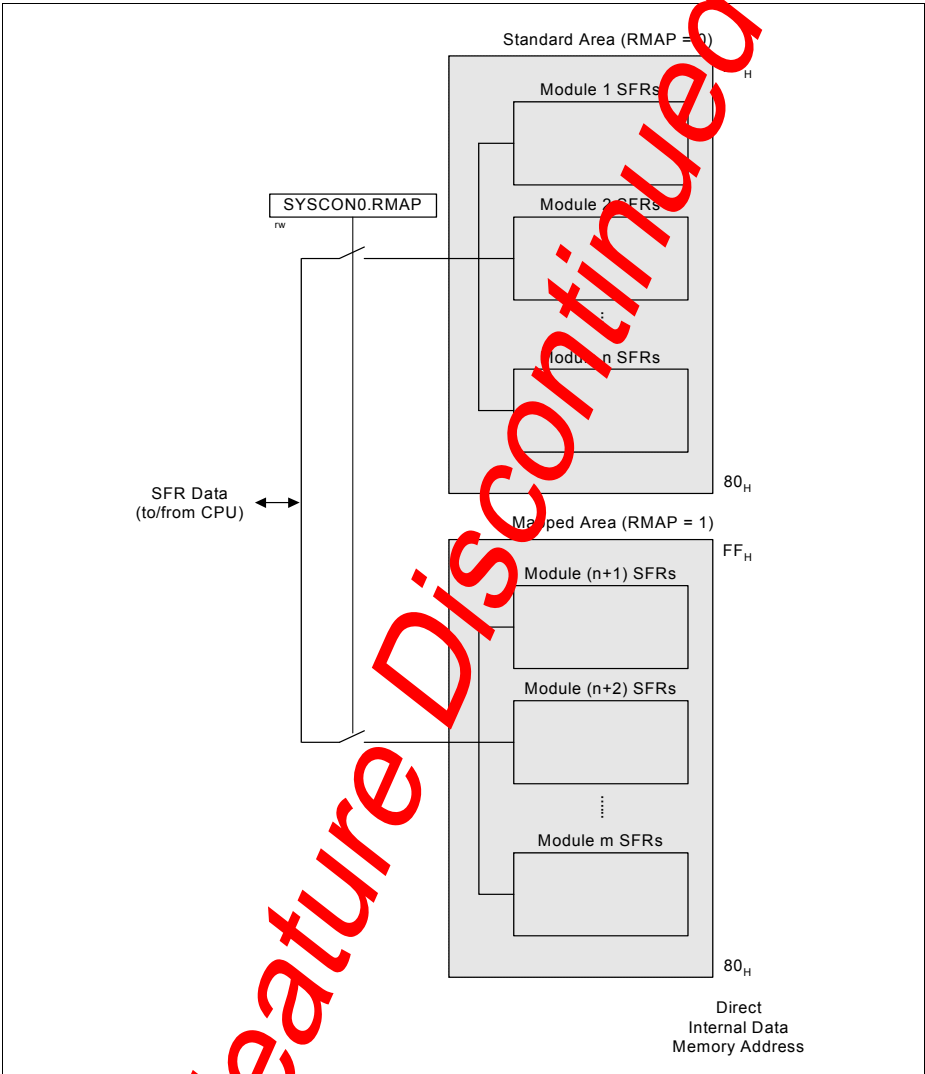


Figure 13-4 Address Extension by Mapping

13.3.3.2 System Control Register 0

The SYSCON0 register contains the bit to select the SFR mapping.

SYSCON0

System control Register 0

(8F_H)

Reset Value: 04_H

RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
		0			AMSEL	0	RMAP
		r			rw	r	rw

Field	Bits	Type	Description
RMAP	0	rw	Special Function Register Map Control 0 _B Accessed to non-mapped (standard) special function register area. 1 _B Accessed to mapped special function register area.
AMSEL	2	rw	Active Memory Map Select 0 _B Active memory map 0: Boot ROM is only mapped from D000 _H . 1 _B Active memory map 1: Last 512 bytes of Boot ROM is mapped to both 0000 _H and D600 _H .
0	[7:3], 1	r	Reserved Returns 0 if read; should be written with 0.

Feature Discontinued

13.3.3.3 Address Extension by Paging

Address extension is further performed at the module level by paging. Certain peripherals have a built-in local address extension mechanism for increasing the number of addressable SFRs. The extended address range is not directly controlled by the CPU instruction itself, but is derived from bit field PAGE in the module page register MOD_PAGE. Hence, the bit field PAGE must be programmed before accessing the SFRs of the target module. Each module may contain a different number of pages and a different number of SFRs per page, depending on the specific requirement. Besides setting the correct RMAP bit value to select the SFR area, the user must also ensure that a valid PAGE is selected to target the desired SFRs.

Figure 13-5 shows how a page inside the extended address range can be selected.

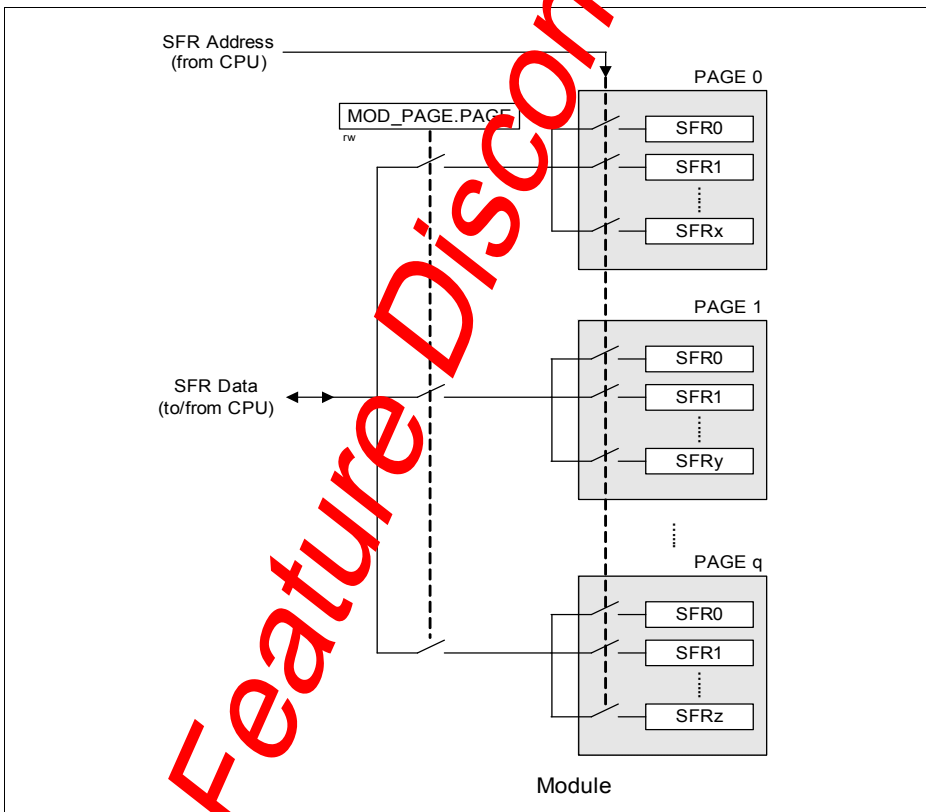


Figure 13-5 Address Extension by Paging

In order to access a register located in a page other than the current one, the current page must be exited. This is done by reprogramming the bit field PAGE in the page register. Only then can the desired access be performed.

If an interrupt routine is initiated between the page register access and the module register access, and the interrupt needs to access a register located in another page, the current page setting can be saved, the new one programmed, and the old page setting restored. This is possible with the storage fields MOD_STx (x = 0 - 3) for the save and restore action of the current page setting. By indicating which storage bit field should be used in parallel with the new page value, a single write operation can:

- Save the contents of PAGE in MOD_STx before overwriting with the new value (this is done at the beginning of the interrupt routine to save the current page setting and program the new page number); or
- Overwrite the contents of PAGE with the contents of MOD_STx, ignoring the value written to the bit positions of PAGE (this is done at the end of the interrupt routine to restore the previous page setting before the interrupt occurred)

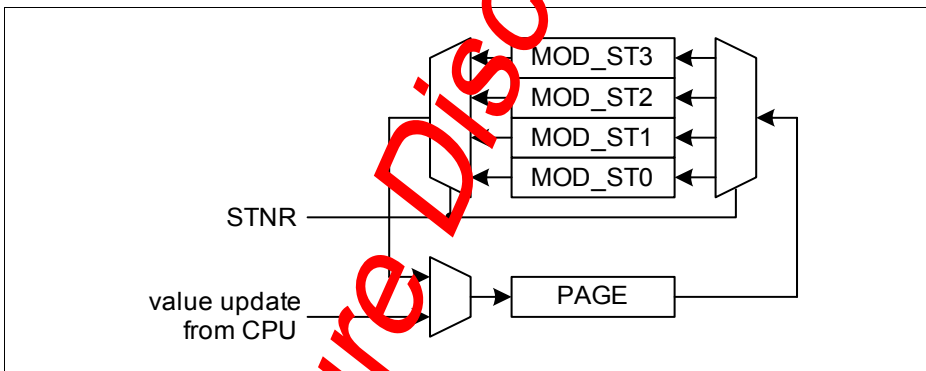


Figure 13-6 Storage Elements for Paging

With this mechanism, a certain number of interrupt routines (or other routines) can perform page changes without loading and storing the previously used page information. The use of only write operations makes the system simpler and faster. Consequently, this mechanism significantly improves the performance of short interrupt routines.

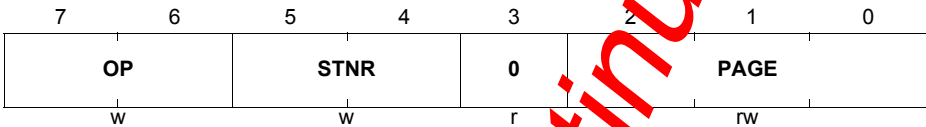
13.3.3.4 Page Register

The page register has the following definition:

MOD_PAGE

Page Register for module MOD

Reset Value: 00_H



Field	Bits	Type	Description
PAGE	[2:0]	rw	Page Bits When written, the value indicates the new page. When read, the value indicates the currently active page.
STNR	[5:4]	w	Storage Number This number indicates which storage bit field is the target of the operation defined by bit field OP. If OP = 10 _B , the contents of PAGE are saved in MOD_STx before being overwritten with the new value. If OP = 11 _B , the contents of PAGE are overwritten by the contents of MOD_STx. The value written to the bit positions of PAGE is ignored. 00 _B MOD_ST0 is selected. 01 _B MOD_ST1 is selected. 10 _B MOD_ST2 is selected. 11 _B MOD_ST3 is selected.

Feature Discontinued

Field	Bits	Type	Description
OP	[7:6]	w	Operation 0x _B Manual page mode. The value of STNR is ignored and PAGE is directly written. 10 _B New page programming with automatic page saving. The value written to the bit positions of PAGE is stored. In parallel, the previous contents of PAGE are saved in the storage bit field STx indicated by STNR. 11 _B Automatic restore page action. The value written to the bit positions PAGE is ignored and instead, PAGE is overwritten by the contents of the storage bit field STx indicated by STNR.
0	3	r	Reserved Returns 0 if read, should be written with 0.

13.3.3.5 Bit-Addressing

SFRs that have addresses in the form of 1x...X000_B (e.g., 80_H, 88_H, 90_H, ..., F0_H, F8_H) are bit addressable.

Feature Discontinued

13.3.3.6 Bit Protection Scheme

The bit protection scheme prevents direct software writing of selected bits (i.e., protected bits) using the PASSWD register. When the bit field MODE is 11_B , writing 10011_B to the bit field PASS opens access to writing of all protected bits, and writing 10101_B to the bit field PASS closes access to writing of all protected bits. In both cases, the value of the bit field MODE is not changed even if PASSWD register is written with 98_H or $A8_H$. It can only be changed when bit field PASS is written with 11000_B ; for example, writing $C0_H$ to PASSWD register disables the bit protection scheme.

Note that access is opened for maximum 32 CCLKs if the "close access" password is not written. If "open access" password is written again before the end of 32 CCLK cycles, there will be a recount of 32 CCLK cycles.

PASSWD

Password Register

(F1_H)

Reset Value: 07_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PASS					PROTECT _S	MODE	
wh					rh	rw	

Field	Bits	Type	Description
MODE	[1:0]	rw	<p>Bit Protection Scheme Control bits</p> <p>00_B Scheme disabled - direct access to the protected bits is allowed.</p> <p>11_B Scheme enabled - the bit field PASS has to be written with the passwords to open and close the access to protected bits. (default)</p> <p>Others: Scheme enabled</p> <p>These two bits cannot be written directly. To change the value between 11_B and 00_B, the bit field PASS must be written with 11000_B; only then, will the MODE[1:0] be registered.</p>
PROTECT_S	2	rh	<p>Bit Protection Signal Status bit</p> <p>This bit shows the status of the protection.</p> <p>0_B Software is able to write to all protected bits.</p> <p>1_B Software is unable to write to any protected bits.</p>

Field	Bits	Type	Description
PASS	[7:3]	wh	<p>Password bits</p> <p>The Bit Protection Scheme (bpc) recognizes three patterns.</p> <p>11000_B Enables writing of the bit field MODE.</p> <p>10011_B Opens access to writing of all protected bits.</p> <p>10101_B Closes access to writing of all protected bits.</p>

Feature Discontinued

13.3.4 Memory Control Unit

The Memory Control Unit (MCU) is the interface between the CPU and the memories, providing three functions to the TC2x_SCR:

- Active memory map selection
- Allows one physical memory (i.e. XRAM) to be accessed as both program and data memory at the same time
- Memory protection

Feature Discontinued

13.3.4.1 Active Memory Map Select

After a reset, the CPU will always start by executing the Boot ROM code in active memory map 1. In active memory map 1, the Boot ROM occupies the program memory address space $0000_H - 01FF_H$ and $D600_H - D7FF_H$, with the remaining program memory address space disabled. The Boot ROM start-up procedure will first jump to $D60X_H$ before switching to active memory map 0 as shown in [Figure 13-7](#). As a result, the Boot ROM memory will be mapped to only $D000_H - D7FF_H$. Also, the remaining program memory blocks (XRAM) are enabled. After the active memory map switch, the remaining Boot ROM start-up procedure will be executed from $D600_H$. The memory organization of the TC26x B-Step shown in this document is after the active memory map switch, i.e. active memory map 0, where the different operating modes are executed.

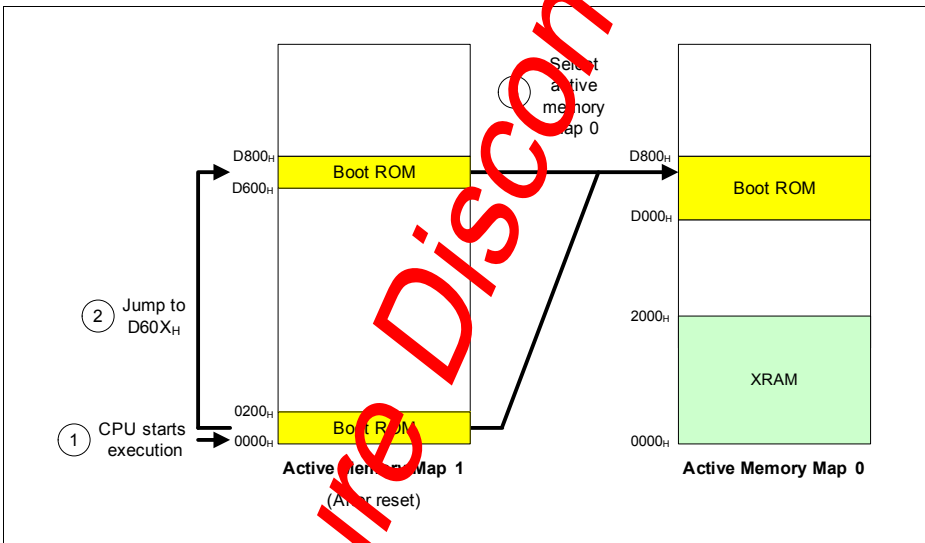


Figure 13-7 Active Memory Map Select

13.3.4.2 Program and XRAM Data Bus Control

The MCU allows one physical memory or memories (non-overlapping) residing in the address range from $0000_H - 1FFF_H$, to be accessed as both program memory and external data memory at the same time.

The 'MOVC @(I,PTX++),A' instruction is used to download software into the program memory where this is implemented as RAM. This instruction can also be used subsequently to modify contents of the program memory. The 'MOVX' instructions are used to access the XRAM as external data memory.

13.3.4.3 Memory Protection Unit

The TC2x_SCR consists of the following memory blocks:

- Boot ROM
- XRAM

The target of the memory protection scheme is to prevent unauthorized read out of critical data from the Boot ROM.

The memory protection scheme is hardware-based and is used to protect the Boot ROM; MOVc read instructions executed out of the “unsafe” program memory address range (e.g. XRAM) that target the Boot ROM are blocked when protection is enabled.

13.3.4.4 Boot ROM Protection

The Boot ROM protection is enabled by default, and the following accesses are restricted:

- MOVc read instructions that are executed from the XRAM will be blocked if the Boot ROM addresses are targeted

Figure 13-8 shows the possible MOVc execution paths (fetch + execute) when the Boot ROM protection is enabled.

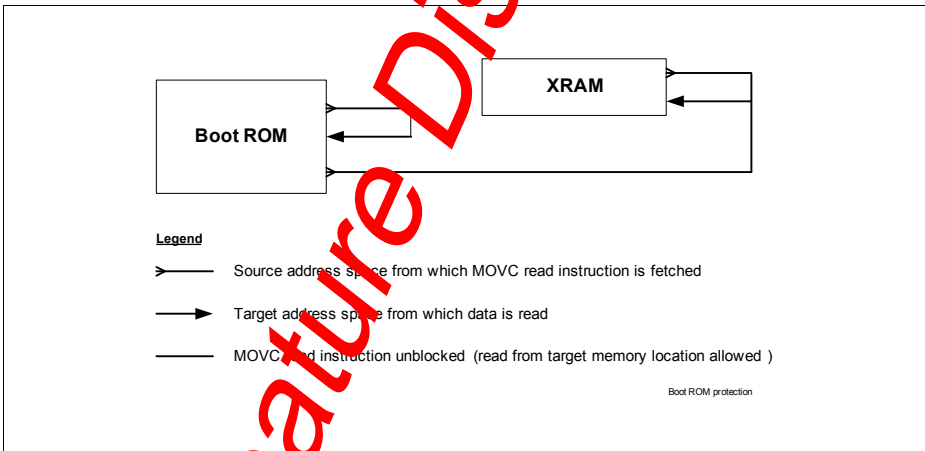


Figure 13-8 Boot ROM Protection Mode

If the Boot ROM protection is enabled:

- MOVc instructions that are executed from the the XRAM can target itself
- MOVc instructions that are executed from the Boot ROM can target itself or the XRAM

13.3.5 Arbiter

The arbiter serves the following purposes:

- Provides the SPI slave interface for communication with the main TC2x system
- Arbitrates the XRAM access between TC2x_SCR core and CPU0 in the main TC2x system (access via SPI slave interface)

13.3.5.1 SPI Slave Interface

The SPI slave interface supports full-duplex serial synchronous communication with the main TC2x system. It activates the Arbiter state machine upon reception of a complete 32-bit SPI data word from the SPI master in the main TC2x system.

Transfer Characteristics:

- Full-duplex operation
- 32-bit data width
- MSB first
- Idle low master shift clock
- Transmit data with clock rising edge; latches in receive data with falling edge
- Maximum baud rate of 50 Mbaud
- Minimum inter-word/frame idle time of 8 PCLK cycles for consecutive writes, 16 PCLK cycles for consecutive reads
- Minimum chip select hold time from last SCLK latching edge of 1 PCLK cycle

The SPI master at the main TC2x system first asserts the chip select signal (active low), which causes the transmit buffer (TBUF) of the SPI slave to reset to default 5555'5555_H. The 32-bit shift register in SPI slave is then loaded with the transmit buffer contents.

When the master shift clock is detected, the SPI slave shifts out data onto the master data-in line with the rising edge of the clock and shifts in data from the master data-out line with the falling edge.

Once the complete 32-bit data is received, the data will be loaded into the receive buffer (RBUF). The contents of TBUF is prepared by the Arbiter state machine.

The shift register is reset when the chip select signal is deasserted by the master SPI.

SPI Software Protocol

The communication between the main TC2x system and XRAM is based on a pre-fixed SPI software protocol. See [Table 13-3](#). The protocol frame can consist of a variable number of data words. Each data word must be of 32-bit width and contains an upper 16-bit command and a lower 16-bit data. Therefore, XRAM access is always a two-byte access, aligned to even address. Transmission is MSB first and master transmit on the rising edge, receives on the falling edge. A XRAM read or write access from main TC2x

system is determined by the MSB of the command word. 1 indicates a write access while a 0 indicates a read access.

Table 13-3 SPI Software Protocol Definition

Bits	Description
31	Read/Write indication: - 1 indicates a write access to XRAM. - 0 indicates a read access from XRAM.
[30:29]	Control bits: - Should be written with 0. - When the 32-bit SPI data word contains the value FFFF'FFFF _H , it indicates that the current word is the last of the frame, i.e. an End-of-Frame (EOF) condition. This word will then be ignored by the arbiter state machine.
[28:16]	13-bit XRAM address. <i>Note: For smaller XRAM where the address width is less than 13 bits, the unused uppermost bits should be written with 0.</i>
[15:0]	16-bit data.

For a XRAM read access, the SPI master in the main TC2x system first transmits the 32-bit data word containing the read command and data to the SPI slave. For a read command, the data from SPI master is not used. Since the SPI communication protocol is full-duplex, the SPI slave will at the same time return a 32-bit word containing an arbitrary value of 5555'5555_H. The SPI slave will only respond with the targeted read data in the next transmission period, which is triggered by the SPI master sending a second word. To read the last data from XRAM, the SPI master can send an EOF containing the value FFFF'FFFF_H.

Similarly for a XRAM write access, the SPI master in the main TC2x system first transmits the 32-bit word containing the write command and the data to be written to XRAM, to the SPI slave. The SPI slave always return a 32-bit word containing the arbitrary value of 5555'5555_H for write commands. The actual write to XRAM will be performed by the arbiter state machine some PCLK cycles after the SPI slave has received the corresponding write command.

Examples of data transfers using the SPI software protocol is shown in [Figure 13-9](#).

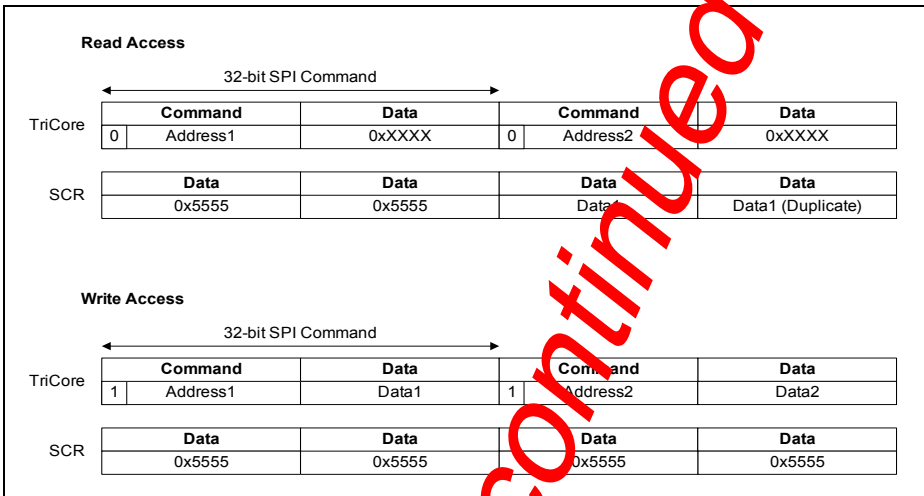


Figure 13-9 Data Transfers Examples using SPI Software Protocol

13.3.5.2 Arbiter State Machine

The Arbiter State Machine is clocked by PCLK. It is activated upon the reception of the complete 32-bit SPI data word from the main TC2x system.

If the word is an EOF, the state machine will do nothing and exit back to the default Idle state, else it will assert the mwait to CPU for 9 PCLK cycles. On the last of these 9 cycles, the state machine will generate the required signals and access the XRAM. The length of the mwait signal generation ensures that the current TC2x_SCR core instruction execution can be stalled temporarily to allow the XRAM access by the Arbiter State Machine.

Note: For each XRAM access, up to 5 wait states (i.e. 10 PCLK cycles) may be inserted into the execution cycle of a TC2x_SCR core instruction.

The access to XRAM is a single cycle access. However, to ensure that the XC800 core is in a safe state during the SPI XRAM access, a minimum of 8 PCLK cycles are required in between SPI words/frames sent from the master if the preceding word contains a write command. If the preceding word contains a read command, a minimum of 16 PCLK cycles are required. If the SPI idle time conditions are not met, the read/written data from/to the XRAM cannot be guaranteed.

In addition, a minimum of 1 PCLK is required for the chip select hold time from the last SCLK latching edge.

Figure 13-10 shows these inter-word/frame and chip select hold times.

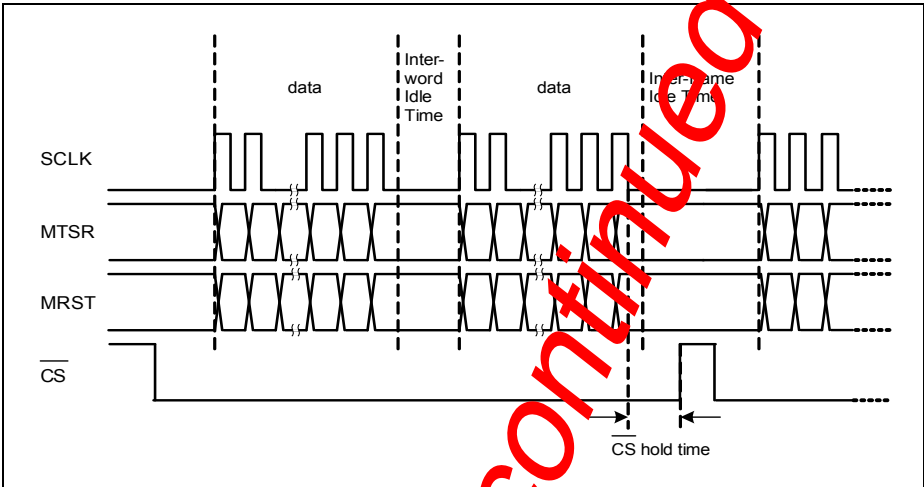


Figure 13-10 Inter-word/frame and chip select hold times

A signal, `spi_slave_busy`, is provided at the Arbitrator interface to indicate that the arbiter is busy with a current SPI XRAM data access. The signal is asserted four PCLK cycles after the detection of the last SPI clock of the current word and it will be deasserted after the SPI XRAM access is completed. The SPI master should transmit the next word/frame only after the deassertion of the `spi_slave_busy` signal.

Feature Discontinued

Feature Discontinued

13.4 TC2x_SCR Firmware

This chapter describes the BootROM firmware for TC26x B-Step. It contains the following sections.

- BootROM Overview
- Startup Procedure
- On-Chip Debug Support (OCDS)

13.4.1 Overview

TC26x B-Step BootROM firmware provides the following features:

- Startup procedure for stable operation of TC26x B-Step chip
- Basic debugging capability via OCDS mode

Startup procedure initializes the TC26x B-Step to a known state to ensure stable operation. The startup procedure consists of all the settings and initialization to be done after SCR reset, before branching to various boot modes. On-Chip Debug Support provides basic debugging capability to the user.

Please refer to the relevant sections for more details.

13.4.1.1 BootROM location

The size of the BootROM is 2KByte and it operates in two different memory maps:

- In Active Memory Map 1 (AMSEL=1), BootROM is mapped to two address spaces, 0000_H - 07FF_H and D000_H - D7FF_H.
- In Active Memory Map 0 (AMSEL=0), BootROM is mapped to address space D000_H - D7FF_H.

Upon reset, the active memory map is 1 and the BootROM code is executed from address 0000_H. At the beginning of the BootROM code, a jump instruction to address D60_{xH} is executed, followed by an instruction to clear the AMSEL bit. This is to switch the active memory map to 0. All other BootROM instructions will be executed in active memory map 0.

13.4.1.2 Boot Mode Option

Entry to various boot modes such as User Mode 0, User Mode 1 and debug mode (OCDS Mode) are done in the startup code of the BootROM. The startup code will depend on the SCR configuration value in the SFR SCR_HWCFG, to enter each mode.

Table 13-4 shows the Boot Modes configuration value.

Table 13-4 TC26x B-Step **Boot Mode**

Boot Mode¹⁾	SCR CFG Value
User Mode 0, XRAM is not programmed	0 _H
User Mode 1, XRAM is programmed and will be executed	01 _H
OCDS Mode, SCR DAP interface	02 _H - 03 _H
OCDS Mode, SCR SPD interface	04 _H - 07 _H
OCDS Mode, SOC DAP interface	0A _H - 0B _H
OCDS Mode, SOC SPD interface	0C _H - 0F _H

1) SPD : Single Pin DAP; DPD: Dual Pin DAP

Feature Discontinued

13.4.2 OCDS Mode

This section describes the On Chip Debug System (OCDS) module and the interaction with the Monitor ROM and external debugger. The OCDS provides basic functionality to support development without the need to purchase expensive on-chip emulator (ICE) device which is the traditional way of debugging the embedded systems. The interface that can be used to support this function is DAP.

The TRAP_EN bit in Extended Operation register (EOA) needs to be enabled for OCDS functionality to work properly.

13.4.2.1 Features

List of features supported:

- Set breakpoints on instruction address and within a specified range
- Set breakpoints on internal RAM address for read and write
- Support unlimited software breakpoint in RAM code region
- Processing External breaks
- Stepping through the program code

In summary, up to four hardware breakpoints can be set for the current implementation. It can be used for detection of specific Instruction Pointer (IP), a range of IP and specific range of read/write of internal RAM, unlimited software breakpoint (only possible in writable region) and external break. When any of the above is encountered, that corresponds to a debug event which will then trigger the execution of Monitor ROM.

13.4.2.2 OCDS Basic Understanding

For a complete understanding of the OCDS concept, it is found in the User's Manual, OCDS chapter.

Introduction of the XC800 Core Debug mode

The Program Counter will freeze when the core enters the debug mode and it will enter this mode when the TRAP instruction is executed and with the TRAP_EN = '1'. It will also enter upon DebugReq activation for IRAM and External Breaks. For hardware breakpoint, the TRAP instruction is actually supplied by the Monitor Mode Control (MMC) hardware block.

Minimum hardware and overhead are added

OCDS hardware consists of the MMC, additional internal RAM and JTAG and additional ROM memory.

The OCDS is built on top of debug mode together with minimum hardware overhead, Monitor ROM and additional 64 bytes of RAM to perform the breakpoint debugging functionality.

Monitor is resident in the BootROM region and it will make use of the additional 64 bytes of internal RAM termed Monitor RAM and together with the hardware to realize the OCDS functionality. The Monitor RAM which occupied the same memory location as the standard 8051 internal RAM, and it is selected through the MFAMS bit (MMCR).

Implementation of OCDS with the debug mode of XC800 Core

Some combination logic together with the support of the Monitor ROM to coordinate the processes. In this document, the tasks of the Monitor ROM will be explained in details in the next section.

13.4.2.3 OCDS Monitor Firmware

This section will describe the Monitor firmware interaction between the Monitor ROM with the OCDS hardware and the debugger software in host PC.

OCDS mode entry

OCDS mode is entered if:

- c) OCDS Boot option for SCR DAP interface, SFR SCR_HWCFG = 02_H - 03_H;
 - d) OCDS Boot option for SCR SPD interface, SFR SCR_HWCFG = 04_H - 07_H;
 - e) OCDS Boot option for SOC DAP interface, SFR SCR_HWCFG = 0A_H - 0B_H;
 - OCDS Boot option for SOC SPD interface, SFR SCR_HWCFG = 0C_H - 0F_H;
- IP hardware breakpoint at XRAM address 0x0000 is set before it exits OCDS mode.

Feature Discontinued

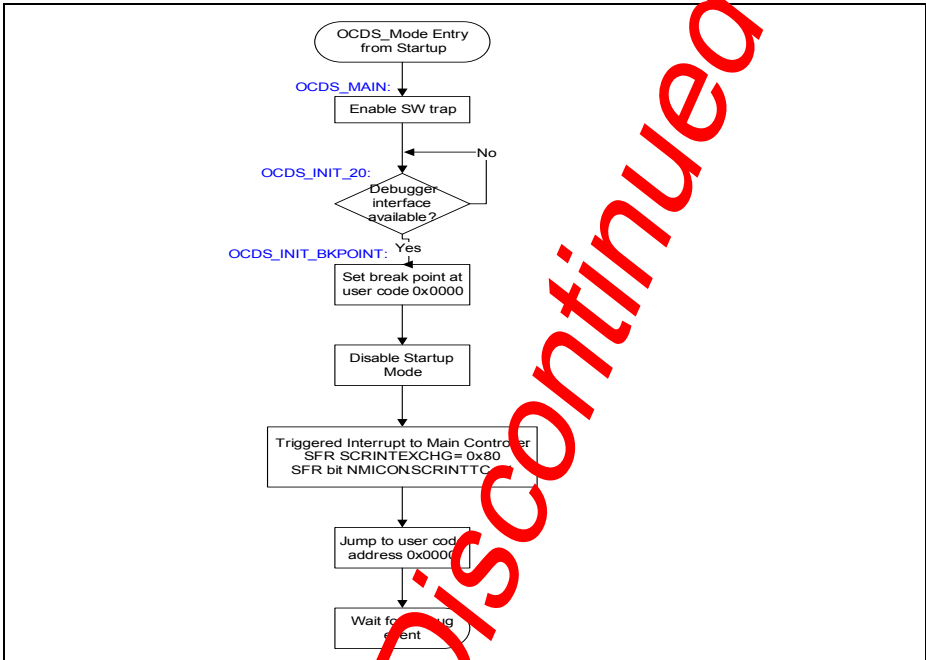


Figure 13-11 OCDS Initialisation

Communication between the debugger and Monitor

The debugger will be the master and it will control all the read and write transactions. A virtual memory map is defined as follows.

Virtual Address translation

To facilitate the debugger to get an insight of the memory map and all the registers in the system, a set of virtual addresses is defined to relate them to the physical address of the program memory, external memory and all of the other memories and registers within the system. In this way, the debugger will read or write to the virtual address and the actual reading and writing to the correct memory segment and register are controlled by the Monitor program.

Feature Discontinued

Table 13-5 Translation of virtual memory to physical memory

Description	Physical address	Valid Virtual Address for TC2x_SCR	Virtual Address (3.5 bytes)
Program memory	0000 _H - 1FFF _H	00000 _H - 01FFF _H (8K)	00000 _H - 0FFFF _H (64K)
External memory	0000 _H - 1FFF _H	10000 _H - 11FFF _H (8K)	10000 _H - 1FFFF _H (64K)
Internal memory	00 _H - FF _H	20000 _H - 200FF _H (256)	20000 _H - 2FFFF _H (64K)
Indirectly accessible OCDS register	HWBPxH/L 08 _H - 0F _H	22208 _H - 2220F _H (8)	22200 _H - 222FF _H (256)
Monitor Virtual Registers	22300 _H - 2233F _H	22300 _H - 2233F _H (up to 64 bytes but not all contain the valid data)	22300 _H - 223FF _H
Reserved			30000 _H - 3FFFF _H
SFR registers and page number. (Including standard, mapped and paged)	80 _H -FF _H As Bit7 is always set, thus only 7 bit is required to identify the desired SFR.	40000 _H - 7FFFF _H (always BIT18=1) BIT 17:15 Page number BIT 14:8 SFR page address BIT7: SFR standard (1), mapped (0) BIT6:0 SFR address	40000 _H - 7FFFF _H

For the XRAM program memory, the virtual translated address is the same as their address. For external memory and internal IRAM, the upper nibble of the virtual address is 1_H and 2_H respectively.

There are 8 indirect OCDS registers. They are the OCDS hardware breakpoint (HWBPxH/L). These registers are only accessible by setting HWBPSR first to select the relevant registers. To simplify the debugger task of setting breakpoint, the registers are mapped to the virtual address. The monitor program will base on the virtual address and write to these registers appropriately.

Feature Discontinued

Table 13-6 Indirectly accessible OCDS register

Register name	Virtual translated Address	Description
HWBP0L	22208 _H	Set the address for breakpoint 0
HWBP0H	22209 _H	
HWBP1L	2220A _H	Set the address for breakpoint 1
HWBP1H	2220B _H	
HWBP2L	2220C _H	Set the address for breakpoint 2
HWBP2H	2220D _H	
HWBP3L	2220E _H	Set the address for breakpoint 3
HWBP3H	2220F _H	

In order to optimise the amount of virtual address allocated to the SFR. A scheme has been devised to cope with all the SFR type i.e. standard, mapped, paged type. In order to access all the SFR, the following fields are required and defined below.

Table 13-7 Distribution of the bits definition

Field	Bit definition	Description
RMAP	BIT7	To indicate which region of SFR to access. '0' - standard and '1' - mapped
SFR_ADDRESS	BIT 6:0	SFR to be accessed. The value of the SFR is always 8 _H or above. Thus it is represented by 7bit omitting the most significant bit.
SFR_PAGE_ADDRESS	BIT14:8	For accessing paged SFR, there is a need to write the page number to the SFR PAGE address first. Once again, it is represented by 7 bit. For non-paged SFR, this field must be written with 00 _H .
PAGE_NUMBER	BIT17:15	For non-paged SFR, the page_number must be set to 0F _H . For paged SFR, this field can be used to specify page number from 0 to 07 _H .

The standard SFR refer to those SFR that required SYSCON0.RMAP=0 before they can be accessed and the mapped SFR refer to those SFR that can be accessed only when SYSCON0.RMAP=1. As for paged SFR, in addition to the standard or mapped properties, they also required the page number to be set, and it is written to the corresponding SFR page address. If the SFR does not paging, then SFR page address field must set to zero.

Bit17-Bit15 will represent the page number from 0x00-0x07, if the value equal to 0x07 and it is a non-paged SFR, then it is not necessary to set the SFR page address and BIT14-8 can be clear to zero. Otherwise, Bit14-8 will indicate the address (7bit only) for that SFR page. Bit7 = 0 will represent standard and 1 if mapped SFR. Bit6-0 will indicate the address (7bit only) for that desired SFR to be read out. The SFR is represented by 7 bit as its address is always above 80H and the MSB is not required and automatically handled by the firmware.

The list of registers that are used in the monitor program need to be preserved, thus to access these few SFR, the corresponding virtual address should be supplied instead of following the format given in [Table 13-7](#)

Table 13-8 Monitor virtual register

Monitor virtual register name	Virtual translated Address	Description
mons_SCU_PAGE	2231A _H	Current SCU_PAGE register
mons_IEN0	2231B _H	Current Interrupt Enabled register
mons_DPL	2231C _H	Current data pointer low byte
mons_DPH	2231D _H	Current data pointer high byte
mons_B	2231E _H	Current B register
mons_A	2231F _H	Current A register
mons_SYSCON0	22330 _H	Current SYSCON0 register
mons_PSW	22332 _H	Program Status Word
mons_RETH	22333 _H	Return address high byte
mons_RETL	22334 _H	Return address low byte
mons_SP	22335 _H	Current Stack Pointer

Protocol definition between the debugger and Monitor ROM

There are 5 Monitor processed commands that are defined and describe as followed.

Table 13-9 Monitor processed commands

Command	Value	Description
READ_BYTE	1X _H	Read a byte.
READ_BLOCK	2X _H	Read a block of data and return byte by byte to the host.
WRITE_BYTE	3X _H	Write a byte to memory location.

Table 13-9 Monitor processed commands

Command	Value	Description
WRITE_BLOCK	4X _H	Write a block of data starting from memory location.
WRITE_PROTECTED_SFR	5X _H	Write a byte to the password protected bit SFR

Note: In the command field, only the upper nibble will indicate the command type, the lower nibble is used to store the 4 MSB bit of the virtual address and is not part of the command.

Some assumptions made:

- Instead of 12 bit, it is round up to 16 bit for easy handling. Thus some are dummy bits that are not used but need to be there for data padding purpose to ensure the proper operation.
- Debugger send commands and data to Monitor, the Monitor ROM will acknowledge all bytes that are the command, parameters and data field that have been received. This is to ensure all the bytes from the debugger is received correctly.
- All commands (read or write) will be issued with a unique command_ID, since there are only five commands defined,.
- A new command 50_H is introduced to support the password protected bit SFR write operation.

Feature Discontinued

13.4.2.4 Monitor processed commands

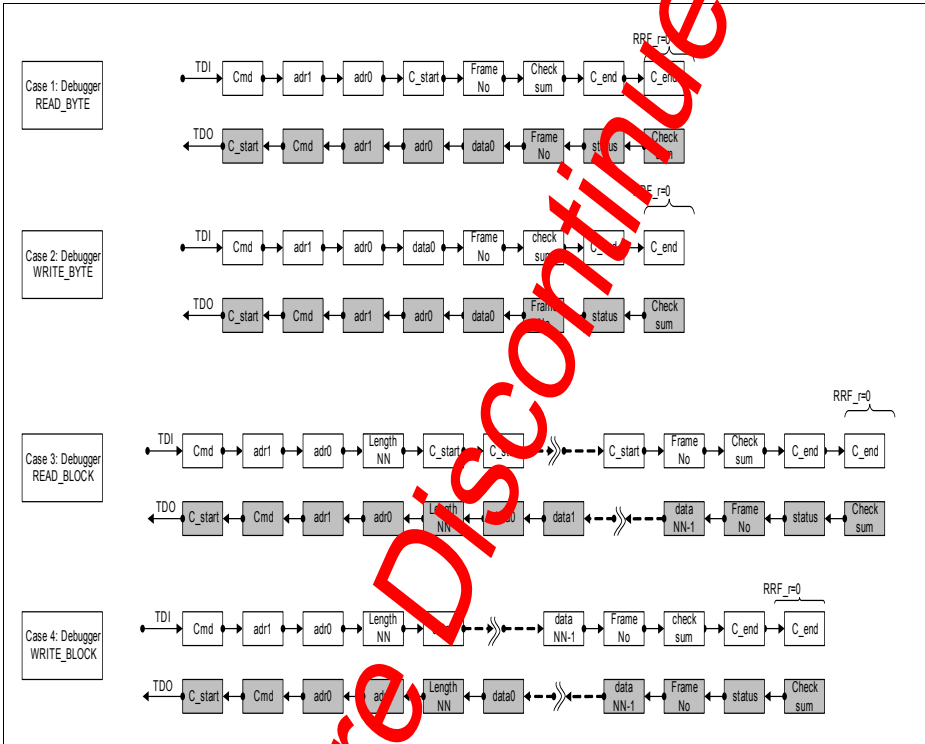


Figure 13-12 Monitor processed commands

The following is a list of the Monitor processed commands that are supported by the Monitor firmware.

Monitor processed command: READ_BYTE (1X_r)

The debugger will send 1 byte of command and 2 bytes of address. Next C_START, Frame No and Check sum are sent. Lastly followed by the C_end and C_end with RRF_r = 0. The corresponding data and the status byte are then checked by the debugger.

Monitor processed command: READ_BLOCK (2X_H)

The debugger will send 1 byte of command, 2 bytes of address and 1 byte of length. Next the C_START is send one after other until all the data are being transferred. Next the Frame No and Checksum are send followed by the C_end and C_end with RRF_r=0.

Monitor processed command: WRITE_BYTE (3X_H)

This command flow is very similar to the READ_BYTE command except that the C_START is not send out by debugger. Instead the data byte is written there.

Monitor processed command: WRITE_BLOCK (4X_H)

This command flow is similar to the READ_BLOCK except the C_START is not send out by the debugger. Instead all the data fields are send. Next the Frame No and checksum are also being sent out followed by the C_end and C_end with RRF_r=0.

Refer to the flow chart [Figure 13-12](#) that describe the commands flow.

Monitor processed command: WRITE_PROTECTED_SFR (5X_H)

This command is added to support writing to the password protected SFR and its functionality is similar to the WRITE_BYTE.

Note: For all the 5 commands being described, the upper 4 bit MSB of the address field is embedded inside the lower nibble of command byte. The Frame No is a 1 byte field wide, it just a sequence number that increment itself and warp around when it is overflows.

Description of the various fields use in the protocol

Various fields are define to establish the communication link between the debugger and Monitor ROM. They are outlined in [Table 13-10](#).

Table 13-10 Different fields that are defined

Description (byte)	Value	Description
C_START (1)	01 _H	To indicate start or ongoing event
C_END (1)	02 _H	To indicate ending event
FRAMENO (1)		The debugger will send a 00 _H first and sequentially increase it by 1 and wraps around if the value exceeds the single byte length. This value must reset and start from 00 _H when Monitor Mode is (re-)entered or when COMRST has been activated.
CHECKSUM (1)		The checksum for the debugger and Monitor side are calculated by XORING all the fields except its checksum.

Table 13-10 Different fields that are defined (cont'd)

Description (byte)	Value	Description
STATUS(1)		The following bit will set to indicate an error condition. BIT0 - Debugger CHECKSUM and the calculated value does not match. Read/Write operation to the memory is executed. BIT1 - Frame number out of sequence. Read/Write operation to the memory is executed. BIT2 - Control bytes, C_START not received correctly. Read/Write operation to the memory is executed. BIT3 - Invalid Range of the memory location. Read/Write operation to the memory is not executed.
ADR0 (1)		This represent the least significant 8 bit of the virtual address.
ADR1 (1)		This represent the next higher 8 bit of the virtual address.
CMD (1)		The upper nibble contains the command as specified in Figure 13-12 . The other nibble represent the most significant 4 bit of the virtual address. Thus together, they made up the 20 bits address.
Length (1)		This field is used in the read block command and write block command. It indicates the number of data bytes to be read or written into the memory. Valid range of this field is from 1 to 255.

JTAG communication and controlling the OCDS hardware control bits

[Table 13-11](#) list the monitor hardware related commands.

Table 13-11 Monitor hardware related command

Mode/Command	Data	Jena	RF	MMODE	MBCON	COMRST	Description
		_r	_r	_r	_r	_r	
HSTATE	XX _H	0	0	0	0	0	Get the status of the OCDS status bits.
HCONFIG	A5 _H	1	0	0	0	0	Debug interface enabled by MMCR2.JENA = 1
HDEBUG	DATA	0	1	0	0	0	Data read and write
HMONITOR	A5 _H	0	0	1	0	0	Enter the Monitor mode

Table 13-11 Monitor hardware related command (cont'd)

Mode/Command	Data	Jena_r	RRF_r	MMODE_r	MBCON_r	COMRST_r	Description
HMBCOUT	XX _H	1	0	0	X	0	Jena_r must be set together with the MBCON_r state to control the MBC output. To activate it, MBCON_r=1 and to disable it MBCON_r=0.
HCOMRST	A5 _H	0	0	0	0		Reset the transaction and start with a new command.

The prefix with H indicate that it is a OCDS hardware related commands.

The HDEBUG command/mode is use for any data read and write transaction and it is actually being used to support the five Monitor processed commands.

Activate the Monitor ROM

When a debug event occurred, the debug mode will be entered and instruction will be fetched from MM_START that represent the upper 8 bit of the program code address bus. MMCR.MMODE is set by the hardware to signal that the Monitor mode is entered.

The following need to be taken care (Extract from OCDS specification)

- Save user Interrupt Enable (IE₀) and disable the global interrupt.
- Save current (user) Stack Pointer (SP) into Monitor RAM.
- Initiate the stack to dedicate area in Monitor RAM
- Save the SYSCON0 into the Monitor RAM, to preserve the user value of RMAP control bit.
- Clear the RMAP and IMODE bit in SYSCON0.
- Set the SCU_PAGE to page 0 and save the current NMICON into the Monitor RAM and disable all sources of NMI by writing zeros to NMICON.
- Save the current (user) I-SW, A, B, DPH and DPL to Monitor IRAM.
- Set RMAP bit in the SYSCON0, to access MMICR.
- Clear the MMCR.DNTR
- Perform a LOCAL to MON_START_ST (or MON_NORMAL_ST), so the break address is saved and debug mode is exited and continue to run in Monitor ROM mode.
- NOP instruction is required here for proper functioning.

- Communication between the debugger and Monitor through the selected interface.

Figure 13-13 shows the flow diagram when the debug event occurred. Debugging a BootROM startup code will cause the device to enter endless loop. A reset of the device is needed to exit the endless loop.

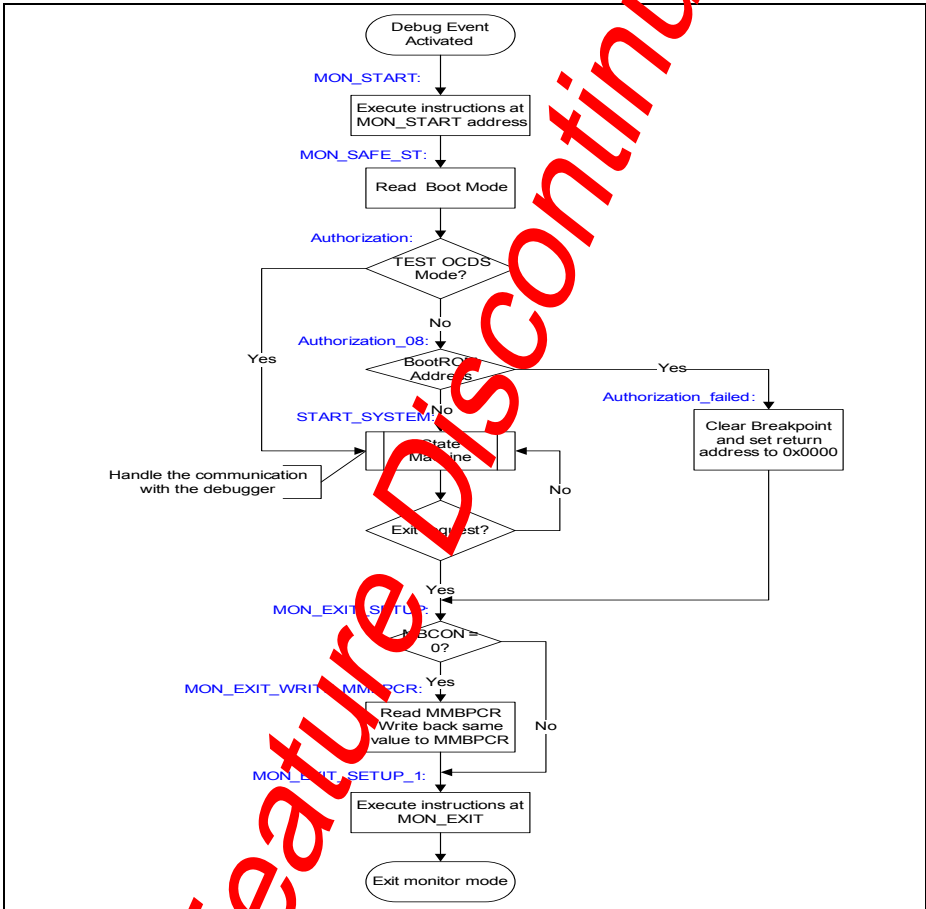


Figure 13-13 Debug mode Flow Diagram

Figure 13-14 shows the state-machine flow diagram in the monitor program. The detailed descriptions and flow diagrams of each state can be found in Chapter .

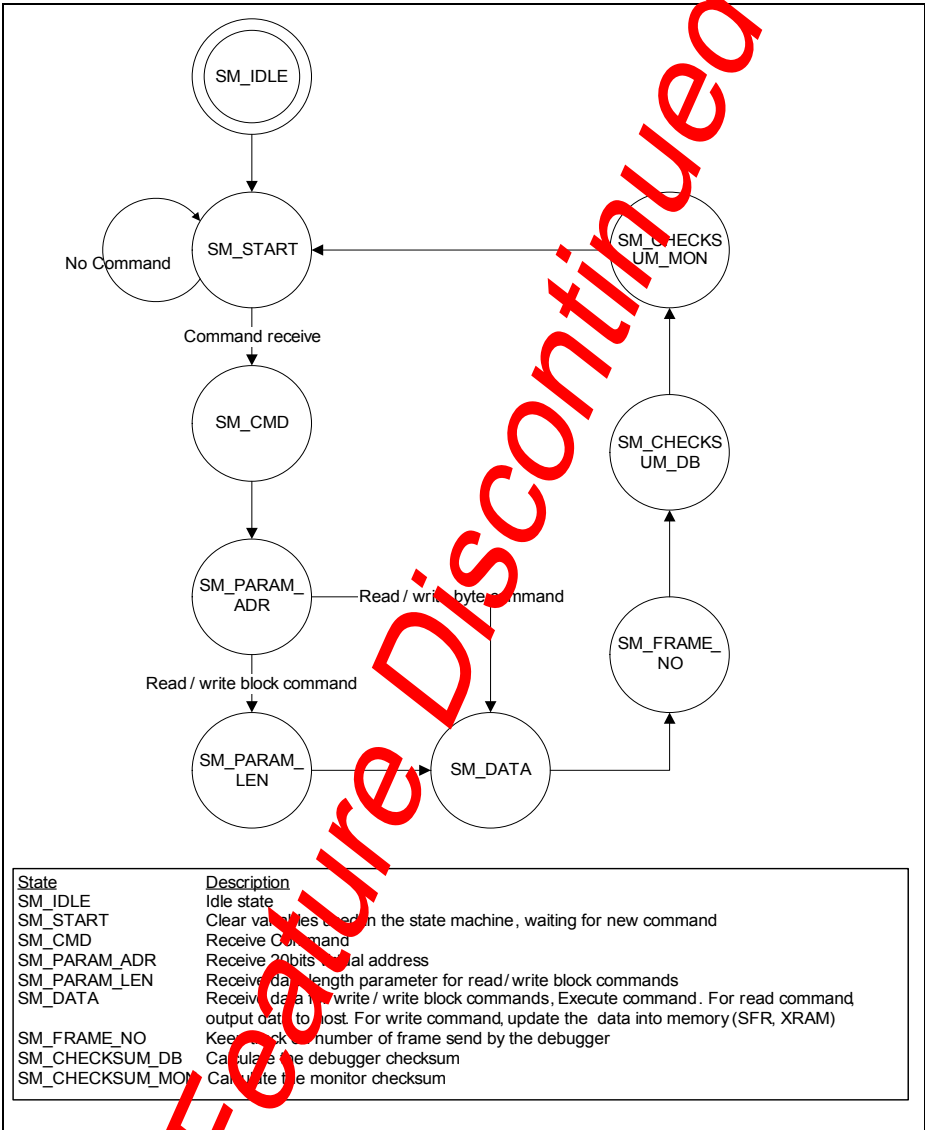


Figure 13-14 State Machine - To handle monitor processed commands from host

Exit the Monitor ROM to return to user mode

To exit monitor mode and return to the user code, the following steps are needed to restore the user status and return to the proper user address.

The steps needed to return to user mode.

- After writing to the MMCR.EXIT=1 followed by NOP, this will trigger the debug mode to be entered.
- Mons_SYSCON0 and mons_RETH are temporary stored into the 2 OCDS scratch registers.
- SYSCON.RMAP is cleared and set the SCU_PAGE=0 to restore back the saved mons_SCU_PAGE.
- IEN0,DPH,DPL,B,A,PSW and SP are restored back respectively from the Monitor IRAM.
- The return addresses which are contained in the mons_RETL and MMWR2 are pushed to the stack.
- The SYSCON0 is updated with the value stored in the MMWR1
- RET is performed, i.e jump to address that are stored in the stack which transfer the control back to the user program.
- NOP is followed to ensure proper operation.

Single step execution

Single step execution:

- Current instruction in the user mode is executed and a new debug event will automatically be triggered and the monitor mode is entered again.
- The return address is calculated by minus 1 of the break address.
- In the event when other break event happened due to single step or others, then step will always take the higher priority and consider first for the calculation of the return address.

Software breakpoint implementation

It is only possible if the original code can be replaced by the TRAP opcode. And this implies that the code memory must be writable. Thus for the XRAM, the XRAM content must be saved by the debugger and modified with the TRAP opcode A5_H. After the break has occurred, the debugger should program back the original user code, so that the normal flow of the program is not affected.

13.4.2.5 Important Information for Debugger

This section will describe the additional tasks that are required by the debugger to produce seamless integration and synchronization flow between the monitor firmware. The debugger is the master for all transaction and it must keep track of the flow and

always in sync with the current MCU resources changed. The OCDS capabilities will be disabled for BootROM region.

Specific Infineon DAP instruction to configure DUT

The debugger on the host side act as the master and is responsible for initiating and controlling all JTAG transactions.

From the BootROM perspective, it is only concern with the JTAG communication between the JTAG module and the OCDS module. The communication can begin once the JTAG Instruction Register (IR) is set with **JTAG_IO_INSTRUCTION1** ($C1_H$). This will effectively connect the external JTAG pins to the OCDS/JTAG Interface module thus enabling communication between the debugger and the OCDS module. Note `JTAG_IO_SELECT_PATH(C0_H)` command is not needed since OCDS is selected by default after reset (`IOPATH = 0`).

Initial entry to the monitor mode

There is only one possible initial entry into the monitor mode.

Upon a reset, with boot mode configured for OCDS by the SOC, it will enter OCDS mode. The IP break point is configured at address 0000_H and execution begin at that specific address and a break event is generated and monitor mode will be entered.

Implementation of the debugger

Upon the entry of the monitor, the PC value are saved into the `mons_RETH` and `mons_RETL`.

The 16 bytes from $1FE8_H$ to $1FF_H$ of the XRAM is been used by the Monitor ROM to perform reading and writing the SFR. Thus the contents at these location must be saved by the debugger when the Monitor mode is entered and restored them before exiting the Monitor mode.

IP and IRAM breakpoints can be support and are set by modified by writing to the OCDS registers. Likewise, the debugger can view and modified any SFR according based on the virtual address translation.

For every transaction, a status byte will be returned. A non-zero value will indicate that there is error, refer to the table for the STATUS byte field definition [Table 13-10](#).

Calculation of the return address upon a break event

The break address is stored in `mons_RETH` (32_H) and `mons_RETL` (33_H) upon a break and enter the monitor mode. Thus the debugger must check what type of break has occurred and calculate the return address accordingly.

For IP and IRAM breakpoint, the return address = **break address - 2**. For any other break, the return address = **break address -1**.

The debugger will need to update the return address into the `mons_RETH` and `mons_RETL` before exiting the monitor mode. To implement the "Go address" function, the debugger should modify the `mons_RETH` and `mons_RETL` directly before exiting.

Limitation of the monitor mode

Interrupt handling, if an external interrupt has occurred, the monitor will not be able to service it. For IRAM read and write break, only the return address can be calculated but the break address might not be known.

Exit sequence of the monitor mode

To exit the Monitor mode, the debugger will need to write to the `MMCR.MEXIT = 1` together with the `MMCR.MEXIT_P = 1`. Upon writing to these bits, the Monitor mode is automatically exited. Prior to this, the starting 16 bytes from (`1FE8H - 1FF7H`) of the XRAM must be restored.

An example of a Debugging session

Here is an example of a debugging session.

- Initial entry of the monitor mode through JTAG mode.
- The PC is saved to the stack by the monitor firmware. The debugger will buffer the 16bytes of XRAM at address `1FE8H - 1FF7H`, into its memories.
- Load a user hex file into the XRAM memory.
- Set the IP and IRAM breakpoints.
- Modify the `mons_RETH` and `mons_RETL` to the starting address for execution.
- Clear all the breakpoint flag.
- Follow the exit sequence.
- The break will occur and once again the monitor mode is entered and PC is saved.
- Debugger will once again buffer the 16 bytes of data into the its memories.
- All the memories and SFR can be read and modified.
- The return address is calculated based on the break address and the breakpoint flags.
- The return address is updated.
- Restore back the 16bytes of XRAM content back to its original location.
- Follow the exit sequence.
- This process will repeat as long as there is a break event.

Detailed description of Monitor Program State Machine

This session described the flow of the major states:

- Start state - start of the state machine

- Command state - Received OCDS command and the upper 4 bits of the virtual address.
- Parameter Address state - Received the lower 16-bits virtual address.
- Parameter Data Length state - Received the frame Length
- Data state - For read command, this state will perform the read commands and send out the data. For write command, this state will receive the data and perform the write operation to the respective memory.

Start and Command States

Figure 13-15 shows the flow of start state and command state. The lower nibble of the command byte contains the uppermost 4 bits of the virtual address.

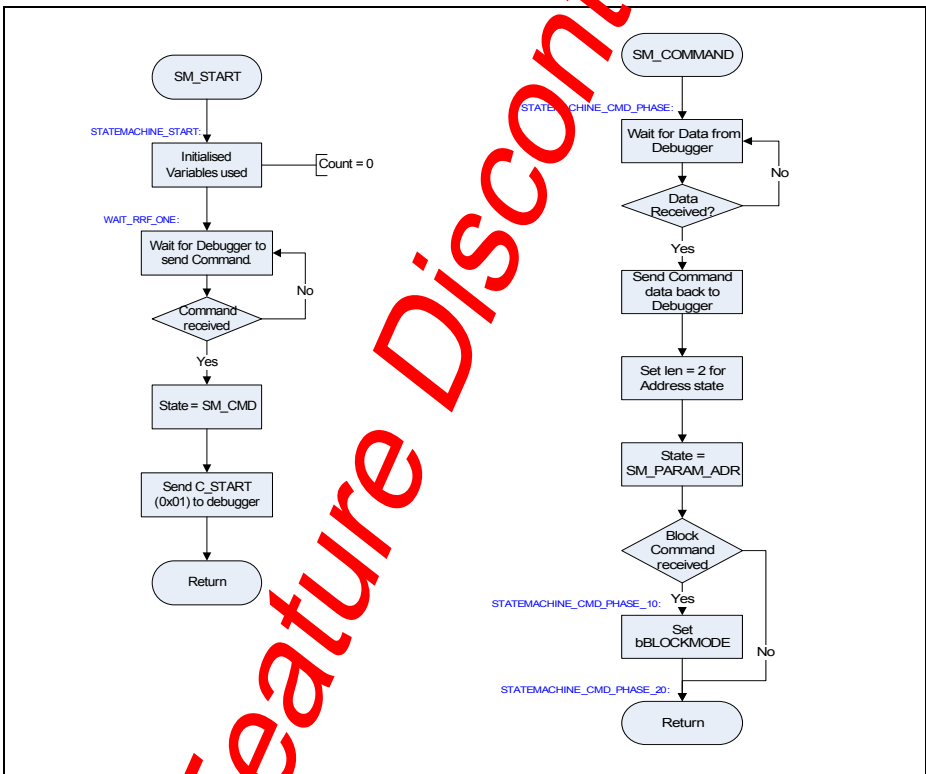


Figure 13-15 Start State and Command State

Parameter Address State

In this state, the next 16 bits of the virtual address are received. The 20-bits virtual address indicates which memory or SFR is to be accessed.

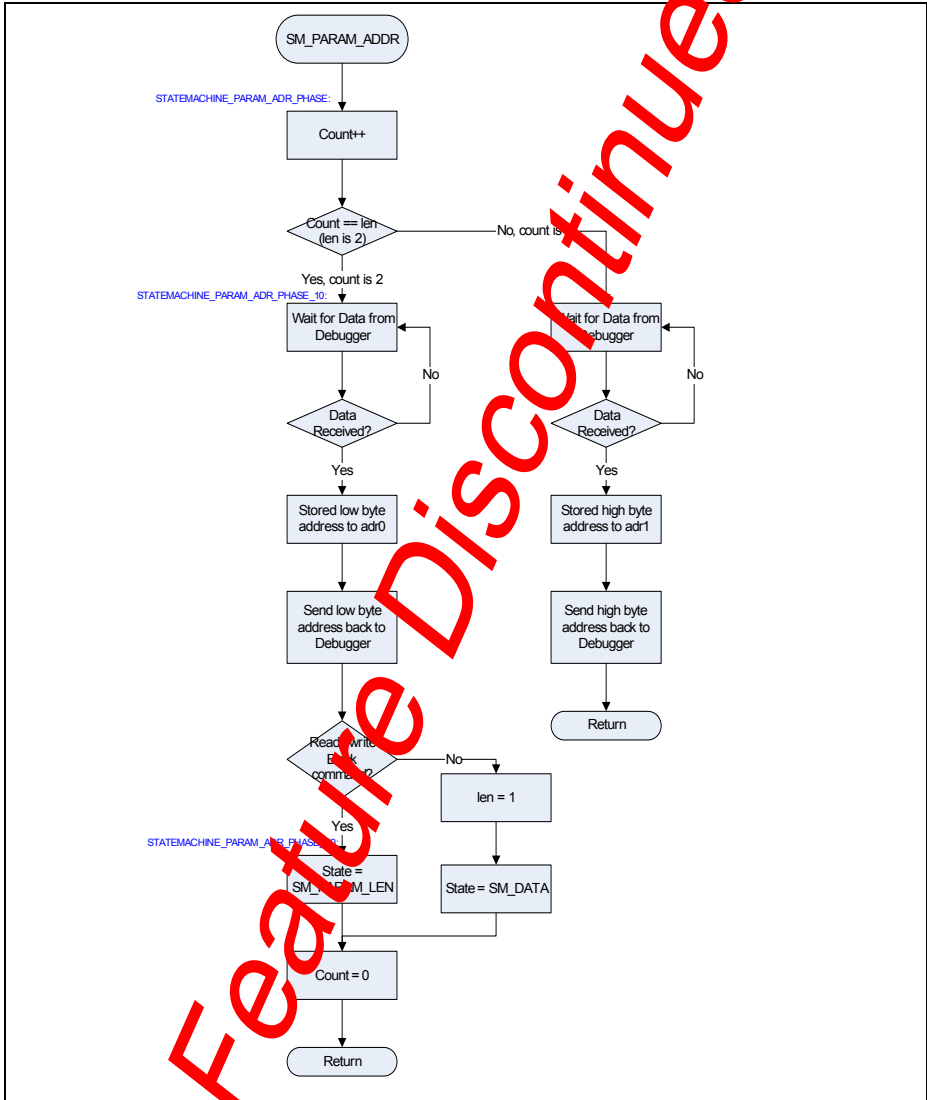


Figure 13-16 Parameter Address State

Parameter Length State

This state is entered if the command is read block command or write block command. The data received indicates the number of data bytes to be received or sent.

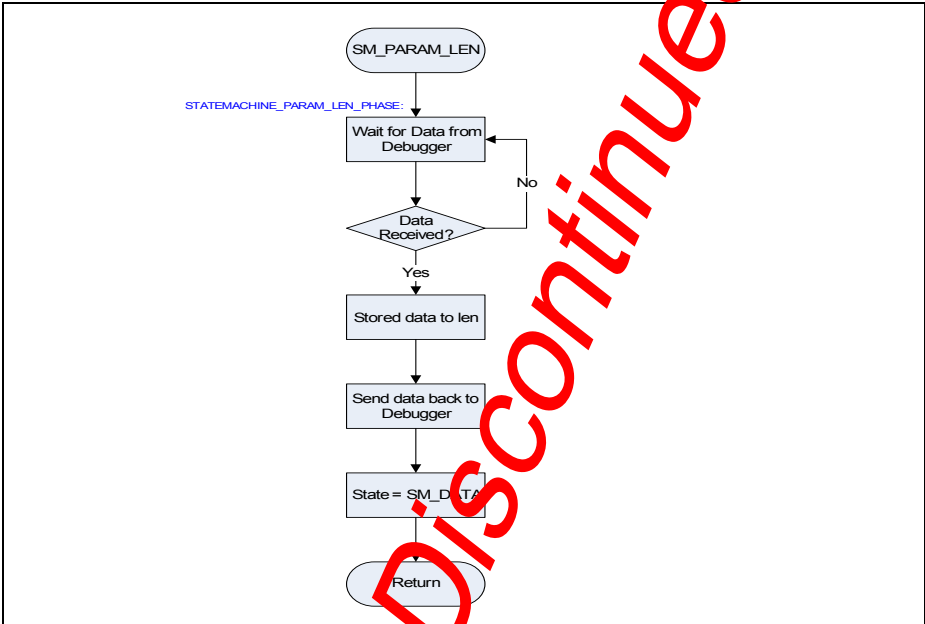


Figure 13-17 Parameter Length State

Data State

For Read_BYTE and READ_BLOCK commands, data is read from the respective memory or SFR and send to the debugger. For the other 3 commands, data is received from the debugger and written to the memory or SFR.

Feature Discontinued

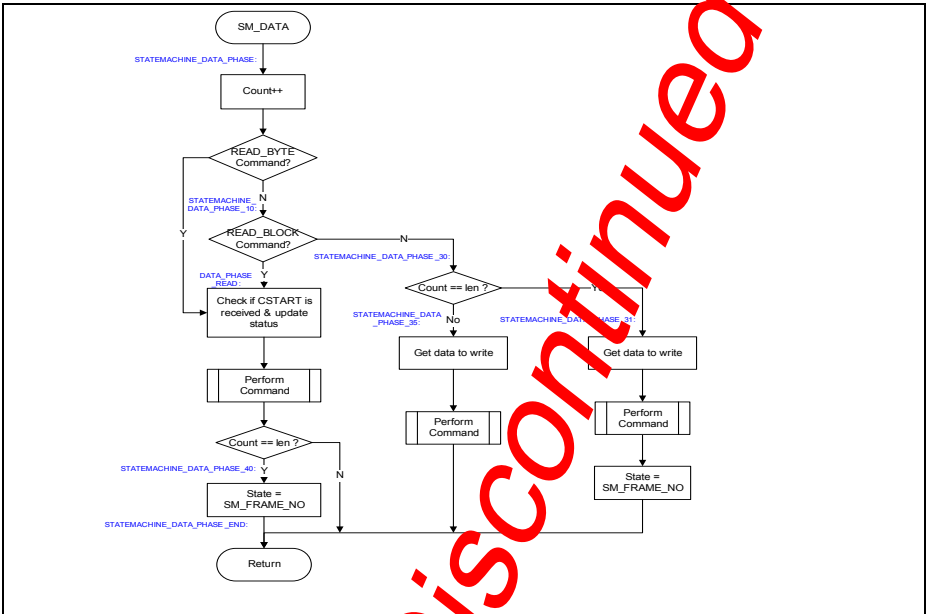


Figure 13-18 Data State

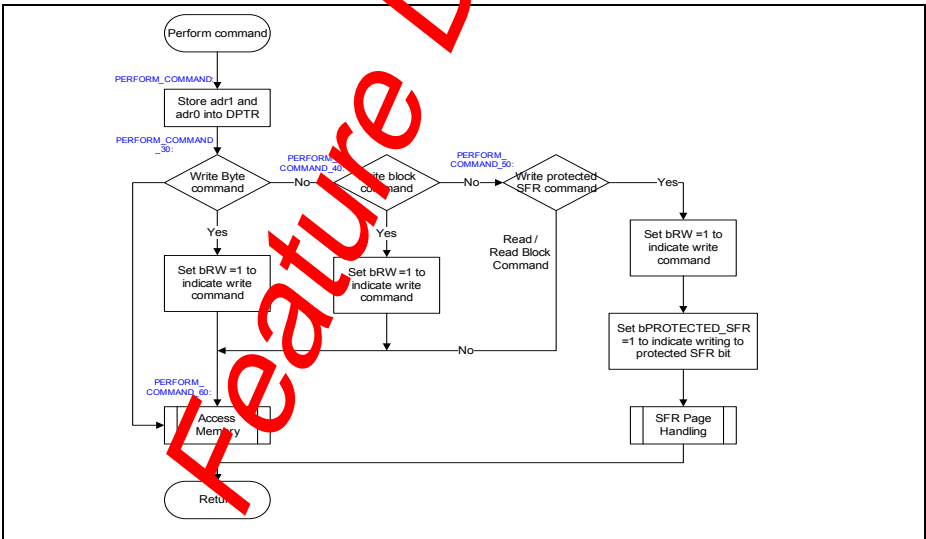


Figure 13-19 Perform Command

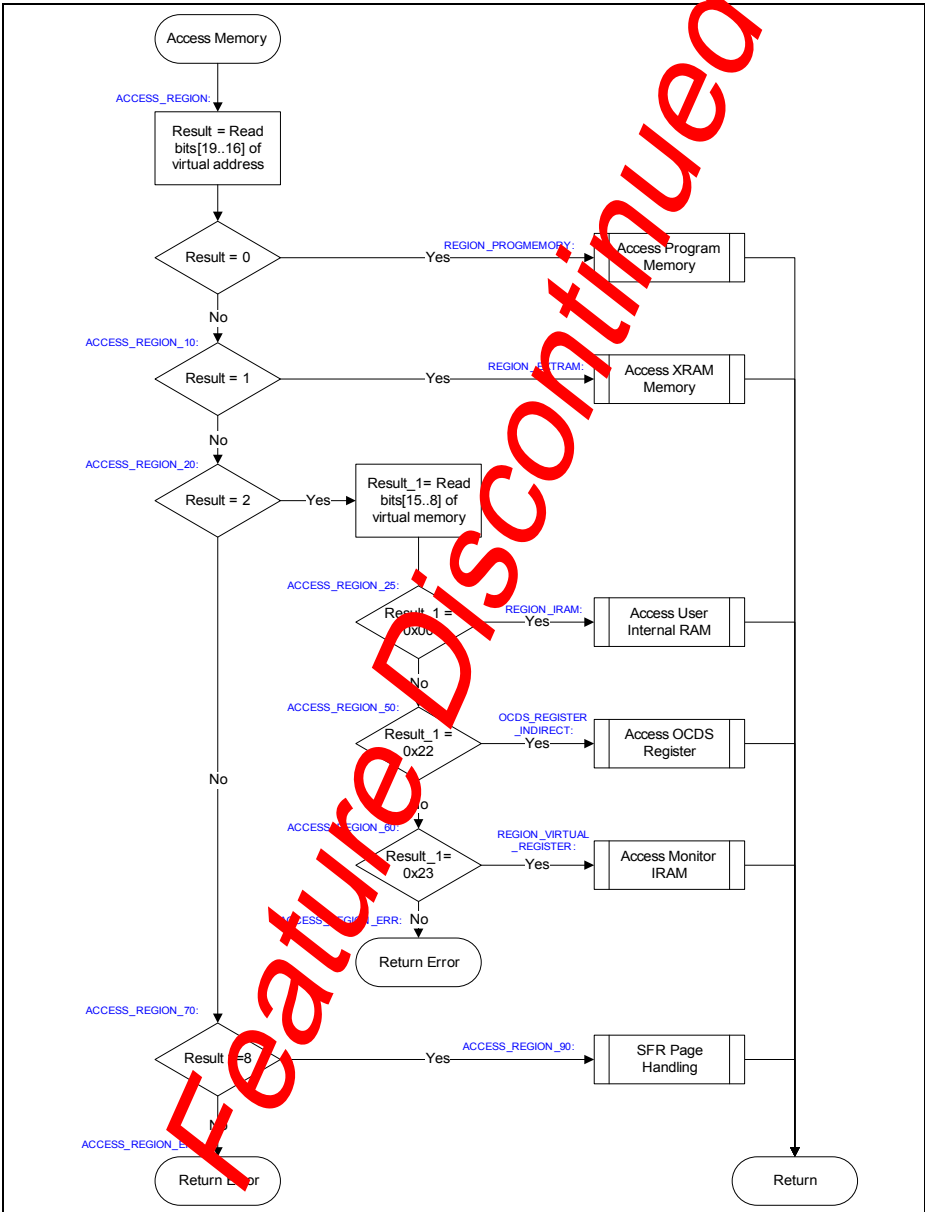


Figure 13-20 Access Virtual Memory

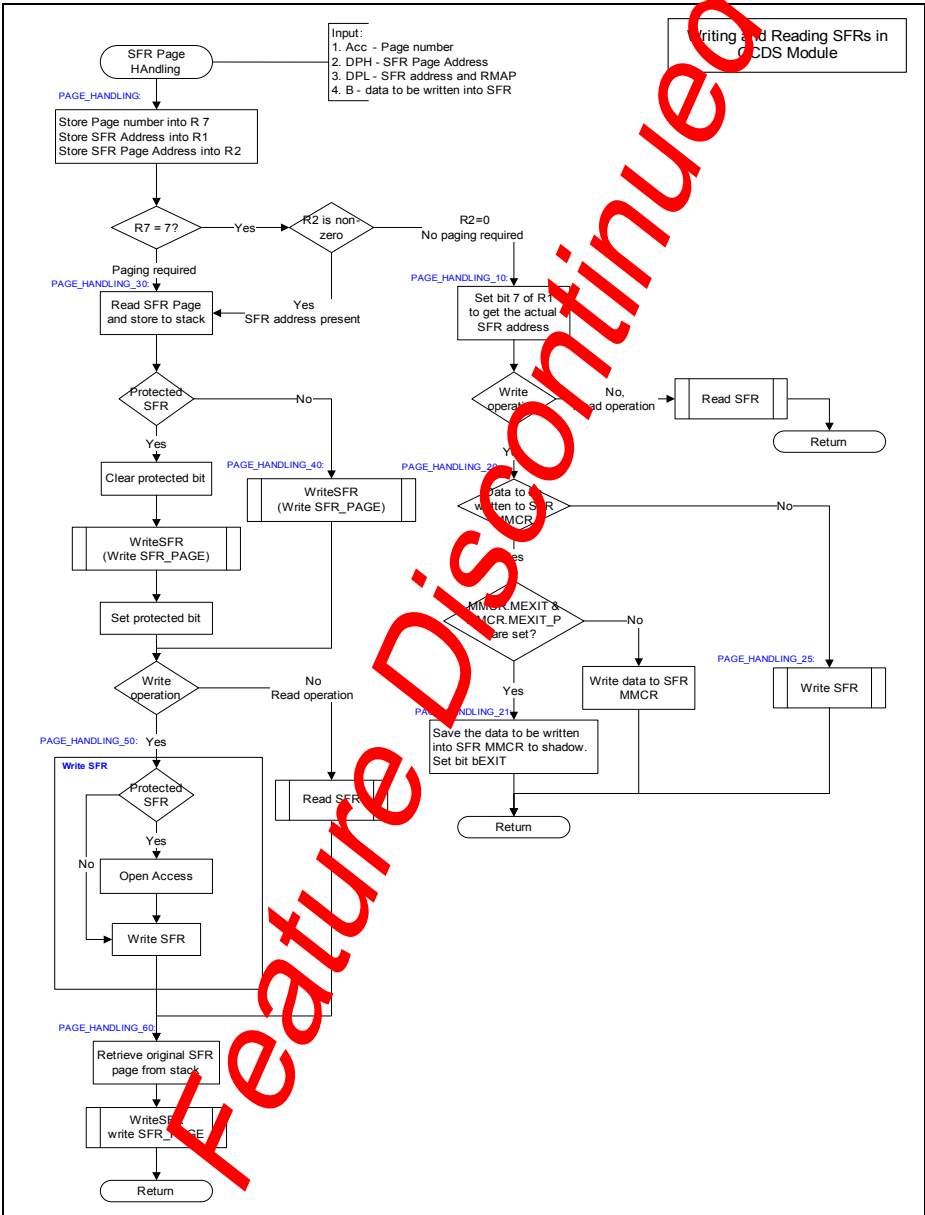


Figure 13-21 Access SFR

13.5 System Control Unit

The System Control Unit (SCU) supports all central control tasks in the TC2x_SCR. The SCU is made up of the following sub-modules:

- Clock System and Control (see [Section 13.5.1 on Page -54](#))
- Reset Control (see [Section 13.5.2 on Page -63](#))
- Interrupt Management (see [Section 13.5.3 on Page -70](#))
- Boot and Startup (see [Section 13.5.4 on Page -76](#))
- Power Management (see [Section 13.5.5 on Page -79](#))
- Flexible Peripheral Management (see [Section 13.5.6 on Page -84](#))
- XRAM Addressing Modes (see [Section 13.5.7 on Page -85](#))
- Module Suspend Control (see [Section 13.5.8 on Page -87](#))
- JTAD ID (see [Section 13.5.9 on Page -87](#))
- General Port Control (see [Section 13.5.10 on Page -88](#))
- Miscellaneous Control (see [Section 13.5.11 on Page -95](#))
- Register Mapping (see [Section 13.5.12 on Page 99](#))

13.5.1 Clock System and Control

Figure 13-22 shows the block diagram of the clock system in TC2x_SCR. It consists of two internal oscillators and a clock control unit (CCU). The two oscillators, a 100 MHz oscillator and a 100 kHz oscillator, are supplied by the internal EVR. These two clock sources to TC2x_SCR are gated with bit PMSWCR0.SCREEN in both normal and standby mode. If this bit is set to "0", TC2x_SCR is disabled and the clock inputs would be gated off. The system clock f_{SYS} is generated either by the 100 MHz or 100 kHz internal oscillator.

Upon any reset, 100 kHz oscillator is the default clock source. It is also the only clock source that is active in both normal and standby mode. In standby mode, the 100 MHz oscillator may be powered down as configured in PMSWCR0.SCRCLKSEL. When necessary, the 100 MHz oscillator can be powered up using bit CMCON.OSCPD. An oscillator watchdog circuit is needed to detect a stable 100 MHz oscillation before selecting the 100 MHz clock source. OSC2L bit in register CMCON is used to indicate the status of the 100 MHz clock.

When 100 MHz oscillator is selected as the clock source for the generation of the system frequency, the oscillator watchdog is also used to detect a 100 MHz clock failure and trigger an NMI interrupt if it is enabled by setting bit NMICON.NMIOSCCLK. When the 100 MHz oscillator falls below the threshold of 70 MHz, the clock source will be automatically switched to the 100 kHz oscillator.

Note: If 100 MHz (DI) clock is selected for WDT and/or RTC and 100 MHz oscillator fails, WDT and/or RTC clock will stop operation and NMI interrupt will be triggered. User can switch to the 100 kHz clock, and reinitialize WDT and RTC to run at 100 kHz.

The CCU generates all necessary clock signals within the microcontroller from the system clock. It consists of:

- Clock source selection in active mode
- Centralized enable/disable circuit for clock control

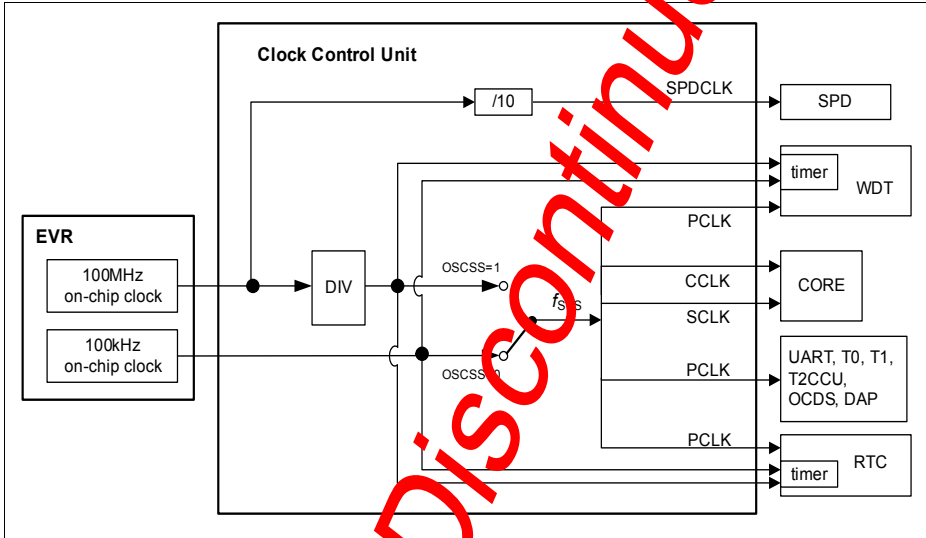


Figure 13-22 Clock System Block Diagram

In active mode, the main module frequencies (synchronous unless otherwise stated) are as follows:

- CPU clock (CCLK, SCLK) = 100 kHz (default) or 100 MHz / DIV
- Peripheral clock (PCLK) = CPU clock

When 100 MHz clock source is selected (CMCON.OSCSS = 1), the clock to the CPU and peripherals will be divided by the programmable factor DIV that is selectable by the bit field CMCON.DIV.

Peripherals that are running in PCLK frequency are UART, T0, T1, T2CCU.

Note: SPD Debug interface can only be supported when the 100 MHz oscillator is available.

Note: WDT and RTC clock running faster than PCLK frequency is not supported.

13.5.1.1 CCU Register

CMCON

Clock Control Register

 (F4_H)

 Reset Value: 00_H

RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
0	OSC2L	OSCPD	OSCSS			DIV	
r	rh	rwh	rw			rw	

Field	Bits	Type	Description
DIV	[3:0]	rw	Clock Divider 0000 _B <i>DIV</i> = 5; $f_{SYS} = 20$ MHz 0001 _B <i>DIV</i> = 8; $f_{SYS} = 12.5$ MHz 0010 _B <i>DIV</i> = 10; $f_{SYS} = 10$ MHz 0011 _B <i>DIV</i> = 16; $f_{SYS} = 6.25$ MHz 0100 _B <i>DIV</i> = 20; $f_{SYS} = 5$ MHz 0101 _B <i>DIV</i> = 32; $f_{SYS} = 3.125$ MHz 0110 _B <i>DIV</i> = 40; $f_{SYS} = 2.5$ MHz 0111 _B <i>DIV</i> = 50; $f_{SYS} = 2$ MHz 1000 _B <i>DIV</i> = 80; $f_{SYS} = 1.25$ MHz 1001 _B <i>DIV</i> = 100; $f_{SYS} = 1$ MHz 1010 _B <i>DIV</i> = 125; $f_{SYS} = 0.8$ MHz 1011 _B <i>DIV</i> = 160; $f_{SYS} = 0.625$ MHz 1100 _B <i>DIV</i> = 200; $f_{SYS} = 0.5$ MHz 1101 _B <i>DIV</i> = 250; $f_{SYS} = 0.4$ MHz 1110 _B Reserved 1111 _B Reserved
OSCSS	4	rw	Oscillator Source Select 0 _B 100 kHz on-chip oscillator is selected. $f_{SYS} = 100$ kHz 1 _B 100 MHz on-chip oscillator is selected. $f_{SYS} = 100/DIV$ MHz <i>Note: User need to ensure that OSCPД must be set to 0 and OSC2L must be 0 before selecting 100 MHz oscillator as the system clock. A 100 MHz clock failure NMI may be triggered if these conditions are not met.</i>

Field	Bits	Type	Description
OSCPD	5	rwh	<p>100 MHz Oscillator Power Down Control in Standby Mode</p> <p>This bit is used to power down the 100 MHz oscillator in standby mode.</p> <p>0_B 100 MHz oscillator is not powered down in standby mode.</p> <p>1_B 100 MHz oscillator is powered down in standby mode.</p> <p>This bit is automatically cleared by hardware. The OSCPД bit is a protected bit. When the Protection Scheme is activated, this bit cannot be written directly. For more information on Protection Scheme, see Section 13.5.11.</p> <p><i>Note: The status of this bit will be updated upon the entry to standby mode based on the setting of PMSWCR0.SCRCLKSEL. User can program OSCPД bit to power down the 100 MHz oscillator in standby mode only if PMSWCR0.SCRCLKSEL is set to 0.</i></p>
OSC2L	6	rh	<p>100 MHz Oscillator Too Low Flag</p> <p>0_B 100 MHz clock oscillates above threshold of 70 MHz.</p> <p>1_B 100 MHz clock oscillates below threshold of 70 MHz.</p>
0	7	r	<p>Reserved</p> <p>Returns 0 if read; should be written with 0.</p>

Feature Discontinued

13.5.2 Reset Control

The TC2x_SCR has four types of resets: main reset, watchdog timer reset, RAM double bit ECC reset and soft reset.

When the TC26x B-Step is first powered up, TC2x_SCR enters user mode 0 and start operation with the release of the system reset via the main reset. The CPU then starts to execute from the Boot ROM firmware. User mode 0 is used by CPUx to program the user code to the internal XRAM via the SPI interface. Once the XRAM is programmed, a new boot mode must be selected via PMSWCR1.SCRCFG. A reset must be issued by CPUx via bit PMSWCR1.SCRSTREQ to start running TC2x_SCR in the new boot mode.

In user mode 0, the 8-bit CPU starts to execute an infinite loop from the Boot ROM firmware and user code will not be executed in user mode 0 once the system reset is released. Please refer to “Boot mode” section for detailed information about the other boot modes.

The Watchdog Timer (WDT) module is capable of resetting the device if it detects a malfunction in the system.

A double bit ECC event in the SRAM can also trigger a reset.

As for soft reset, it can be triggered by application software where applicable.

13.5.2.1 Types of Reset

The following reset types are triggered by the TC2x_SCR.

- Main reset
 - Requested asynchronously when power failure (undervoltage) or synchronously by the main controller CPUx via bit PMSWCR1.SCRSTREQ. The PORST will also triggered this type of reset.
 - Reset all registers including the reset indication bits.
- SCU Watchdog Timer (WDT) reset
 - Requested by WDT overflow or out of boundary refresh event if it is enabled via RSTCON.WDTRSTEN.
 - Possible to reset all registers except the reset indication bits.
- RAM Double Bit ECC Reset
 - Requested by ECC event if it is enabled via RSTCON.ECCRSTEN.
 - Possible to reset all registers except the reset indication bits.
- Soft reset
 - Requested by soft reset event
 - Possible to reset all registers except the reset indication bits.

This section describes the definition and controls depending on the reset source.

Main Reset

When the TC26x B-Step is first powered up, TC2x_SCR enters user mode 0 and start operation with the release of the system reset via the main reset. User mode 0 is used by CPUx to program the user code to the internal XRAM via the SPI interface. Once the XRAM is programmed, a new boot mode must be selected via PMSWCR1.SCRCFG. A second main reset must then be issued as triggered by CPUx via bit PMSWCR1.SCRSTREQ to start running TC2x_SCR in the new boot mode.

In user mode 0, the 8-bit CPU starts to execute an infinite loop from the Boot ROM firmware and user code will not be executed. For more information on the other boot modes (user mode 1 and Debug mode), refer to “Boot mode” section.

Note: Resetting TC2x_SCR to enter a new boot mode is only possible via the reset that is triggered by bit PMSWCR1.SCRSTREQ from the main controller.

Watchdog Timer Reset

The watchdog timer reset is an internal reset and can be disabled by bit RSTCON.WDTRSTEN. The Watchdog Timer (WDT) maintains a counter that must be refreshed or cleared periodically. If the WDT is not serviced correctly and in time, it will generate an NMI request to the CPU and then reset the device after a predefined time-out period if this reset type is enabled. When the reset is disabled, no reset will be triggered when the overflow event or out of boundary refresh event happens.

Bit RSTST.WDTRST is used to indicate the watchdog timer reset status. The reset status can also be read via PMSWCR2.SCR.WDT bit in the main controller.

When a watchdog timer reset occurs, it is necessary to write to NMICON.SCRINTTC to indicate a valid watchdog timer reset. The main controller receives the valid signal and sends an acknowledge signal to TC2x_SCR. Once the acknowledge is received from the main controller, NMICON.SCRINTTC is cleared automatically and the watchdog timer reset indication bit (WDTRST) is allowed to be cleared via software.

Note: The watchdog timer event can be cleared by EVR reset.

Double Bit ECC Reset

The double bit ECC reset is an internal reset and can be disabled by bit RSTCON.ECCRSTEN. The 8-bit data from the CPU is encoded with an Error Correction Code (ECC) before being stored in the XRAM memory. During a read access, data is retrieved from the memory and decoded for dynamic error detection and correction. If a double bit error is detected, it will generate a reset if this reset type is enabled. When the reset is disabled, no reset will be triggered when the ECC event happens.

Bit RSTST.ECCRES is used to indicate the ECC reset status. The reset status can also be read via PMSWCR3.SCRECC bit in the main controller.

When an ECC reset occurs, it is necessary to write to NMICON.SCRINTTC to indicate a valid ECC reset. The main controller receives the valid signal and sends an

acknowledge signal to TC2x_SCR. Once the acknowledge is received from the main controller, NMICON.SCRINTTC is cleared automatically and the ECC reset indication bit (ECCRST) is allowed to be cleared via software.

Note: The ECC event can be cleared by EVR reset.

Soft Reset

Soft reset is an internal reset that is caused by a software set of the soft reset request bit, SWRQ, in RSTCON register. It can be triggered by application software where necessary.

Bit RSTST.SOFRST is used to indicate the soft reset status. The reset status can also be read via PMSWCR2.SCRRST bit in the main controller.

When a soft reset occurs, it is necessary to write to NMICON.SCRINTTC to indicate a valid soft reset. The main controller receives the valid signal and sends an acknowledge signal to TC2x_SCR. Once the acknowledge is received from the main controller, NMICON.SCRINTTC is cleared automatically and the soft reset indication bit (SOFRST) is allowed to be cleared via software.

Note: The soft reset event can be cleared by EVR reset.

Feature Discontinued

13.5.2.2 Module Reset Behavior

Table 13-12 gives an overview on how the various modules or functions of the TC2x_SCR are affected with respect to the reset type. A 'X' means that the module/function is reset to its default state.

Table 13-12 Effect of Reset on Modules/Functions

Module/ Function	Main Reset	Double Bit ECC Reset	Soft Reset	Watchdog Reset
CPU Core	X	X	X	X
SCU	X	X except reset indication bits	X except reset indication bits	X except reset indication bits
Peripherals	X	X	X	X
Debug System	X	X	X	X
Port Control	X	X	X	X
On-Chip Static RAM	Affected, unreliable	Not affected, reliable	Not affected, reliable	Not affected, reliable
Clock System	X	X	X	X

Feature Discontinued

13.5.2.3 Reset Register Description

RSTST register consist of the reset indication bits of a ECC reset, WDT reset and soft reset. **Table 13-13** shows the reset value of RSTST register after these events.

RSTST
SCR Reset Status Register (F5_H) **Reset Value: 00_H**
RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
		0			SOFTRST	WDTRST	ECCRST
		r			rwh	rwh	rwh

Field	Bits	Type	Description
ECCRST	0	rwh	Double Bit ECC Reset Indication Bit 0 _B No ECC reset occurred. 1 _B ECC reset has occurred. This bit can only be set by hardware and clear by software.
WDTRST	1	rwh	Watchdog Timer Reset Indication Bit 0 _B No watchdog reset occurred. 1 _B Watchdog reset has occurred. This bit can only be set by hardware and clear by software.
SOFTRST	2	rwh	Soft Reset Indication Bit 0 _B No soft reset occurred. 1 _B Soft reset has occurred. This bit can only be set by hardware and clear by software.
0	[7:3]	r	Reserved Returns 0 if read; should be written with 0.

Table 13-13 Reset Value of Register RSTST

Reset Source	Reset Value
Double bit ECC Reset	0000 0X1 _B
WDT Reset	0000 0X1X _B

Table 13-13 Reset Value of Register RSTST (cont'd)

Reset Source	Reset Value
Soft Reset	0000 0000 _H
Main Reset	0000 0000 _H

MRSTST
Main Reset Status Register
(FF_H)
Reset Value: 00_H
RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
			0			RST	SMURST
			r			rwh	rwh

Field	Bits	Type	Description
SMURST	0	rwh	SMU Reset Status Bit 0 _B No reset issued by SMU. 1 _B Reset issued by SMU. This bit can only be set by hardware and clear by software.
RST	1	rwh	Main System Reset Status Bit 0 _B Main system is not in reset. 1 _B Main system is in reset. This bit can only be set by hardware and clear by software.
0	[7:2]	r	Reserved Returns 0 if read; should be written with 0.

RSTCON
Reset Control Register
(F6_H)
Reset Value: 03_H
RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
SWRQ			0			WDT RSTEN	ECC RSTEN
rwh			r			rw	rw

Field	Bits	Type	Description
ECCRSTEN	0	rw	Double Bit ECC Reset Enable Bit 0 _B ECC reset is disabled. 1 _B ECC reset is enabled.
WDTRSTEN	1	rw	Watchdog Reset Enable Bit 0 _B Watchdog reset is disabled. 1 _B Watchdog reset is enabled.
SWRQ	7	rwh	Soft Reset Request 0 _B No action. 1 _B On set, soft reset is requested. This bit is automatically cleared by hardware. The SWRQ bit is a protected bit. When the Protection Scheme is activated, this bit cannot be written directly. For more information on Protection Scheme, see Section 13.5.11 .
0	[6:2]	r	Reserved Returns 0 if read; should be written with 0.

Feature Discontinued

13.5.3 Interrupt Management

This section describes the management of interrupts by the system control unit.

13.5.3.1 Overview

The Interrupt Management sub-module in the SCU controls the non-core-generated interrupt requests to the core. The core has one non-maskable interrupt (NMI) node and total 14 maskable interrupt nodes. [Figure 13.5.11](#) shows the block diagram of the Interrupt Management sub-module.

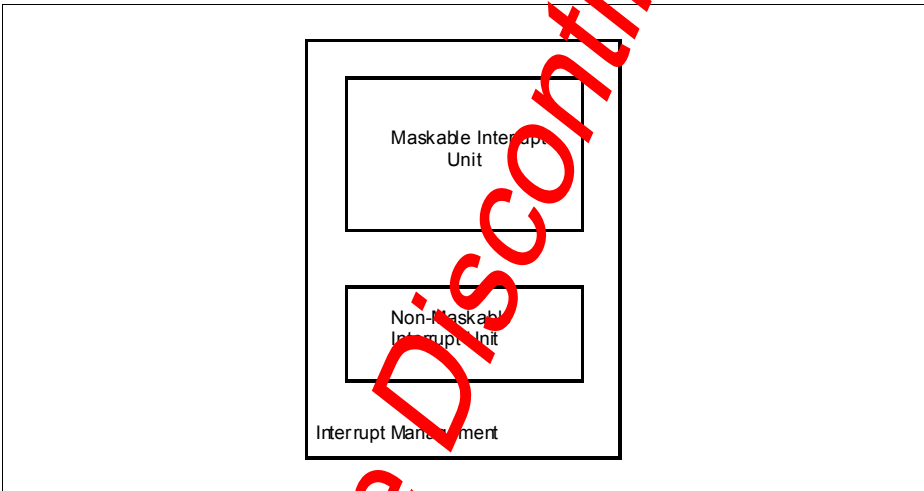


Figure 13-23 Interrupt Management Block Diagram

The non-maskable interrupt unit controls the NMI requests. Incoming NMI request is not maskable and in this sense, differs from the regular interrupts. In addition, NMI request always has the highest priority to be serviced. In the TC2x_SCR, five different sources can generate an NMI: watchdog timer prewarning, OCDS NMI, RAM ECC error, main controller interrupt, external NMI (via EXTNMI pin) and 100 MHz oscillator clock failure. Some NMI sources can be triggered by one of several events. These NMI sources are ORed to generate an NMI interrupt directly to the core. The triggering NMI sources/events are indicated in the NMI Status Register (NMISR), and in some cases the event flags are located in the peripheral register. The NMI node source control is via the NMI Control Register (NMICON).

There are generally 3 types of maskable inputs into the core: internal, external and extended interrupts. The maskable interrupt unit will generate the respective interrupt node request to the core and will maintain corresponding SCU flags and control. In

general, to support all types of peripheral interrupts, an interrupt node of the core may be shared among several interrupt sources.

Internal Interrupts

There are in total three internal interrupts, which separately come from Timer 0, Timer 1 and UART within the core. These interrupt request signals are controlled directly by the interrupt flag generated within the core. These interrupts do not go through Interrupt Management Unit in the SCU.

External Interrupts

The generation of interrupt request from an external source by edge detection in SCU is shown in **Figure 13-24**. External interrupts can be positive, negative or double edge triggered. Register EXICON0 specifies the active edge for the external interrupt. Among the external interrupts, external interrupt 0 and external interrupt 1 may be selected to bypass edge detection in SCU, as direct input to the core's NINT0 and NINT1.

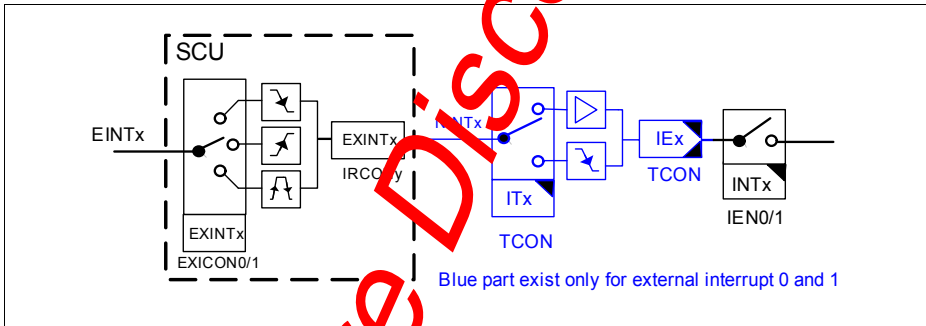


Figure 13-24 Interrupt Request Generation of External Interrupts

Extended Interrupts

Extended interrupts are not core on-chip peripherals for core-external trigger of interrupt requests to the core. There are nine such interrupt.

Interrupt signals from such on-chip peripherals are pulse triggered and active for two CPU clock cycles. These interrupt signals belonging to the same interrupt node will be latched as one direct interrupt request to the core. IRCONx (where x = 0-2) or peripheral registers hold the interrupt event flags for these extended and external interrupt events. Corresponding bits in the Interrupt Enable Registers (IEN) within the core may block or transfer these interrupt requests to the core interrupt controller. An enabled interrupt request is acknowledged when the core vectors to the interrupt routine. The software routine should clear the interrupt flags in the IRCONx registers.

As there are more peripheral interrupts than interrupt nodes supported by the core, some interrupts are multiplexed to the same interrupt node. Where possible and necessary, critical peripheral interrupts (e.g. SSC) have their own dedicated interrupt node. Refer to [Chapter 7, Interrupt System](#).

13.5.3.2 Interrupt Node Assignment

[Table 13-14](#) shows the interrupt node assignment for TC2x SCU.

Table 13-14 Interrupt Node Assignment within TC2x_SCR

Interrupt Node	Vector Address	Assignment for TC2x_SCR
NMI	0073 _H	Watchdog Timer, RAM ECC, OCDS, Main controller, External NMI (via EXT_NMI pin), 100 MHz oscillator clock failure
XINTR0	0003 _H	External Interrupt 0
XINTR1	000B _H	Timer 0
XINTR2	0013 _H	External Interrupt 1
XINTR3	001B _H	Timer 1
XINTR4	0023 _H	UART
XINTR5	002B _H	T2/TCU, LIN
XINTR6	0033 _H	WCAN
XINTR7	003B _H	SSC
XINTR8	0043 _H	External Interrupt 2
XINTR9	004B _H	External Interrupt [6:3]
XINTR10	0053 _H	External Interrupt [9:7]
XINTR11	005B _H	External Interrupt [12:10]
XINTR12	0063 _H	External Interrupt [15:13]
XINTR13	006B _H	RTC

Events/Interrupts to CPU

During run mode, there are several events (see [Table 13-15](#)) that could trigger an interrupt to the CPU.

Table 13-15 Events/Interrupts to CPUx

Modules	Events
WDT	Watchdog Timer overflow, Watchdog Timer out of boundary refresh
Memory	RAM ECC double bit error
Reset	Soft reset via bit RSTCON.SWRQ

13.5.3.3 NMI Event Flags Handling

Each NMI event and status flag is reset to default value with all types of reset. It include all the flags of NMISR register: FNMIWDT, FNMIWCLS and FNMIWEC. In the case of watchdog reset and ECC reset, the requestor can be identified via the reset indicator bits WDRTRST and ECCRST of RSTCON register.

13.5.3.4 Interrupt from/to CPUx

There is an interrupt signal from CPUx to trigger an 8-bit NMI event. It is triggered by setting a bit in register (PMSWCR2.TCINTREQ) that is located in the main SCU. The interrupt data exchange register, TCINTEXCHG, is used to indicate the type of SOC events that causes this interrupt.

In SCR, there is also an interface to trigger an interrupt to CPUx. This is done via bit NMICON.SCRINTTC. The type of SCR events that causes this interrupt to CPUx can be stored in register SCRINTEXCHG.

Feature Discontinued

13.5.3.5 Interrupt Related Registers

Several interrupt related registers are located in the SCU.

Interrupt Event Enable Control

The three interrupt events of SSC module are of interrupt structure 2. As there is no enable/disable bit(s) for these interrupt events within the module, bits are defined in the SCU register MODIEN to provide for this purpose.

MODIEN

Peripheral Interrupt Enable Register (FD_H)

Reset Value: 07_H

RMAP: 0, PAGE: 0

7	6	5	4	3	2	1	0
		0			RIREN	TIREN	EIREN
		r			rw	rw	rw

Field	Bits	Type	Description
EIREN	0	rw	SSC Error Interrupt Enable 0 _B Error interrupt is disabled 1 _B Error interrupt is enabled
TIREN	1	rw	SSC Transmit Interrupt Enable 0 _B Transmit interrupt is disabled 1 _B Transmit interrupt is enabled
RIREN	2	rw	SSC Receive Interrupt Enable 0 _B Receive interrupt is disabled 1 _B Receive interrupt is enabled
0	[7:3]	r	Reserved Returns 0 if read; should be written with 0.

Interrupt Data Exchange Registers

2 registers are used to exchange interrupt data between main controller and TC2x_SCR as described in [Section 13.5.3.4](#).

SCRINTEXCHG
SCR Interrupt Data Exchange Register (FE_H)
Reset Value: 00_H
RMAP: 0, PAGE: 0

7	6	5	4	3	2	1	0
SCRINT							

rw

Field	Bits	Type	Description
SCRINT	[7:0]	rw	Data exchange from SCR to main controller Fast data exchange between SCR and main controller. The data can be read via PMSWCR2.SCRINT in the main SCU.

TCINTEXCHG
TriCore Interrupt Data Exchange Register (FE_H)
Reset Value: 00_H
RMAP: 0, PAGE: 0

7	6	5	4	3	2	1	0
TCINT							

r

Field	Bits	Type	Description
TCINT	[7:0]	r	Data exchange from main controller to SCR Fast data exchange between main controller and SCR. The data can only be read in SCR.

Other Interrupt Related Registers

The following registers are related to the interrupt system:

- NMICON
- NMISR
- IRCON0
- EXICON0, EXICON1

 All interrupt related registers, except MODIEN, SCRINTEXCHG and TCINTEXCHG registers, are described in the Interrupt System [Chapter 7](#).

13.5.4 Boot and Startup

This section describes the boot mode for TC2x_SCR. After the first power-up, TC2x_SCR will enter User mode 0. Subsequently, bit PMSWCR1.SCRCFG is used to change the boot mode that TC2x_SCR will enter after a reset is asserted. A reset to the TC2x_SCR can be triggered by a PMSWCR1.SCRSTREQ reset request to start it off in the chosen mode. However, if an internal TC2x_SCR reset happens during standby mode (eg, WDT reset), the previous SCRCFG value will be used. **Table 13-16** shows the various boot modes available in TC2x_SCR.

Table 13-16 Boot Modes

SCRCFG Value	Boot Mode	Descriptions
00 _H	User Mode 0	Default mode after cold reset with no user code in XRAM and CPUx can download code using this mode. User code in XRAM will not be executed
01 _H	User Mode 1	User Code in XRAM at address 0000 _H will be executed
02 _H	OCDS Mode	Debug mode using SCR DAP0_0 pin and DAP1_0 pin
03 _H		Debug mode using SCR DAP0_1 pin and DAP1_1 pin
04 _H - 05 _H		Debug mode using SCR SPD_0 pin
06 _H - 07 _H		Debug mode using SCR SPD_1 pin
0A _H - 0B _H		Debug mode using SOC DAP interface
0C _H - 0F _H		Debug mode using SOC SPD interface

At end of the startup routine, an interrupt is triggered to the main controller by setting bit NMICON.SCRINTTC to 1 with a value of 80_H in SCRINTEXCHG register depending on the boot mode. This is used to indicate that TC2x_SCR has finished executing the startup code and is ready for CPUx to download code to XRAM in User mode 0 or jump to user code in User mode 1 or jump to debug monitor code in OCDS mode.

Note: To change the boot mode, TC2x_SCR has to be enabled first. Changing of boot mode when TC2x_SCR is disabled will have no effect.

13.5.4.1 Boot ROM Operating Mode

After a reset, the CPU will always start by executing the Boot ROM code which occupies the program memory address space 0000H – 07FFH. The Boot ROM start-up procedure

will first switch the address space for the Boot ROM to D000H – D7FFH. The remaining Boot ROM start-up procedure will be executed from D60XH. The memory organization of the TC2x_SCR shown in this document is after the Boot ROM address switch where the different operating modes are executed.

User Mode 0

User Mode 0 is the default boot mode after a cold reset where XRAM memory is not programmed. This mode allows the downloading of code to XRAM memory via CPUx. In this mode, the CPU will only execute start-up code in Boot ROM. At the end of start-up code, an interrupt would be triggered to the main controller by setting bit NMICON.SCRINTTC to 1 with a value of 80_H in SCRINTEXCHG register. This is used to indicate that TC2x_SCR is ready for CPUx to download code to XRAM.

User Mode 1

If the User mode 1 is selected, the Boot ROM will jump to program memory address 0000_H to execute the user code in the XRAM memory. To use this mode, the XRAM must be pre-loaded with user code. This is the normal operating mode of the TC2x_SCR.

For the last 8 bytes of XRAM starting at address 1FF8_H, user need to program 4 sets of pre-fixed bytes with each set containing 55_H followed by AA_H. User code will not be executed and SCR will enter an endless loop if the memory content does not match these data sequence. It is used to avoid an unintentional entry to User Mode 1. Before entering the endless loop, the start-up code will trigger an interrupt to the main controller by setting bit NMICON.SCRINTTC to 1 with a value of 81_H in SCRINTEXCHG register. If there is a match, the same interrupt is triggered to the main controller with a value of 80_H in SCRINTEXCHG register to indicate the execution of user code.

OCDS Mode with SCR DAP/SPD pin

If the OCDS mode with SCR DAP/SPD pin is selected, the debug mode will be entered for debugging program code. The OCDS hardware is initialized and a jump to program memory address 0000_H is performed next. The user code in the XRAM memory is executed and the debugging process may be started.

Note: The DAP/SPD pin must be directly accessible by TC2x_SCR. The control of these pins must be given to TC2x_SCR using bit Pn_PCSR.SELx before enabling this boot mode.

OCDS Mode with SOC DAP/SPD pin

If the OCDS mode with SOC DAP/SPD pin is selected, the debug mode will be entered for debugging program code using the SOC DAP/SPD pins that are also used by the main debug system. The OCDS hardware is initialized and a jump to program memory

address 0000_H is performed next. The user code in the XRAM memory is executed and the debugging process may be started.

Feature Discontinued

13.5.5 Power Management

This section describes the features and functionality provided for power management of the TC2x_SCR. In addition, the exiting of standby mode via TC2x_SCR will be described in this section. Standby mode is one of the power saving modes in TC26x B-Step.

13.5.5.1 Overview

The TC2x_SCR power-management system allows software to configure the various processing units so that they automatically adjust to draw the minimum necessary power for the application.

There are generally two power modes: Active Mode and Idle Mode. The operation of the system components in each of these states can be configured by software. The power modes provide flexible reduction of power consumption through a combination of techniques, including:

- Stopping the CPU clock
- Stopping the clocks of other system components individually
- Clock-speed reduction of some peripheral components

In Active mode, the CPU and peripherals is running in system frequency.

In Idle mode, the Core is stopped with its clock disabled. Peripherals, with respective input clocks not disabled, are still functional. The watchdog timer, however, must be disabled by user before the system enters into an idle mode; otherwise, it may generate an internal reset when an overflow occurs and this will disrupt the idle mode. The idle mode is terminated automatically when any enabled interrupt signal is detected.

13.5.5.2 Functional Description

This section describes the power save modes, their operations, and entry and exit. It also describes the respective behavior of TC2x_SCR system components.

Idle Mode

Software requests Idle Mode by setting the bit PCON.IDLE to 1.

The power management state machine (PMSM) posts an idle request signal to the CPU. The CPU finishes its current operation, sends an acknowledge signal back to the PMSM, and then enters into an inactive state in which the CPU clock (CCLK) is disabled.

The system returns to either active mode when there is an incoming interrupt request to an enabled interrupt mode.

Interrupt to the CPU causes the CCLK to recover. Upon RETI instruction, the CPU will return to execute the next instruction following the instruction which sets the IDLE bit to 1.

Note: The user is advised to disable watchdog timer prior to entering the Idle Mode.

13.5.5.3 Exit Standby Mode via TC2x_SCR

In standby mode, one of the power saving modes in TC26x B-Step, TC2x_SCR could be operating in either active mode or idle mode. Hence, TC2x_SCR could be one of the wake-up source to exit standby mode. Before entering standby mode, the followings steps related to TC2x_SCR need to be setup before issuing the standby request :

- Configure the wake-up signals related to TC2x_SCR that would trigger the exit from standby mode via PMSWCR0 register
- Configure the clock selection for TC2x_SCR in standby mode via SCRCLKSEL bit.

To exit from standby mode, one of the following wake-up sources from TC2x_SCR could be used :

- WDT event
- SRAM ECC double bit error
- Software request
- WCAN WUF event
- RTC compare match event

Besides the specified events such as WDT overflow event, SRAM ECC error, WCAN event or RTC compare match event, other types of events from SSC, T0, T1, T2CCU and UART could issue a wake-up request via the bit SCRWKP in STDBYWKP register.

13.5.5.4 Register Description

PCON [Note: This register is located within XC800 core]

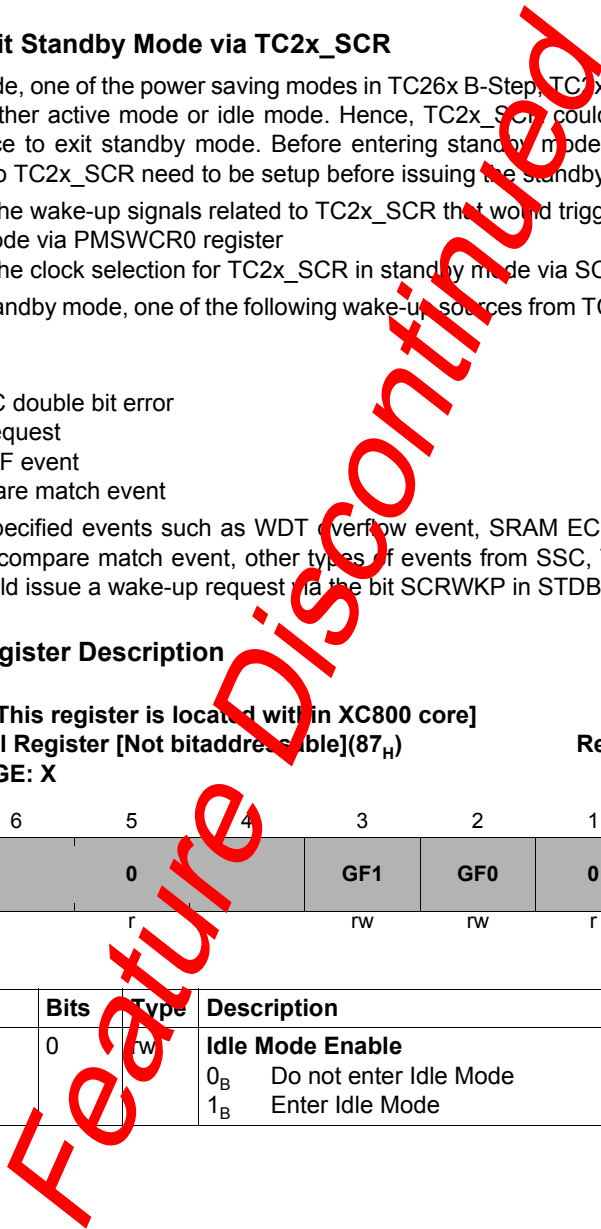
Power Control Register [Not bitaddressable](87_H)

Reset Value: 00_H

RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
SMOD		0		GF1	GF0	0	IDLE
rw		r		rw	rw	r	rw

Field	Bits	Type	Description
IDLE	0	rw	Idle Mode Enable 0 _B Do not enter Idle Mode 1 _B Enter Idle Mode



STDBYWKP
Standby Mode Wake-up Register (FA_H)
Reset Value: 00_H
RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
	0		ECCWK SEL	WDTWK SEL	WCANWK SEL	RTCWK SEL	SCRWK P
	r		rw	rw	rw	rw	rwh

Field	Bits	Type	Description
SCRWK	0	rwh	Software Wake-up from Standby mode 0 _B No request to trigger a wake-up from standby mode. 1 _B Trigger a wake-up from standby mode. This bit is automatically cleared by hardware.
RTCWKSEL	1	rw	RTC Wake-up Select from Standby mode 0 _B Wake-up from standby mode via RTC is not selected. 1 _B Wake-up from standby mode via RTC is selected.
WCANWKSEL	2	rw	WCAN WUF Wake-up Select from Standby mode 0 _B Wake-up from standby mode via WCAN WUF is not selected. 1 _B Wake-up from standby mode via WCAN WUF is selected.
WDTWKSEL	3	rw	WDT Wake-up Select from Standby mode 0 _B Wake-up from standby mode is not selected when WDT event occurred. 1 _B Wake-up from standby mode is selected when WDT event occurred.
ECCWKSEL	4	rw	XRAM ECC Wake-up Select from Standby mode 0 _B Wake-up from standby mode is not selected when XRAM ECC event occurred. 1 _B Wake-up from standby mode is selected when XRAM ECC event occurred.
0	[7:5]	r	Reserved Returns 0 if read; should be written with 0.

PMSR0 and PMSR1 registers provide the power management status for CPUx. These status registers store the latched value of PMCSRx.PMST [x = 0, 1] bits located in the main SCU. During standby mode, the main SCU will be powered down and the standby mode status will become unavailable. The standby mode status will be obtained via EVR, indicated by the PMSR1.STBY bit.

PMSR0
Power Management Status Register 0 (FD_H)
Reset Value: 11_H
RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
0	PMST1			0	PMST0		
r	rh			r	rh		

Field	Bits	Type	Description
PMST0	[2:0]	rh	Power Management Status for CPU0 000 _B Reserved, do not use this combination 001 _B Normal Run Mode ¹⁾ 010 _B CPU Idle Mode requested 011 _B CPU Idle Mode acknowledged 100 _B Sleep Mode requested 101 _B Reserved, do not use this combination 110 _B Standby Mode requested 111 _B Reserved, do not use this combination
0	3	r	Reserved Returns 0 if read; should be written with 0.
PMST1	[6:4]	rh	Power Management Status for CPU1 000 _B Reserved, do not use this combination 001 _B Normal Run Mode ¹⁾ 010 _B CPU Idle Mode requested 011 _B CPU Idle Mode acknowledged 100 _B Sleep Mode requested 101 _B Reserved, do not use this combination 110 _B Standby Mode requested 111 _B Reserved, do not use this combination
0		r	Reserved Returns 0 if read; should be written with 0.

1) After a reset, all CPUs are in "Normal Run Mode", but this does not mean that all CPUs are executing code. This mode also includes the CPU "halt" mode which is the start-up default for all except CPU0.

PMSR1
Power Management Status Register 1 (FE_H)
Reset Value: 01_H
RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
	0		STBY				
	r		rh			r	

Field	Bits	Type	Description
0	[3:0]	r	Reserved Returns 0 if read; should be written with 0.
STBY	4	rh	Standby Mode Status 0 _B Standby Mode not entered 1 _B Standby Mode entered
0	[7:5]	r	Reserved Returns 0 if read; should be written with 0.

Feature Discontinued

13.5.6 Flexible Peripheral Management

The Flexible Peripheral Management sub-module provides the system designer greater control on the operational status of each individual peripheral. Peripherals which are not required for a particular functionality can be disabled by programming the assigned register bits which would gate off the clock inputs. This would further reduce overall power consumption of the microcontroller.

Each register bit controls one peripheral. When this bit is set, the request signal to gate the peripheral clock is activated. The peripheral will then synchronize the gating off of the clock to the peripheral. In TC2x_SCR, the clock inputs of all peripherals as specified in the PMCON1 register are gated off upon power on reset.

Feature Discontinued

13.5.6.1 Peripheral Management Registers

PMCON1

Peripheral Management Control Register 1(F7_H)

Reset Value: 00_H

RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
	0			WCAN _DIS	RTC _DIS	T2CCU _DIS	SSC _DIS
	r			rw	rw	rw	rw

Field	Bits	Type	Description
SSC_DIS	0	rw	SSC Disable Request. Active high. 0 _B SSC is in normal operation. 1 _B Request to disable the SSC. (default)
T2CCU_DIS	1	rw	T2CCU Disable Request. Active high. 0 _B T2CCU is in normal operation. 1 _B Request to disable the T2CCU. (default)
RTC_DIS	2	rw	RTC Disable Request. Active high. 0 _B RTC is in normal operation. 1 _B Request to disable the RTC. (default)
WCAN_DIS	3	rw	WCAN Disable Request. Active high. 0 _B WCAN is in normal operation. 1 _B Request to disable the WCAN. (default)
0	[7:4]	r	Reserved Returns 0 if read; should be written with 0.

13.5.7 XRAM Addressing Modes

External Data Memory (XRAM) is accessed using MOVX instructions only. These use either 8-bit or 16-bit indirect address. The DPTR register is used for 16-bit addressing and either register R0 or R7 is used to form the 8-bit address.

13.5.7.1 Access to XRAM Using the DPTR (16-bit addressing Mode)

The external memory can be accessed by two read/write instructions, which use the 16-bit DPTR for indirect addressing. These instructions are:

- MOVX A, @DPTR (Read)
- MOVX @DPTR, A (Write)

13.5.7.2 Access to XRAM Using the Register R0/R1 (8-bit addressing Mode)

The TC2x_SCR provides also instructions for accesses to XRAM using the 8-bit address (indirect addressing with registers R0 or R1). These instructions are:

- MOVX A, @Ri (Read)
- MOVX @Ri, A (Write)

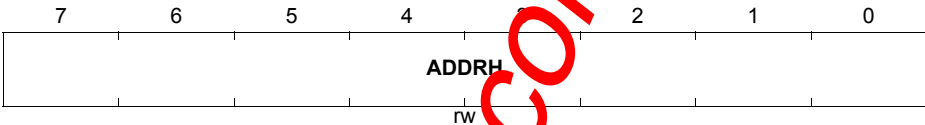
To access the XRAM properly, higher order address must be put in SFR XADDRH. And this write instruction must be preceding the MOVX instructions.

XADDRH

On-Chip XRAM Address Higher Order (F3_H)

Reset Value: 00_H

RMAP: 0, PAGE: 1



Field	Bits	Type	Description
ADDRH	[7:0]	rw	Higher Order of On-chip XRAM Address The range for TC2x_SCR is from 00 _H .

Feature Discontinued

13.5.8 Module Suspend Control

When the On-Chip Debug Support (OCDS) is in Monitor Mode (MMCR2.MMODE = 1) and the Debug-Suspend signal is active (MMCR2.DSUSP = 1), timers in certain modules in TC2x_SCR can be suspended based on the settings of their corresponding module suspend bits in register MODSUSP. When suspended, only the timer stops counting as the counter input clock is gated off. The modules are still clocked so that module registers are accessible.

MODSUSP

Module Suspend Control Register (FB_H)
RMAP: 0, PAGE: 1

Reset Value: 01_H

7	6	5	4	3	2	1	0
0				CCTSUSP	T2SUSP	RTCSUSP	WDSUSP
				r	rw	rw	rw

Field	Bits	Type	Description
WDSUSP	0	rw	SCU Watchdog Timer Debug Suspend Bit 0 _B WDT will not be suspended. 1 _B WDT will be suspended.
RTCSUSP	1	rw	Real-time Clock Debug Suspend Bit 0 _B Real-time clock will not be suspended. 1 _B Real-time clock will be suspended.
T2SUSP	2	rw	Timer 2 Debug Suspend Bit 0 _B Timer2 will not be suspended. 1 _B Timer2 will be suspended.
CCTSUSP	3	rw	Compare/Capture Timer Debug Suspend Bit 0 _B CCT will not be suspended. 1 _B CCT will be suspended.
0	[7:4]		Reserved Returns 0 if read; should be written with 0.

13.5.9 JTAG ID

The following assignment of JTAG ID are used for TC2x_SCR :

VERSION = 1_H, PART_NUMBER = 01EC_H, i.e. JTAG ID Register = 101EC083_H

13.5.10 General Port Control

The SCU contains control registers for input pin selection of SSC, UART, External interrupts [15:0], T0, T1, T2CCU and WCAN modules.

13.5.10.1 Input Pin Function Selection

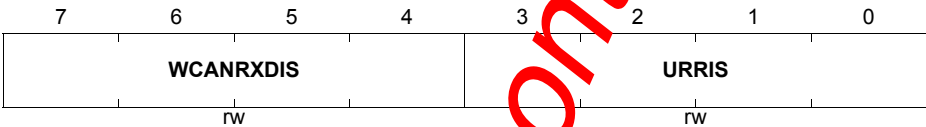
MODPISEL0 register are used for input pin selection of UART and WCAN modules.

MODPISEL0

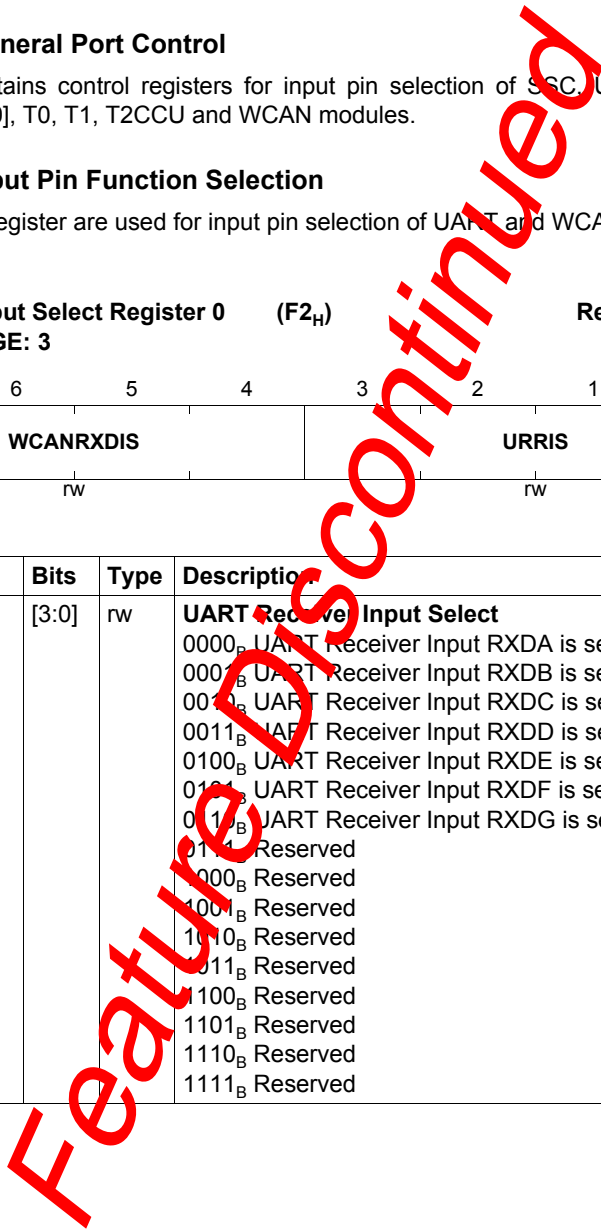
Peripheral Input Select Register 0 (F2_H)

Reset Value: 00_H

RMAP: 0, PAGE: 3



Field	Bits	Type	Description
URRIS	[3:0]	rw	UART Receiver Input Select 0000 _B UART Receiver Input RXDA is selected. 0001 _B UART Receiver Input RXDB is selected. 0010 _B UART Receiver Input RXDC is selected. 0011 _B UART Receiver Input RXDD is selected. 0100 _B UART Receiver Input RXDE is selected. 0101 _B UART Receiver Input RXDF is selected. 0110 _B UART Receiver Input RXDG is selected. 0111 _B Reserved 1000 _B Reserved 1001 _B Reserved 1010 _B Reserved 1011 _B Reserved 1100 _B Reserved 1101 _B Reserved 1110 _B Reserved 1111 _B Reserved



Field	Bits	Type	Description
WCANRXDIS	[7:4]	rw	WCAN Receiver Input Select 0000 _B WCAN Receiver Input WCANRXDA is selected. 0001 _B WCAN Receiver Input WCANRXDB is selected. 0010 _B WCAN Receiver Input WCANRXDC is selected. 0011 _B WCAN Receiver Input WCANRXDD is selected. 0100 _B WCAN Receiver Input WCANRXDE is selected. 0101 _B WCAN Receiver Input WCANRXDF is selected. 0110 _B WCAN Receiver Input WCANRXDG is selected. 0111 _B Reserved 1000 _B Reserved 1001 _B Reserved 1010 _B Reserved 1011 _B Reserved 1100 _B Reserved 1101 _B Reserved 1110 _B Reserved 1111 _B Reserved

MODPISEL1 register are used for input pin selection of WCAN module and External interrupt 0, 1 and 2.

MODPISEL1

Peripheral Input Select Register 1 (F3_H)

Reset Value: 00_H

RMAP: 0, PAGE: 3

7	6	5	4	3	2	1	0
EXINT2IS		EXINT1IS		EXINT0IS		WCANSYNCS	
rw		rw		rw		rw	

Field	Bits	Type	Description
WCANSYNCS	[1:0]	rw	WCAN SYNC Input Select 00 _B WCAN Input WCANSYNCA is selected. 01 _B WCAN Input WCANSYNCB is selected. 10 _B Reserved 11 _B Reserved

Field	Bits	Type	Description
EXINT0IS	[3:2]	rw	External Interrupt Input 0 Input Select 00 _B External Interrupt Input EXINT0A is selected. 01 _B External Interrupt Input EXINT0B is selected. 10 _B External Interrupt Input EXINT0C is selected. 11 _B External Interrupt Input EXINT0D is selected.
EXINT1IS	[5:4]	rw	External Interrupt Input 1 Input Select 00 _B External Interrupt Input EXINT1A is selected. 01 _B External Interrupt Input EXINT1B is selected. 10 _B External Interrupt Input EXINT1C is selected. 11 _B External Interrupt Input EXINT1D is selected.
EXINT2IS	[7:6]	rw	External Interrupt Input 2 Input Select 00 _B External Interrupt Input EXINT2A is selected. 01 _B External Interrupt Input EXINT2B is selected. 10 _B External Interrupt Input EXINT2C is selected. 11 _B External Interrupt Input EXINT2D is selected.

MODPISEL2 register are used for input pin selection of T2CCU modules.

MODPISEL2
Peripheral Input Select Register 2 (74_H)
Reset Value: 00_H
RMAP: 0, PAGE: 3

7	6	5	4	3	2	1	0
T2CC3EXINT6IS		T2CC2EXINT5IS		T2CC1EXINT4IS		T2CC0EXINT3IS	
rw		rw		rw		rw	

Field	Bits	Type	Description
T2CC0EXINT3IS	[1:0]	rw	External Interrupt 3/T2CCU Capture Channel 0 Input Select 00 _B External Interrupt 3 Input EXINT0A/T2CCU Capture Channel 0 Input T2CC0A is selected. 01 _B External Interrupt 3 Input EXINT0B/T2CCU Capture Channel 0 Input T2CC0B is selected. 10 _B External Interrupt 3 Input EXINT0C/T2CCU Capture Channel 0 Input T2CC0C is selected. 11 _B Reserved

Field	Bits	Type	Description
T2CC1EXINT4IS	[3:2]	rw	External Interrupt 4/T2CCU Capture Channel 1 Input Select 00 _B External Interrupt 4 Input EXINT4A/T2CCU Capture Channel 1 Input T2CC1A is selected. 01 _B External Interrupt 4 Input EXINT4B/T2CCU Capture Channel 1 Input T2CC1B is selected. 10 _B External Interrupt 4 Input EXINT4C/T2CCU Capture Channel 1 Input T2CC1C is selected. 11 _B Reserved
T2CC2EXINT5IS	[5:4]	rw	External Interrupt 5/T2CCU Capture Channel 2 Input Select 00 _B External Interrupt 5 Input EXINT5A/T2CCU Capture Channel 2 Input T2CC2A is selected. 01 _B External Interrupt 5 Input EXINT5B/T2CCU Capture Channel 2 Input T2CC2B is selected. 10 _B External Interrupt 5 Input EXINT5C/T2CCU Capture Channel 2 Input T2CC2C is selected. 11 _B Reserved
T2CC3EXINT6IS	[7:6]	rw	External Interrupt 6/T2CCU Capture Channel 3 Input Select 00 _B External Interrupt 6 Input EXINT6A/T2CCU Capture Channel 3 Input T2CC3A is selected. 01 _B External Interrupt 6 Input EXINT6B/T2CCU Capture Channel 3 Input T2CC3B is selected. 10 _B External Interrupt 6 Input EXINT6C/T2CCU Capture Channel 3 Input T2CC3C is selected. 11 _B Reserved

MODPISEL3 register are used for input pin selection of T2CCU and SSC modules.

MODPISEL3
Peripheral Input Select Register 3 (F5_H)
Reset Value: 00_H
RMAP: 0, PAGE: 3

7	6	5	4	3	2	1	0
CS			T2IS		T2EXIS		
rw			rw		rw		

Field	Bits	Type	Description
T2EXIS	[2:0]	rw	Timer 2 External Input Select 000 _B Timer 2 Input T2EXA is selected. 001 _B Timer 2 Input T2EXB is selected. 010 _B Timer 2 Input T2EXC is selected. 011 _B Timer 2 Input T2EXD is selected. 100 _B Timer 2 Input T2EXE is selected. 101 _B Timer 2 Input T2EXF is selected. 110 _B Timer 2 Input T2EXG is selected. 111 _B Timer 2 Input T2EXH is selected.
T2IS	[4:3]	rw	Timer 2 Input Select 00 _B Timer 2 Input T2A is selected. 01 _B Timer 2 Input T2B is selected. 10 _B Timer 2 Input T2C is selected. 11 _B Reserved
CIS	[7:5]	rw	Slave Mode Clock Input Select 000 _B SSC Slave Clock Input SCLKA is selected. 001 _B SSC Slave Clock Input SCLKB is selected. 010 _B SSC Slave Clock Input SCLKC is selected. 011 _B Reserved 100 _B Reserved 101 _B Reserved 110 _B Reserved 111 _B Reserved

MODPISEL4 registers are used for input pin selection of SSC modules.

MODPISEL4

Peripheral Input Select Register 4 (F6_H)

Reset Value: 00_H

RMAP: 0, PAGE: 3

7	6	5	4	3	2	1	0
0			SIS			MIS	
r			rw			rw	

Feature Discontinued

Field	Bits	Type	Description
MIS	[2:0]	rw	Master Mode Receive Input Select 000 _B SSC Master Receiver Input MRSTA is selected. 001 _B SSC Master Receiver Input MRSTB is selected. 010 _B SSC Master Receiver Input MRSTC is selected. 011 _B Reserved 100 _B Reserved 101 _B Reserved 110 _B Reserved 111 _B Reserved
SIS	[5:3]	rw	Slave Mode Receive Input Select 000 _B SSC Slave Receiver Input MTSRA is selected. 001 _B SSC Slave Receiver Input MTSRB is selected. 010 _B SSC Slave Receiver Input MTSRC is selected. 011 _B Reserved 100 _B Reserved 101 _B Reserved 110 _B Reserved 111 _B Reserved
0	[7:6]	r	Reserved Returns 0 in read; should be written with 0.

MODPISEL5 registers are used for input pin selection of T0 and T1 modules.

MODPISEL5

Peripheral Input Select Register 5 (F7_H)

Reset Value: 00_H

RMAP: 0, PAGE: 3

7	6	5	4	3	2	1	0
0			TOIS		T1IS		
r			rw		rw		

Field	Bits	Type	Description
T1IS	[1:0]	rw	Timer 1 Input Select 00 _B Timer 1 Input T1A is selected. 01 _B Timer 1 Input T1B is selected. 10 _B Timer 1 Input T1C is selected. 11 _B Reserved
T0IS	[3:2]	rw	Timer 0 Input Select 00 _B Timer 0 Input T0A is selected. 01 _B Timer 0 Input T0B is selected. 10 _B Timer 0 Input T0C is selected. 11 _B Reserved
0	[7:4]	r	Reserved Returns 0 if read; should be written with 0.

Feature Discontinued

13.5.11 Miscellaneous Control

This module consists of the Bit-Protection Scheme and general system control SFRs.

13.5.11.1 Bit Protection Register

The Bit-Protection Scheme disallows direct software writing of selected bits (i.e. Protected bits) by the SFR PASSWD. When the bit field MODE is 11_B, writing 10011_B to the bit field PASS opens access to writing of all protected bits and writing 10101_B to the bit field PASS closes access to writing of all protected bits. Note that access is opened for maximum 32 CCLKs if the “close access” password is not written. If “open access” password is written again before the end of 32 CCLK cycles, there will be a recount of 32 CCLK cycles.

PASSWD

Password Register

(F2_H)

Reset Value: 07_H

RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
PASS					PROTECT _S	MODE	
w					rh	rw	

Field	Bits	Type	Description
MODE	[1:0]	rw	<p>Bit-Protection Scheme Control Bit</p> <p>00_B Scheme Disabled 01_B Scheme Enabled 10_B Scheme Enabled 11_B Scheme Enabled (default)</p> <p>These two bits cannot be written directly. To change the value between 11_B and 00_B, the bit field PASS must be written with 11000_B, only then the MODE[1:0] will be registered.</p>
PROTECT_S	2	rh	<p>Bit-Protection Signal Status Bit</p> <p>This bit shows the status of the protection.</p> <p>0_B Software is able to write to all protected bits. 1_B Software is unable to write to any protected bits.</p>

Field	Bits	Type	Description
PASS	[7:3]	w	Password Bits The Bit-Protection Scheme (bpl) recognizes three patterns. 11000 _B Enables writing of the bit field MODE. 10011 _B Opens access to writing of all protected bits. 10101 _B Closes access to writing of all protected bits.

The list of protected bits is shown in [Table 13-17](#).

Table 13-17 List of Protected Bits

Register	Bit Field
CMCON	OSCRPD
RSTCON	SWRQ
WDTCN	WDTEN
CNT0 - CNT3 (RTC registers)	All bits in the registers

13.5.11.2 System Control Registers

SYSCON0 contains the main system control and status bits.

SYSCON0

System control Register 0

(F_H)

Reset Value: 04_H

RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
	0			0	0	0	RMAP
	r			r	r	r	rw

Field	Bits	Type	Description
RMAP	0	rw	Special Function Register Map Control 0 _B Accessed to non-mapped (standard) special function register area. 1 _B Accessed to mapped special function register area.
0	1, 2, [7:3]	r	Reserved Returns 0 if read; should be written with 0.

13.5.11.3 Safety Control Register

TC2x_SCR can be used to evaluate the various safety relevant inputs and informs SMU by setting bit ALARM in SCTRR register when violations are detected. This safety system would work only when the TC2x_SCR and the main system are in RUN mode.

SCTRR

SCR Safety Control Register
RMAP: 0, PAGE: 1

(FC_H)

Reset Value: 00_H

7	6	5	4	3	2	1	0
			0				ALARM
			r				rwh

Field	Bits	Type	Description
ALARM	0	rwh	SCR Alarm Trigger 0 _B No action 1 _B Trigger an alarm signal to report a fault condition This bit is automatically cleared by hardware.
0	[7:1]	r	Reserved Returns 0 if read; should be written with 0.

13.5.11.4 Spare Register

SPAREINOUT0.PULL_CTRL decides if pull-up and pull-down function is controlled by 8-bit controller or VADC Multiplexer Diagnostics (MD) feature is enabled. The 8-bit controller should be selected for data and control via Pn_PCSR register before PULL_CTRL is configured.

Note: This configuration applies to P00.2 and P00.3 only. For other port pins, the pull-up and pull-down is controlled by the 8-bit controller by default when Pn_PCSR.SELx is set to 1.

SPAREINOUT0

Spare Input Output Register 0
RMAP: 0, PAGE: 6

(FC_H)

Reset Value: 00_H

7	6	5	4	3	2	1	0
			0				PULL_CTRL
			r				rw

Field	Bits	Type	Description
PULL_CTRL	0	rw	Pull Control for Port 00 Pin 2 and Pin 3 This bit is used only if Pn_PCSR SELx is set to 1. 0 _B VADC Multiplexer Diagnostics (MD) feature is enabled, and pull-up and pull-down is not controlled by 8-bit controller. 1 _B VADC Multiplexer Diagnostics (MD) feature is disabled, and pull-up and pull-down is controlled by 8-bit controller.
0	[7:1]	r	Reserved Returns 0 if read, should be written with 0.

Feature Discontinued

13.5.12 Register Mapping

The BPI of the SCU supports the local address extension mechanism and the SFRs of the SCU kernel are organized into 6 pages.

SFR SCU_PAGE, at address F1_H, contains the page value and the page control information.

SCU_PAGE

Page Register for SCU

(F1_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
OP		STNR		0	PAGE		
w		w			rwh		

Field	Bits	Type	Description
PAGE	[2:0]	rwh	Page Bits When written, the value indicates the new page address. When read, the value indicates the currently active page = address + 1].
STNR	[5:4]	w	Storage Number This number indicates which storage bit field is the target of the operation defined by bit OP. If OP = 10, the contents of PAGE are saved in STx before being overwritten with the new value. If OP = 11, the contents of PAGE are overwritten by the contents of STx. The value written to the bit positions of PAGE is ignored. 00 _B ST0 is selected. 01 _B ST1 is selected. 10 _B ST2 is selected. 11 _B ST3 is selected.

Feature Discontinued

Field	Bits	Type	Description
OP	[7:6]	w	Operation 0x _B Manual page mode. The value of STNR is ignored and PAGE is directly written. 10 _B New page programming with automatic page saving. The value written to the bit positions of PAGE is stored. In parallel, the former contents of PAGE are saved in the storage bit field STx indicated by STNR. 11 _B Automatic restore page action. The value written to the bit positions PAGE is ignored and instead, PAGE is overwritten by the contents of the storage bit field STx indicated by STNR.
0	3	r	Reserved Returns 0 if read; should be written with 0.

Feature Discontinued

The addresses of the kernel SFRs are listed in [Table 13-18](#). Not listed in the tables is the SFR SYSCON0, which can be accessed in both standard (non-mapped) and mapped address of 8F_H.

Table 13-18 SFR Address List for Pages 0 - 7

Address	Page 0	Page 1	Page 2	Page 3
F2 _H	IRCON0	PASSWD	BCON	MODPISEL0
F3 _H	IRCON1	XADDRH	BGL	MODPISEL1
F4 _H	IRCON2	CMCON	BGH	MODPISEL2
F5 _H	NMISR	RSTST	LINST	MODPISEL3
F6 _H	NMICON	RSTCON	WDYCON	MODPISEL4
F7 _H	EXICON0	PMCON1	WDTREL	MODPISEL5
FA _H	EXICON1	STDBYWKP	WDTWINB	
FB _H	EXICON2	MODSUSP	WDTL	
FC _H	EXICON3	SCTRR	WDTH	
FD _H	MODIEN	PMSR0		
FE _H	SCRINTEXCHG	PMSR1		
FF _H	TCINTEXCHG	MRSYST		
Address	Page 4	Page 5	Page 6	Page 7
F2 _H				
F3 _H				
F4 _H				
F5 _H				
F6 _H				
F7 _H				
FA _H				
FB _H				
FC _H			SPAREINOUT0	
FD _H				
FE _H				
FF _H				

Feature Discontinued

13.6 Watchdog Timer

13.6.1 Overview

The Watchdog Timer (WDT) provides a highly reliable and secure way to detect and recover from software or hardware failures. The WDT is reset at a regular interval that is predefined by the user. The CPU must service the WDT within this interval to prevent the WDT from causing an TC2x_SCR system reset. Hence, routine service of the WDT confirms that the system is functioning properly. This ensures that an accidental malfunction of the TC2x_SCR will be aborted in a user-specified time period. The WDT reset is default enabled. It can also be disabled via PSTCON register.

The WDT is by default disabled.

In debug mode, the WDT is default suspended and stops counting (its debug suspend bit is default set i.e., MODSUSP.WDTSUSP = 1). Therefore during debugging, there is no need to refresh the WDT.

Features

- 16-bit Watchdog Timer
- Programmable reload value for upper 8 bits of timer
- Programmable window boundary
- Clock source from either the 100 kHz clock or the 100 MHz/DIV clock
- Selectable input frequency of fWDTCLK/2 or fWDTCLK/128

Feature Discontinued

13.6.2 System Information

This section consist of the system information required to use the WDT.

13.6.2.1 Reset effects

The Watchdog Timer maintains a counter which must be refreshed or cleared periodically. Otherwise, the counter will overflow and the watchdog reset will be asserted. The assertion of the reset could be disabled by setting bit RSTCON.WDTRSTEN. The occurrence of a WDT reset is indicated by the bit WDTRST in RSTST register.

13.6.2.2 Clocking Configuration

The WDT runs on either the 100 kHz clock or the 100 MHz/DIV clock. Bit WDTCON.WDTCLK is used to select the WDT clock source, fWDTCLK.

13.6.2.3 Interrupt Events and Assignment

Table 13-19 shows the non-maskable interrupt node assignment of the WDT interrupt source.

Table 13-19 WDT Events' Non-maskable Interrupt Node Control

Event	Interrupt Node Enable Bit	Interrupt Node Flag Bit	Vector Address
WDT Overflow	NMICON.NMIWDT	NMISR.FNMIWDT	73H

13.6.2.4 Module Suspend Control

The timer in WDT is by default suspended on entering debug mode. The WDT can be allowed to run in debug mode by clearing the bit WDTSUSP in SFR MODSUSP to 0. Refer to the definition of register MODSUSP.

Feature Discontinued

13.6.3 Functional Description

The Watchdog Timer (WDT) is a 16-bit timer, which is incremented by a count rate of $f_{WDTCLK}/2$ or $f_{WDTCLK}/128$. The f_{WDTCLK} clock input is selectable between the 100 kHz clock or the 100 MHz/DIV clock using the WDTCLK bit. This 16-bit timer is realized as two concatenated 8-bit timers. The upper 8 bits of the WDT can be preset to a user-programmable value via a watchdog service access in order to vary the watchdog expire time. The lower 8 bits are reset on each service access. Figure 13-25 shows the block diagram of the WDT unit.

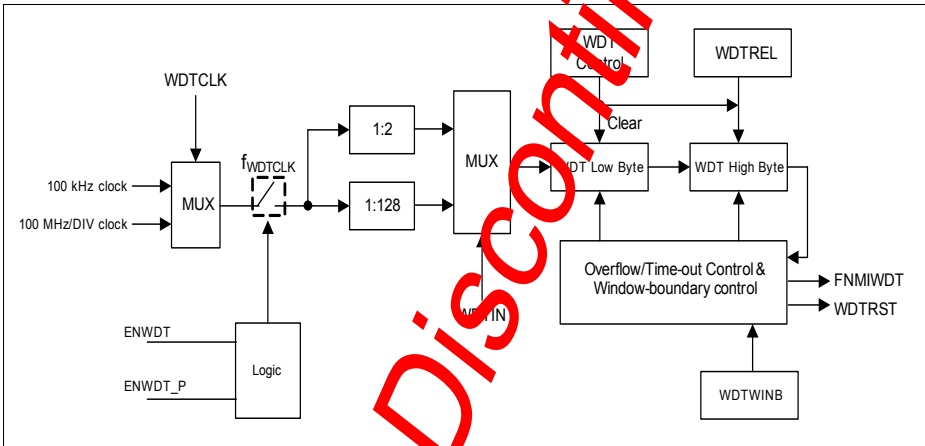


Figure 13-25 WDT Block Diagram

If the WDT is enabled by setting $WDTEN$ to 1, the timer is set to a user-defined start value and begins counting up. It must be serviced before the counter overflows. Servicing is performed through a refresh operation (setting bit $WDTRS$ to 1). This reloads the timer with the start value, and normal operation continues.

If the WDT is not serviced before the timer overflows, a system malfunction is assumed and normal mode is terminated. A WDT NMI request ($FNMIWDT$) is then asserted and prewarning is entered. The prewarning lasts for 30_H count. During the prewarning period, refreshing of the WDT is ignored and the WDT cannot be disabled. A reset ($WDTRST$) of the TC2x_SCR is imminent (if it is enabled by bit $RSTCON.WDTRSTEN$) and can no longer be avoided. The occurrence of a WDT reset is indicated by the bit $WDTRST$ in $RSTST$ register. If refresh happens at the same time an overflow occurs, WDT will not go into prewarning period.

Note: In TC26x B-Step, it is possible to wake-up the main controller from standby mode when the WDT overflow event happened by setting bit $STDBYWKP.WDTWKSEL$ to 1.

The WDT must be serviced periodically so that its count value will not overflow. Servicing the WDT clears the low byte and reloads the high byte with the present value in bit field WDTREL. Servicing the WDT also clears the bit WDTRS.

The WDT has a “programmable window boundary”, which disallows any refresh during the WDT’s count-up. A refresh during this window-boundary constitutes an invalid access to the WDT and causes the WDT to activate WDTRST, although no NMI request is generated in this instance. The window boundary is from 0000_H to the value obtained from the concatenation of WDTWINB and 00_H. This feature can be enabled by WINBEN.

After being serviced, the WDT continues counting up from the value ($\langle \text{WDTREL} \rangle * 2^8$). The time period for an overflow of the WDT is programmable in two ways:

- The input frequency to the WDT can be selected via bit WDTIN in register WDTCON to be either $f_{\text{WDTCLK}}/2$ or $f_{\text{WDTCLK}}/128$.
- The reload value WDTREL for the high byte of WDT can be programmed in register WDTREL.

The period P_{WDT} between servicing the WDT and the next overflow can be determined by the following formula:

$$P_{\text{WDT}} = \frac{2^{(1+\langle \text{WDTIN} \rangle * 6)} * (2^{16} - \text{WDTREL} * 2^8)}{f_{\text{WDTCLK}}} \quad (13.1)$$

If the Window-Boundary Refresh feature of the WDT is enabled, the period P_{WDT} between servicing the WDT and the next overflow is shortened if WDTWINB is greater than WDTREL. See also [Figure 13-26](#). This period can be calculated by the same formula by replacing WDTREL with WDTWINB. In order for this feature to be useful, WDTWINB cannot be smaller than WDTREL.

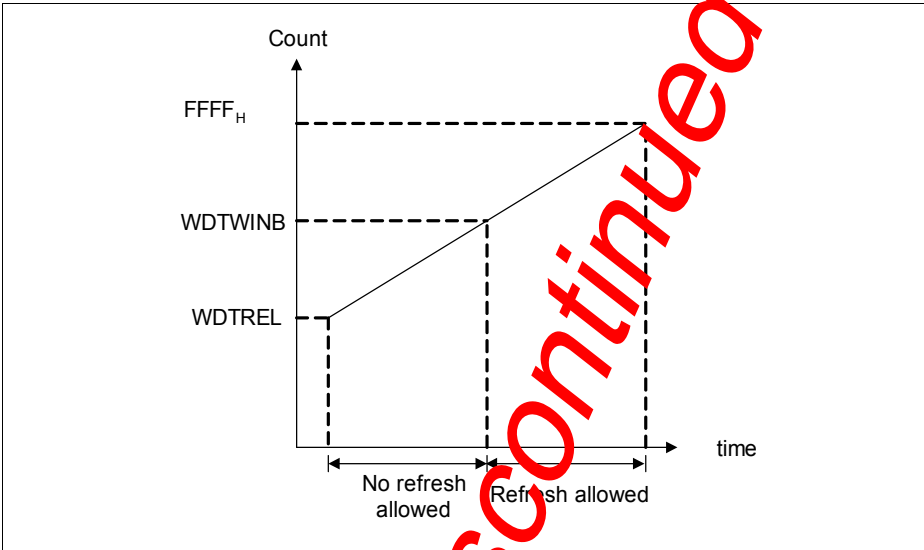


Figure 13-26 WDT Timing Diagram

Table 13-20 lists the possible ranges for the watchdog time which can be achieved using a certain module clock. Some numbers are rounded to 3 significant digits.

Table 13-20 Watchdog Time Range

Reload value in WDTREL	Prescaler for fWDT					
	2 (WDTIN = 0)			128 (WDTIN = 1)		
	20 MHz	16 MHz	12 MHz	20 MHz	16 MHz	12 MHz
FF _H	25.5 μs	32.0 μs	42.67 μs	1.63 ms	2.05 ms	2.73 ms
7F _H	3.3 ms	4.13 ms	5.5 ms	211 ms	264 ms	352 ms
00 _H	6.55 ms	8.19 ms	10.92 ms	419 ms	524 ms	699 ms

Note: For safety reasons, the user is advised to rewrite WDTCON each time before the WDT is serviced.

Note: The Watchdog Timer can be suspended when OCDS enters Monitor Mode and has the Debug-suspend signal activated, provided the respective suspend bit, WDTXSUSP in SFR MODSUSP, are set. See Module Suspend Control section.

13.6.4 Registers Description

Five SFRs control the operations of the WDT. They can be accessed from the SCU page 2 of the mapped SFR area.

Table 13-21 lists the addresses of these SFRs.

Table 13-21 Register Map

Address	Register
F6 _H	WDTCON
F7 _H	WDTREL
FA _H	WDTWINB
FB _H	WDTL
FC _H	WDTH

13.6.4.1 Watchdog Timer Registers

The Watchdog Timer Current Count Value is contained in the Watchdog Timer Register WDTH and WDTL, which are non-bitaddressable read-only register. The operation of the WDT is controlled by its bitaddressable Watchdog Control Register WDTCON. This register also selects the input clock prescaling factor. The register WDTREL specifies the reload value for the high byte of the timer. WDTWINB specifies Watchdog Window-Boundary count value.

WDTREL

Watchdog Timer Reload Register (F7_H)

Reset Value: 00_H

RMAP: 0, PAGE: 2

7	6	5	4	3	2	1	0
WDTREL							
rw							

Field	Bits	Type	Description
WDTREL	[7:0]	rw	Watchdog Timer Reload Value (for the high byte of WDT)

WDTCN
Watchdog Timer Control Register (F6_H)
Reset Value: 00_H
RMAP: 0, PAGE: 2

7	6	5	4	3	2	1	0
0	WINBEN	WDTPR	WDTCLK	WDTEN	WDTRS	WDTIN	
r	rw	rh	rw	rw	rwh	rw	

Field	Bits	Type	Description
WDTIN	0	rw	Watchdog Timer Input Frequency Selection 0 _B Input frequency is fWDTCLK/2 1 _B Input frequency is fWDTCLK/128.
WDTRS	1	rwh	WDT Refresh Start Active high. Set to start refresh operation on the watchdog timer. Cleared automatically by hardware after it is set by software.
WDTEN	2	rw	WDT Enable 0 _B WDT is disabled 1 _B WDT is enabled WDTEN is a protected bit. If the Protection Scheme is activated then this bit cannot be written directly. See protection Scheme for more details.. <i>Note: Clearing WDTEN bit to 0 during Prewarning Mode (WDTPR = 1) has no effect.</i>
WDTCLK	3	rw	Watchdog Timer Clock Source Selection 0 _B 100 kHz clock is selected 1 _B 100 MHz/DIV clock is selected
WDTPR	4	rh	Watchdog Prewarning Mode Flag 0 _B Normal mode (default after reset) 1 _B The Watchdog is operating in Prewarning Mode This bit is set to 1 when a Watchdog error is detected. The Watchdog Timer has issued an NMI trap and is in Prewarning Mode. A reset of the chip occurs after the prewarning period has expired.

Feature Discontinued

Field	Bits	Type	Description
WINBEN	5	rw	Watchdog Window-Boundary Enable 0 _B Watchdog Window-Boundary feature is disabled. (default) 1 _B Watchdog Window-Boundary feature is enabled.
0	[7:6]	r	Reserved Returns 0 if read; should be written with 0.

WDTL
Watchdog Timer, Low Byte
RMAP: 0, PAGE: 2
(FB_H)
Reset Value: 00_H

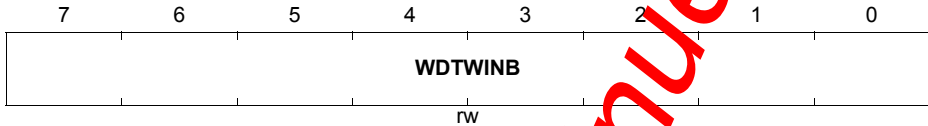
7	6	5	4	3	2	1	0
WDT							
rh							

Field	Bits	Type	Description
WDT	[7:0]	rh	Watchdog Timer Current Value

WDTH
Watchdog Timer, High Byte
RMAP: 0, PAGE: 2
(FC_H)
Reset Value: 00_H

7	6	5	4	3	2	1	0
WDT							
rh							

Field	Bits	Type	Description
WDT	[7:0]	rh	Watchdog Timer Current Value

WDTWINB
Watchdog Window-Boundary Count (FA_H)
Reset Value: 00_H
RMAP: 0, PAGE: 2


Field	Bits	Type	Description
WDTWINB	[7:0]	rw	Watchdog Window-Boundary Count Value This value is programmable. Within this Window-Boundary range from 0000 _H to (WDTWINB, 00 _H), the WDT cannot do a Refresh, else it will cause a WDTRST to be asserted. WDTWINB is matched to WDTM.

Feature Discontinued

13.7 Interrupt System

The XC800 Core supports one non-maskable interrupt (NMI) and 14 maskable interrupt requests. In addition to the standard interrupt functions supported by the core, e.g., configurable interrupt priority and interrupt masking, the TC2x_SCR interrupt system provides extended interrupt support capabilities such as the mapping of each interrupt vector to several interrupt sources to increase the number of interrupt sources supported, and additional status registers for detecting and identifying the interrupt source.

13.7.1 Interrupt Sources

The TC2x_SCR supports 14 interrupt vectors with four priority levels. These interrupt vectors are assigned to the on-chip peripherals: Timer 0, Timer 1, UART, RTC, WCAN, and SSC are each assigned one dedicated interrupt vector. T2CCU and LIN interrupts has to share an interrupt vector. External interrupt 0, 1 and 2 are each assigned one dedicated interrupt vector. The remaining 13 external interrupts are shared among 4 interrupt vectors.

A non-maskable interrupt (NMI) with the highest priority is shared by the following:

- Watchdog Timer, warning before overflow
- OCDS, on user IRAM event
- RAM ECC error
- External NMI (via EXTNMI pin)
- Main controller
- 100 MHz oscillator clock failure

Figure 13-27, **Figure 13-28**, **Figure 13-29** and **Figure 13-30** give a general overview of the interrupt sources and nodes and their corresponding control and status flags. **Figure 13-32** gives the corresponding overview for the NMI sources.

Feature Discontinued

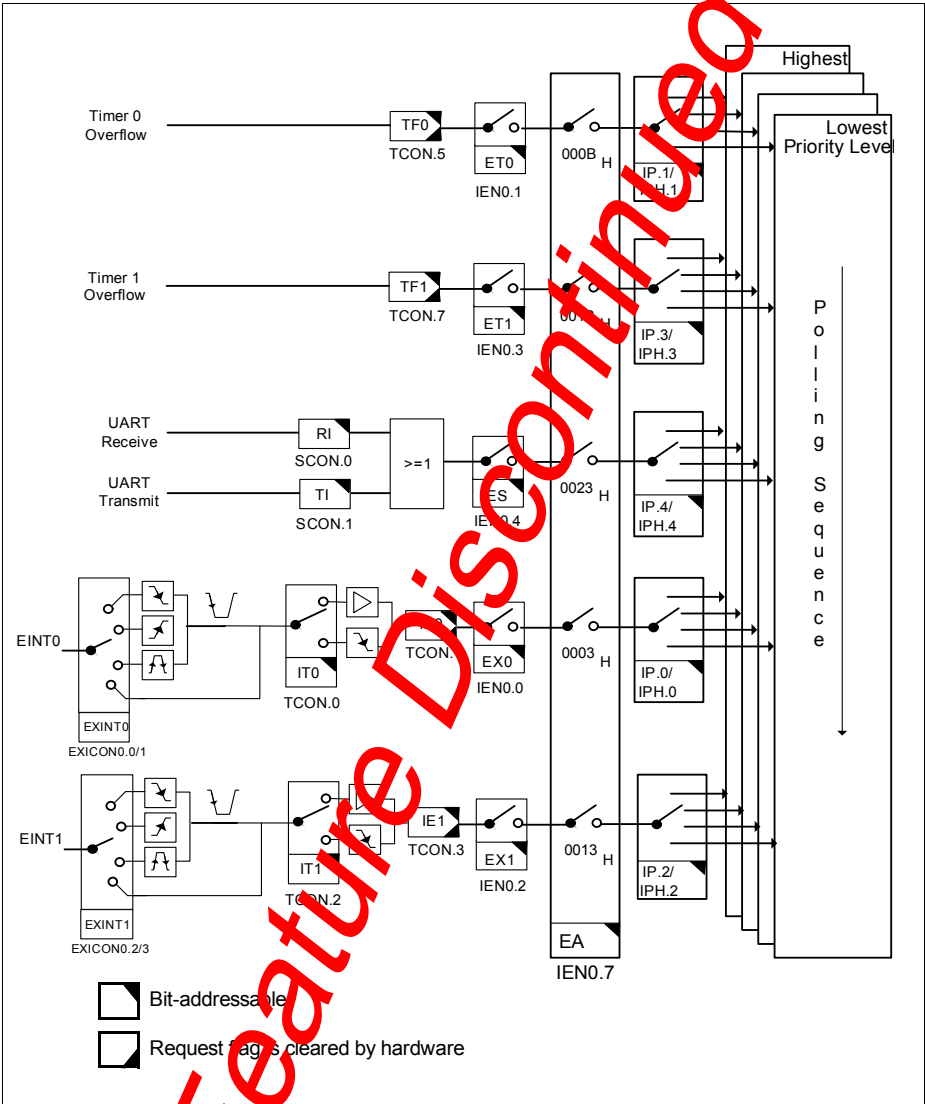


Figure 13-27 Interrupt Request Sources (Part 1)

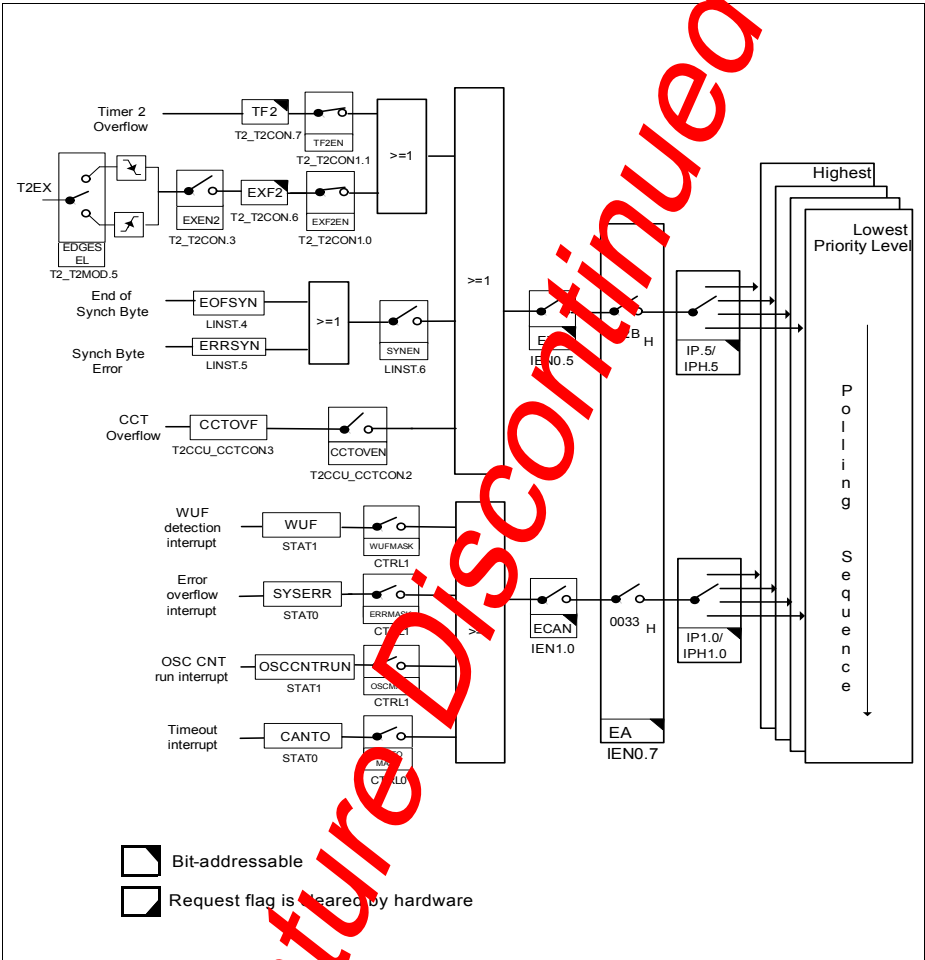


Figure 13-28 Interrupt Request Sources (Part 2)

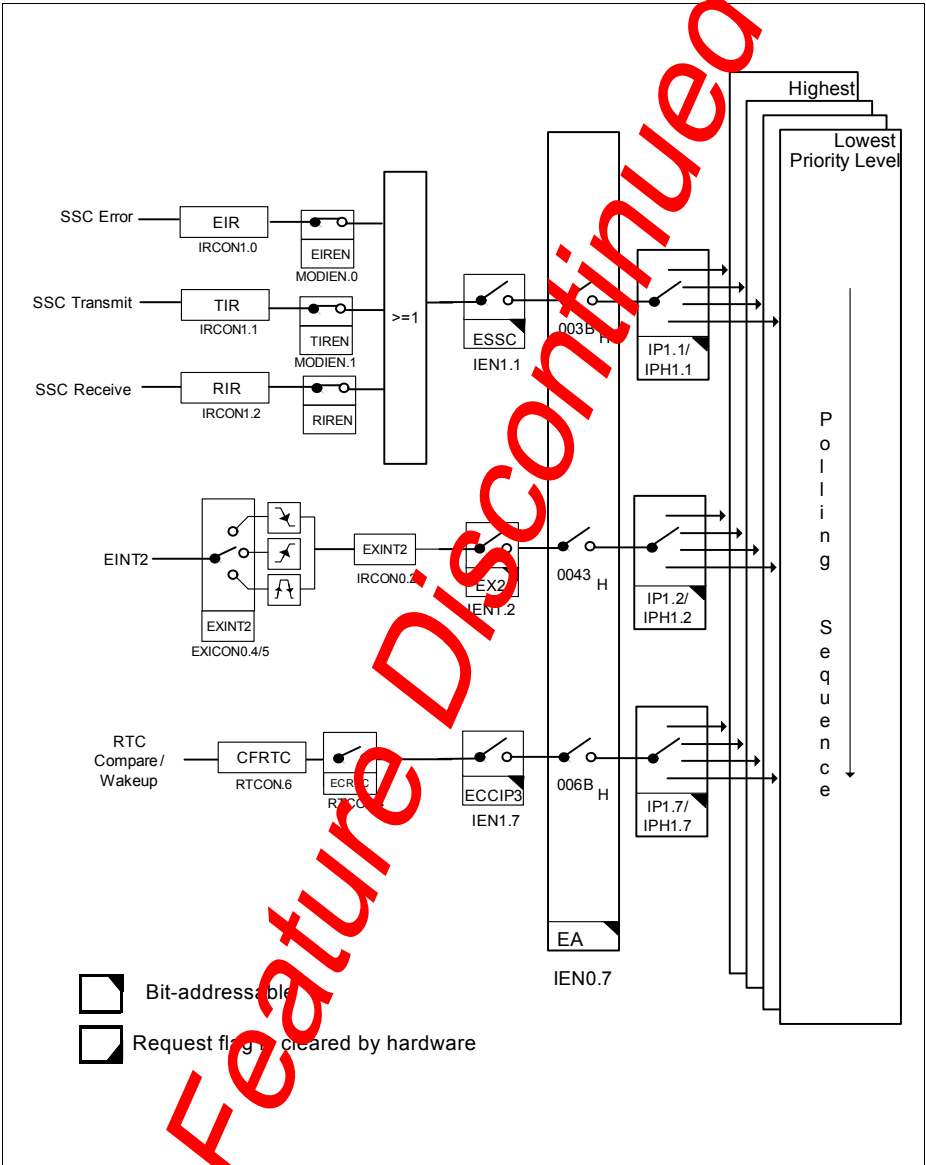


Figure 13-29 Interrupt Request Sources (Part 3)

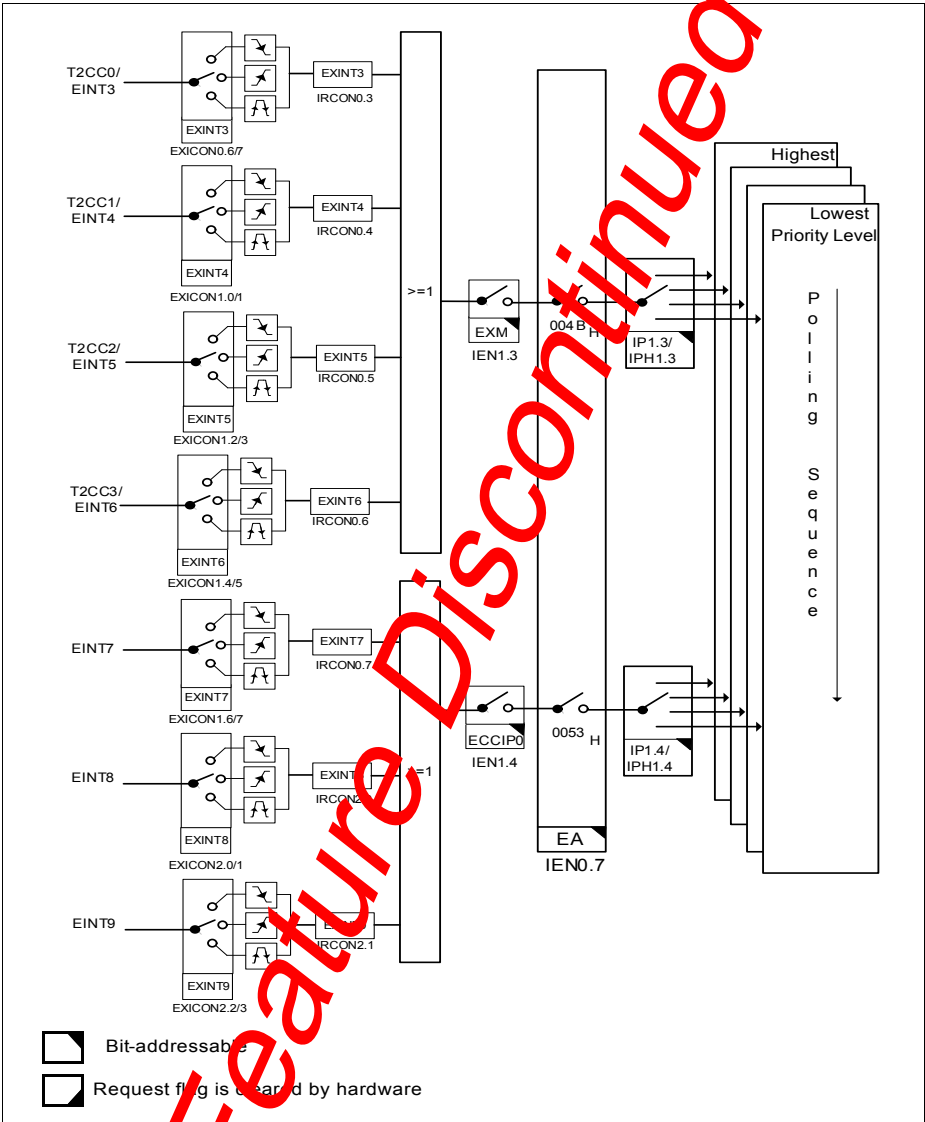


Figure 13-30 Interrupt Request Sources (Part 4)

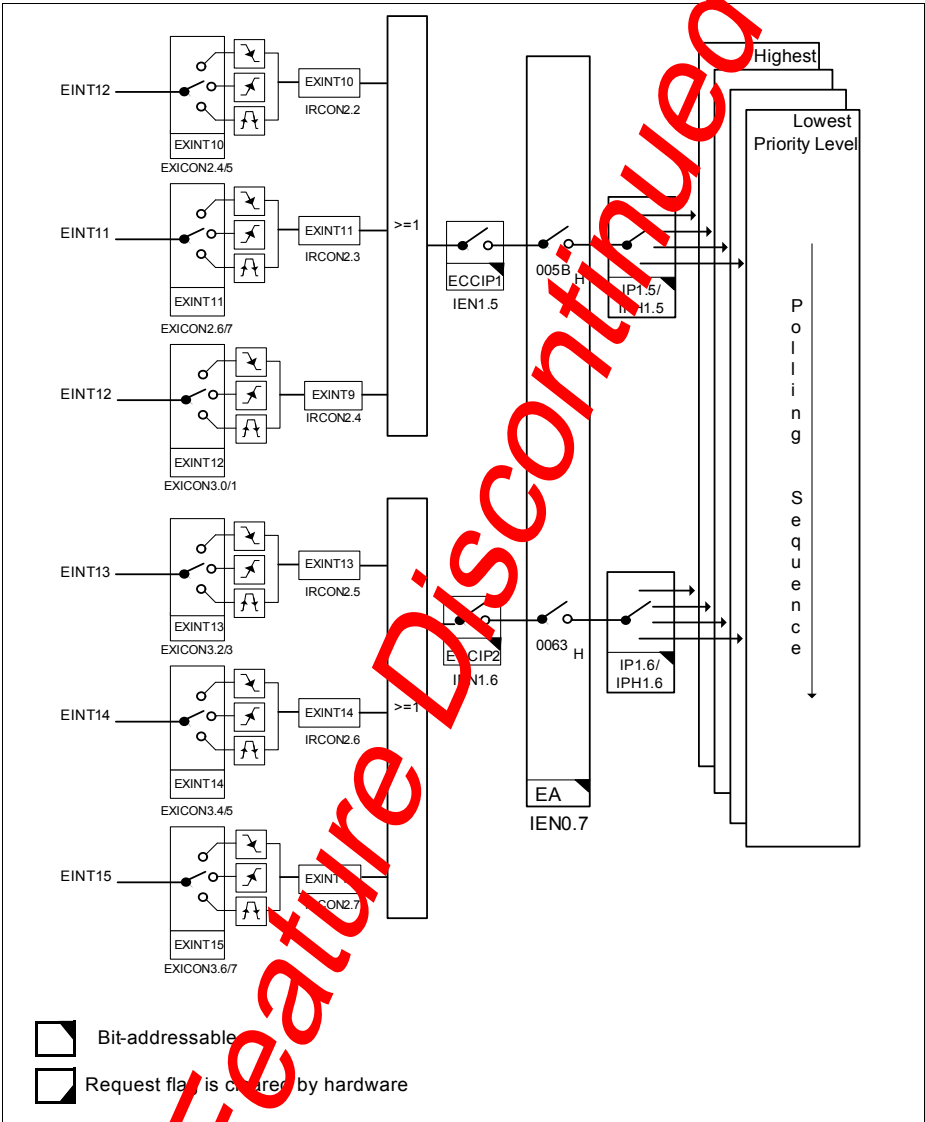


Figure 13-31 Interrupt Request Sources (Part 5)

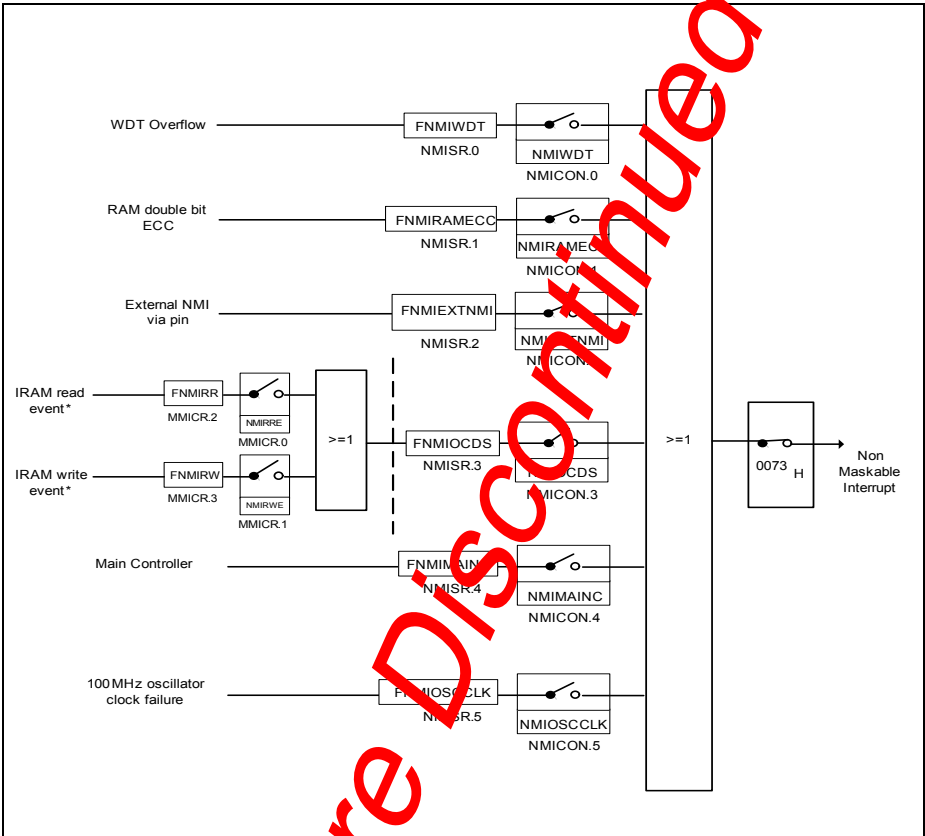


Figure 13-32 Non-Maskable Interrupt Request Source

Feature Discontinued

13.7.1.1 Interrupt Source and Vector

Each interrupt event source has an associated interrupt vector address for the interrupt node it belongs to. This vector is accessed to service the corresponding interrupt node request. The interrupt service of each interrupt node can be individually enabled or disabled via an enable bit. The assignment of the TC2x_SCP interrupt sources to the interrupt vector address and the corresponding interrupt node enable bits are summarized in [Table 13-22](#).

Table 13-22 Interrupt Vector Address

Interrupt Node	Vector Address	Assignment for TC2x_SCP	Enable Bit	SFR
NMI	0073 _H	Watchdog Timer NMI	NMIWDT	NMICON
		RAM Double Bit ECC Error NMI	NMIRAMECC	
		External NMI via EXTNMI pin	NMIEXTNMI	
		OCDS NMI	NMIOCDS	
		Main Controller NMI	NMIMAINC	
		100 MHz oscillator clock NMI	NMIOSCCLK	
XINTR0	0003 _H	External Interrupt 0	EX0	IEN0
XINTR1	000B _H	Timer 0	ET0	
XINTR2	0013 _H	External Interrupt	EX1	
XINTR3	001B _H	Timer 1	ET1	
XINTR4	0023 _H	UART	ES	
XINTR5	002B _H	T2CCU	ET2	
		LIN		

Feature Discontinued

Table 13-22 Interrupt Vector Address (cont'd)

Interrupt Node	Vector Address	Assignment for TC2x_SCR	Enable Bit	SFR
XINTR6	0033 _H	CAN wake-up filter interrupt	ECAN	IEN1
XINTR7	003B _H	SSC	ESSC	
XINTR8	0043 _H	External Interrupt 2	EX2	
XINTR9	004B _H	External Interrupt 3	EX3	
		External Interrupt 4	EX4	
		External Interrupt 5	EX5	
		External Interrupt 6	EX6	
XINTR10	0053 _H	External Interrupt 7	ECCIP0	
		External Interrupt 8		
		External Interrupt 9		
XINTR11	005B _H	External Interrupt 10	ECCIP1	
		External Interrupt 11		
		External Interrupt 12		
XINTR12	0063 _H	External Interrupt 13	ECCIP2	
		External Interrupt 14		
		External Interrupt 15		
XINTR13	006B _H	RTC Interrupt	ECCIP3	

13.7.1.2 Interrupt Source and Priority

An interrupt that is currently being serviced can only be interrupted by a higher-priority interrupt, but not by another interrupt of the same or lower priority. Hence, an interrupt of the highest priority cannot be interrupted by any other interrupt request.

If two or more requests of different priority levels are received simultaneously, the request with the highest priority is serviced first. If requests of the same priority are received simultaneously, an internal polling sequence determines which request is serviced first. Thus, within each priority level, there is a second priority structure determined by the polling sequence as shown in [Table 13-23](#).

Table 13-23 Priority Structure within Interrupt Level

Source	Level
Non-maskable Interrupt (NMI)	(highest)
External Interrupt 0	1

Table 13-23 Priority Structure within Interrupt Level (cont'd)

Source	Level
Timer 0	2
External Interrupt 1	3
Timer 1	4
UART	5
T2CCU / LIN	6
CAN wake-up filter	7
SSC	8
External Interrupt 2	9
External Interrupt [6:3]	10
External Interrupt [9:7]	11
External Interrupt [12:10]	12
External Interrupt [15:13]	13
Real-time Clock	14

13.7.2 Interrupt Structure

An interrupt event source may be generated from the on-chip peripherals or from external. Detection of interrupt events is controlled by the respective on-chip peripherals. Interrupt status flags are available for determining which interrupt event has occurred, especially useful for an interrupt node which is shared by several event sources. Each interrupt node (except NMI) has a global enable/disable bit. In most cases, additional enable bits are provided for enabling/disabling particular interrupt events (provided for NMI events). No interrupt will be requested for any occurred event that has its interrupt enable bit disabled.

The TC2x_SCR has, in general, two interrupt structures distinguished mainly by the manner in which the pending interrupt request (one per interrupt vector/node going directly to the core) is generated (due to the events) and cleared.

Common among these two interrupt structures is the interrupt masking bit, EA, which is used to globally enable or disable all interrupt requests (except NMI) to the core. Resetting bit EA to 0 only masks the pending interrupt requests from the core, but does not block the capture of incoming interrupt requests.

Note: The NMI node is similar to the other interrupt nodes, except for the exclusion of EA bit. Effectively, NMI node is non-maskable.

13.7.2.1 Interrupt Structure 1

For interrupt structure 1 (see [Figure 13-33](#)), the interrupt event will set the interrupt status flag which doubles as a pending interrupt request to the core. An active pending interrupt request will interrupt the core only if its corresponding interrupt node is enabled. Once an interrupt node is serviced (interrupt acknowledged), its pending interrupt request (represented by the interrupt status flag) may be automatically cleared by hardware (the core).

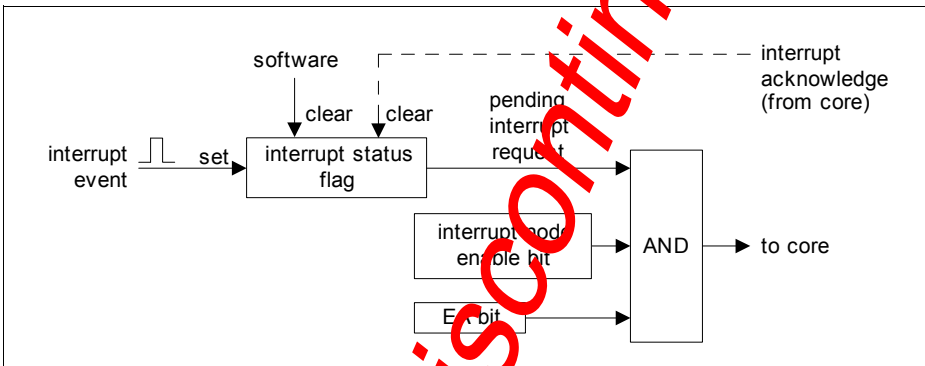


Figure 13-33 Interrupt Structure 1

For the TC2x_SCR, interrupt sources Timer 0, Timer 1, external interrupt 0 and external interrupt 1 (each have a dedicated interrupt node) will have their respective interrupt status flags TF0, TF1, IE0 and IE1 in register TCON cleared by the core once their corresponding pending interrupt request is serviced. In the case that an interrupt node is disabled (e.g. software polling is used), its interrupt status flag must be cleared by software since the core will not be interrupted (and therefore the interrupt acknowledge is not generated). For the UART, which has its dedicated interrupt node, interrupt status flags RI and TI in register SC01 will not be cleared by the core even when its pending interrupt request is serviced. The UART interrupt status flags (and hence the pending interrupt request) can only be cleared by software.

13.7.2.2 Interrupt Structure 2

This structure applies to the Timer 2, LIN, external interrupts 2 to 15, WCAN, SSC and RTC interrupt sources. For interrupt structure 2 (see [Figure 13-34](#)), the interrupt status flag does not directly drive the pending interrupt request.

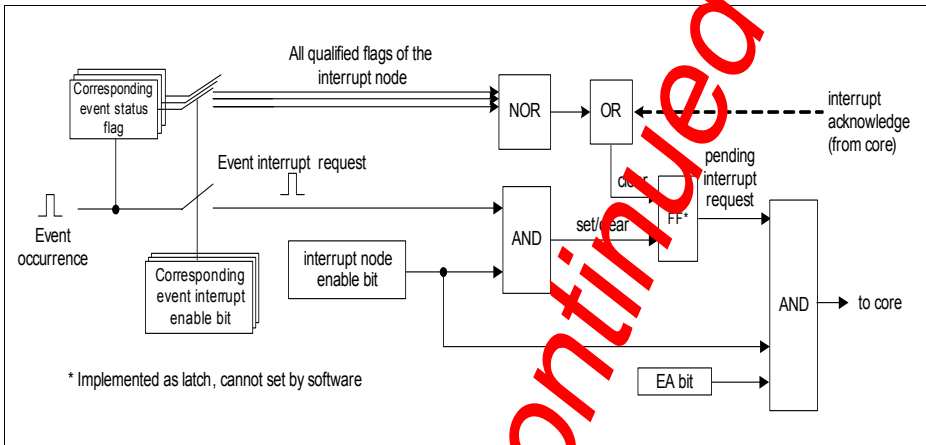


Figure 13-34 Interrupt Structure 2

An event generated by its corresponding interrupt source will set the status flag, and in parallel, if the event is enabled for interrupt, activate the pending interrupt request. Consider the case where interrupt event occurred while its interrupt node was disabled (assume global interrupt enable EA is set). When the interrupt node is enabled later, previously activated pending interrupt request will now cause an active interrupt request to the core. Note for the special case of NMI node, that it is essentially non-maskable.

An active pending interrupt request interrupts the core and is automatically cleared by hardware (the core) once the interrupt node is serviced (interrupt acknowledged); the status flag remains set and must be cleared by software. A pending interrupt request can also be cleared by software: only on clearing all interrupt-enabled status flags of the node will indirectly clear its pending interrupt request. Note that this is not exactly like interrupt structure 1 where the pending interrupt request is cleared directly by resetting the node's interrupt status flags.

In summary, the following lists in descending order the priority handling of pending interrupt request for interrupt nodes of structure 2: 1) CPU acknowledge of interrupt on vectoring to the service routine will clear the pending interrupt request, 2) Any event occurring and enabled for interrupt will set the pending interrupt request, 3) on clearing of all interrupt-enabled status flags will clear the pending interrupt request.

13.7.3 Interrupt Handling

The interrupt request signals are sampled at phase 2 in each machine cycle. The sampled requests are then polled during the following machine cycle. If one interrupt node request was active at phase 2 of the preceding cycle, the polling cycle will find it

and the interrupt system will generate a LCALL to the node's service routine, provided this hardware-generated LCALL is not blocked by any of the following conditions:

1. An interrupt of equal or higher priority is already in progress.
2. The current (polling) cycle is not in the final cycle of the instruction in progress.
3. The instruction in progress is RETI or any write access to registers IEN0/IEN1 or IP/IPH or IP1/IPH1.

Any of these three conditions will block the generation of the LCALL to the interrupt service routine. Condition 2 ensures that the instruction in progress is completed before vectoring to any service routine. Condition 3 ensures that if the instruction in progress is RETI or any write access to registers IEN0/IEN1 or IP/IPH or IP1/IPH1, then at least one more instruction will be executed before any interrupt is vectored to; this delay guarantees that changes of the interrupt status can be observed by the CPU.

The polling cycle is repeated with each machine cycle, and the values polled are the values that were present at phase 2 of the previous machine cycle. Note that if any interrupt flag is active but its node interrupt request was not responded to for one of the conditions already mentioned, and if the flag is no longer active at a later time when servicing the interrupt node, the corresponding interrupt source will not be serviced. In other words, the fact that the interrupt flag was once active but not serviced is not remembered. Every polling cycle interrogates only the pending interrupt requests.

The processor acknowledges an interrupt request by executing a hardware generated LCALL to the corresponding service routine. In some cases, hardware also clears the flag that generated the interrupt, while in other cases, the flag must be cleared by the user's software. The hardware-generated LCALL pushes the contents of the Program Counter (PC) onto the stack (but it does not save the PSW) and reloads the PC with an address that depends on the interrupt node being vectored to, as shown in the [Table 13-22](#).

Program execution returns to the next instruction after calling the interrupt when the RETI instruction is encountered. The RETI instruction informs the processor that the interrupt routine is no longer in progress, then pops the two top bytes from the stack and reloads the PC. Execution of the interrupted program continues from the point where it was stopped. Note that the RETI instruction is important because it informs the processor that the program has left the current interrupt priority level. A simple RET instruction would also have returned execution to the interrupted program, but it would have left the interrupt control system on the assumption that an interrupt was still in progress. In this case, no interrupt of the same or lower priority level would be acknowledged.

13.7.4 Interrupt Response Time

Due to an interrupt event of (the various sources of) an interrupt node, its corresponding request signal will be sampled active at phase 2 in every machine cycle. The value is not polled by the circuitry until the next machine cycle. If the request is active and conditions

are right for it to be acknowledged, a hardware subroutine call to the requested service routine will be the next instruction to be executed. The call itself takes two machine cycles. Thus, a minimum of three complete machine cycles will elapse from activation of the interrupt request to the beginning of execution of the first instruction of the service routine as shown in **Figure 13-35**.

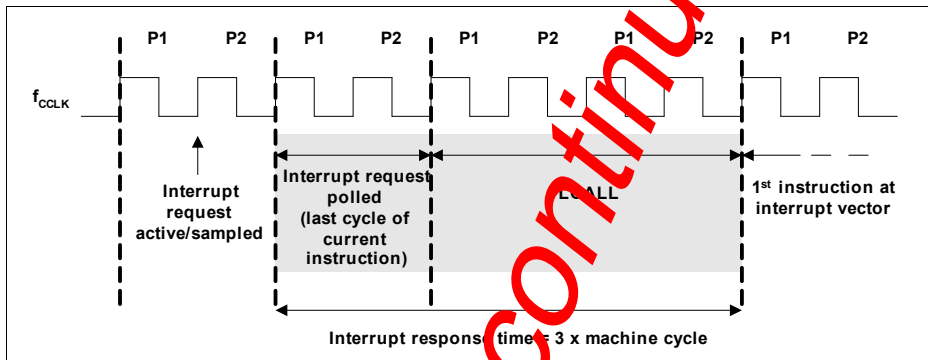


Figure 13-35 Minimum Interrupt Response Time

A longer response time would be obtained if the request is blocked by one of the three previously listed conditions:

1. If an interrupt of equal or higher priority is already in progress, the additional wait time will depend on the nature of the other interrupt's service routine.
2. If the instruction in progress is not in final cycle, the additional wait time cannot be more than three machine cycles since the longest instructions (MUL and DIV) are only four machine cycles long. See **Figure 13-36**.
3. If the instruction in progress is RETI or a write access to registers IEN0, IEN1 or IP(H), IP1(H), the additional wait time cannot be more than five cycles (a maximum of one more machine cycle to complete the instruction in progress, plus four machine cycles to complete the next instruction, if the instruction is MUL or DIV). See **Figure 13-37**.

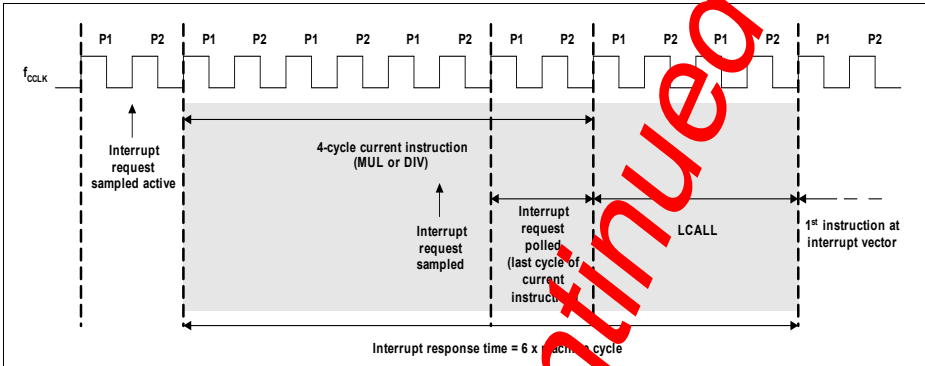


Figure 13-36 Interrupt Response Time for Condition 2

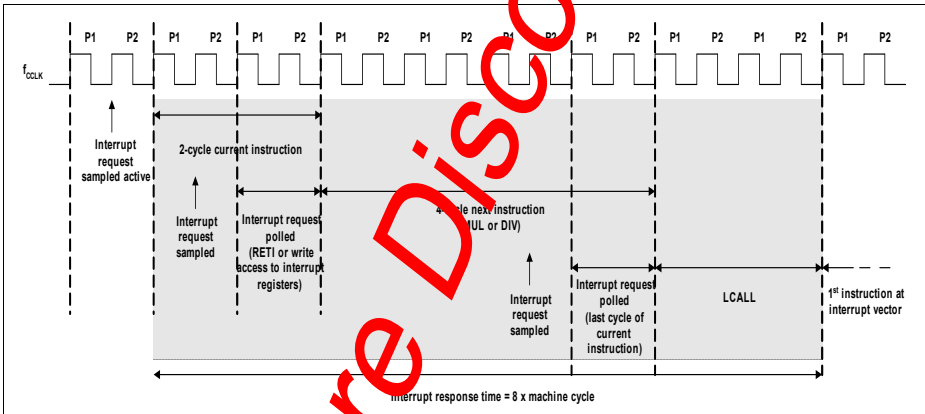


Figure 13-37 Interrupt Response Time for Condition 3

Thus in a single interrupt system, the response time is always more than three machine cycles and less than nine machine cycles (wait states are not considered). When considering wait states, the interrupt response time will be extended depending on the user instructions (also the hardware generated LCALL) being executed during the interrupt response time (shaded region in [Figure 13-36](#) and [Figure 13-37](#)).

Feature Discontinued

13.7.5 Registers Description

Interrupt Special Function Registers or bits are used for interrupt configuration such as node enable, external interrupt control, interrupt flags and interrupt priority setting.

Table 13-24 lists the SFRs and corresponding address.

Table 13-24 Register Map

Address	Register
RMAP = 0 or 1	
A8 _H	IEN0
E8 _H	IEN1
B8 _H	IP
B9 _H	IPH
F8 _H	IP1
F9 _H	IPH1
88 _H	TC0CN
98 _H	SC0CN
RMAP = 0, SCU Page = 0	
F2 _H	IRCON0
F3 _H	IRCON1
F4 _H	IRCON2
F5 _H	NMISR
F6 _H	NMICON
F7 _H	EXICON0
FA _H	EXICON1
FB _H	EXICON2
FC _H	EXICON3

Feature Discontinued

13.7.5.1 Interrupt Node Enable Registers

Each interrupt node can be individually enabled or disabled by setting or clearing the corresponding bit in the interrupt enable registers IEN0 and IEN1. Register IEN0 also contains the global interrupt masking bit (EA), which can be cleared to block all pending interrupt requests at once.

The NMI interrupt vector is shared by a number of sources, each of which can be enabled or disabled individually via register NMICON.

After reset, the enable bits in IEN0, IEN1 and NMICON are cleared to 0. This implies that all interrupt nodes are disabled by default.

IEN0

Interrupt Enable Register 0

 (A8_H)

 Reset Value: 00_H

RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
EA	0	ET2	ES	ET1	EX1	ET0	EX0
rw	r	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
EX0	0	rw	Interrupt Node XINTR0 Enable 0 _B XINTR0 is disabled 1 _B XINTR0 is enabled
ET0	1	rw	Interrupt Node XINTR1 Enable 0 _B XINTR1 is disabled 1 _B XINTR1 is enabled
EX1	2	rw	Interrupt Node XINTR2 Enable 0 _B XINTR2 is disabled 1 _B XINTR2 is enabled
ET1	3	rw	Interrupt Node XINTR3 Enable 0 _B XINTR3 is disabled 1 _B XINTR3 is enabled
ES	4	rw	Interrupt Node XINTR4 Enable 0 _B XINTR4 is disabled 1 _B XINTR4 is enabled
ET2	5	rw	Interrupt Node XINTR5 Enable 0 _B XINTR5 is disabled 1 _B XINTR5 is enabled

Field	Bits	Type	Description
EA	7	rw	Global Interrupt Mask 0 _B All pending interrupt requests (except NMI) are blocked from the core. 1 _B Pending interrupt requests are not blocked from the core.
0	6	r	Reserved Returns 0 if read; should be written with 0.

IEN1
Interrupt Enable Register 1
RMAP: X, PAGE: X
(E8_H)
Reset Value: 00_H

7	6	5	4	3	2	1	0
ECCIP3	ECCIP2	ECCIP1	ECCIP0	EXM	EX2	ESSC	ECAN
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ECAN	0	rw	Interrupt Node XINTR6 Enable 0 _B XINTR6 is disabled 1 _B XINTR6 is enabled
ESSC	1	rw	Interrupt Node XINTR7 Enable 0 _B XINTR7 is disabled 1 _B XINTR7 is enabled
EX2	2	rw	Interrupt Node XINTR8 Enable 0 _B XINTR8 is disabled 1 _B XINTR8 is enabled
EXM	3	rw	Interrupt Node XINTR9 Enable 0 _B XINTR9 is disabled 1 _B XINTR9 is enabled
ECCIP0	4	rw	Interrupt Node XINTR10 Enable 0 _B XINTR10 is disabled 1 _B XINTR10 is enabled
ECCIP1	5	rw	Interrupt Node XINTR11 Enable 0 _B XINTR11 is disabled 1 _B XINTR11 is enabled

Field	Bits	Type	Description
ECCIP2	6	rw	Interrupt Node XINTR12 Enable 0 _B XINTR12 is disabled 1 _B XINTR12 is enabled
ECCIP3	7	rw	Interrupt Node XINTR13 Enable 0 _B XINTR13 is disabled 1 _B XINTR13 is enabled

NMICON
NMI Control Register (F6_H) **Reset Value: 00_H**
RMAP: 0, PAGE: 0

	7	6	5	4	3	2	1	0
SCRINTTC	0	NMI-OSCCLK	NMI-MAINC	NMI-OCDS	NMI-EXTNMI	NMI-RAMECC	NMIWDT	
rw	r	rw	rw	rw	rw	rw	rw	

Field	Bits	Type	Description
NMIWDT	0	rw	Watchdog Timer NMI Enable 0 _B WDT NMI is disabled. 1 _B WDT NMI is enabled.
NMIRAMECC	1	rw	RAM Double bit ECC Error NMI Enable 0 _B RAM double bit ECC error NMI is disabled. 1 _B RAM double bit ECC error NMI is enabled.
NMIEXTNMI	2	rw	External NMI (via EXTNMI pin) NMI Enable 0 _B External NMI (via EXTNMI pin) is disabled. 1 _B External NMI (via EXTNMI pin) is enabled. <i>Note: It is recommended to clear the NMISR.FNMIEXTNMI flag before enabling External NMI.</i>
NMIOCDs	3	rw	OCDS NMI Enable 0 _B OCDS NMI is disabled. 1 _B OCDS NMI is enabled.
NMIMAINC	4	rw	Main Controller NMI Enable 0 _B Main Controller NMI is disabled. 1 _B Main Controller NMI is enabled.

Field	Bits	Type	Description
NMIOSCCLK	5	rw	Failure of 100 MHz oscillator clock NMI Enable 0 _B Failure of 100 MHz oscillator clock NMI is disabled. 1 _B Failure of 100 MHz oscillator clock NMI is enabled.
SCRINTTC	7	rwh	CPUx interrupt Enable Trigger an interrupt to CPUx. This bit will be reset automatically to 0. 0 _B No interrupt event is triggered. 1 _B An CPUx interrupt is triggered.
0	6	r	Reserved Returns 0 if read; should be written with 0.

13.7.5.2 External Interrupt Control Registers

The 16 external interrupts, EXT_INT[15:0], are driven into the TC2x_SCR from the ports. The external interrupt pin assignment for TC2x_SCR is shown in [Table 13-25](#). External interrupts can be positive, negative or double edge triggered. Registers EXICON0 and EXICON1 specify the active edge for the external interrupt. Among the external interrupts, external interrupt 0 and external interrupt 1 can be selected to bypass edge detection in the SCU, for direct feed through to the core. This signal to the core can be further programmed to either low-level or negative transition activated, by the bits IT0 and IT1 in the TCON register. However, for edge detection in SCU, TCON.IT0/1 must be set to falling edge triggered. An active edge event detected in SCU will generate internally two CCLK cycle low pulse for detection by core.

If the external interrupt is positive (negative) edge triggered, the external source must hold the request pin low (high) for at least one CCLK cycle, and then hold it high (low) for at least one CCLK cycle to ensure that the transition is recognized. If edge detection is bypassed for external interrupt 0 and external interrupt 1, the external source must hold the request pin "high" or "low" for at least two CCLK cycles.

External interrupts 3 to 15 share between 4 interrupt nodes. Therefore in addition to the corresponding interrupt logic enable, each external interrupt 3 to 15 may be disabled individually, and are disabled by default after reset.

Table 13-25 External interrupt [15:0] Pin Functions and Selection

Pin	Function	Description	Selected By
P02.0	EXINT0A	External Interrupt Input 0	MODPISEL1.EXINT0IS = 00 _B
P02.4	EXINT0B		MODPISEL1.EXINT0IS = 01 _B
P00.4	EXINT0C		MODPISEL1.EXINT0IS = 10 _B
ESR0 ¹⁾	EXINT0D		MODPISEL1.EXINT0IS = 11 _B
P02.1	EXINT1A	External Interrupt Input 1	MODPISEL1.EXINT1IS = 00 _B
P02.5	EXINT1B		MODPISEL1.EXINT1IS = 01 _B
P00.5	EXINT1C		MODPISEL1.EXINT1IS = 10 _B
ESR1 ²⁾	EXINT1D		MODPISEL1.EXINT1IS = 11 _B
P02.2	EXINT2A	External Interrupt Input 2	MODPISEL1.EXINT2IS = 00 _B
P02.6	EXINT2B		MODPISEL1.EXINT2IS = 01 _B
P00.6	EXINT2C		MODPISEL1.EXINT2IS = 10 _B
P14.1 ²⁾	EXINT2D		MODPISEL1.EXINT2IS = 11 _B
P02.0	EXINT3A	External Interrupt Input 3	MODPISEL2.T2CC0EXINT3IS = 00 _B
P00.0	EXINT3B		MODPISEL2.T2CC0EXINT3IS = 01 _B
P02.6	EXINT3C		MODPISEL2.T2CC0EXINT3IS = 10 _B
P02.1	EXINT4A	External Interrupt Input 4	MODPISEL2.T2CC1EXINT4IS = 00 _B
P00.1	EXINT4B		MODPISEL2.T2CC1EXINT4IS = 01 _B
P02.7	EXINT4C		MODPISEL2.T2CC1EXINT4IS = 10 _B
P02.2	EXINT5A	External Interrupt Input 5	MODPISEL2.T2CC2EXINT5IS = 00 _B
P00.2	EXINT5B		MODPISEL2.T2CC2EXINT5IS = 01 _B
P00.6	EXINT5C		MODPISEL2.T2CC2EXINT5IS = 10 _B
P02.3	EXINT6A	External Interrupt Input 6	MODPISEL2.T2CC3EXINT6IS = 00 _B
P00.3	EXINT6B		MODPISEL2.T2CC3EXINT6IS = 01 _B
P00.7	EXINT6C		MODPISEL2.T2CC3EXINT6IS = 10 _B
P02.4	EXINT7	External Interrupt Input 7	-
P02.5	EXINT8	External Interrupt Input 8	-
P02.6	EXINT9	External Interrupt Input 9	-
P02.7	EXINT10	External Interrupt Input 10	-
P00.4	EXINT11	External Interrupt Input 11	-
P00.5	EXINT12	External Interrupt Input 12	-

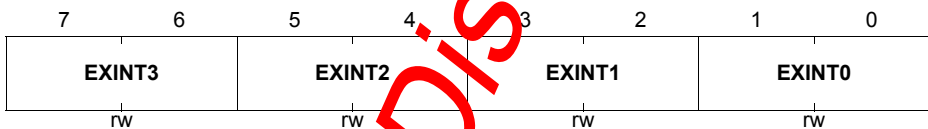
Table 13-25 External interrupt [15:0] Pin Functions and Selection (cont'd)

Pin	Function	Description	Selected By
P00.6	EXINT13	External Interrupt Input 13	-
P00.7	EXINT14	External Interrupt Input 14	-
P15.1 ²⁾	EXINT15	External Interrupt Input 15	-

1) The pad control of this pin is in the ESR sub-module (part of 32-bit SCU module). The input function of the pad must be enabled before this pin can be used as external interrupt input.

2) The pad control of these pins are in the 32-bit Ports module. The input function of the pad must be enabled before this pin can be used as external interrupt input.

Note: Several external interrupts support alternative input pin. When switching inputs, the active edge/level trigger select and the level of the associated pins should be considered to prevent unintentional interrupt generation.

EXICON0
External Interrupt Control Register 0 (F7_H)
Reset Value: F0_H
RMAP: 0, PAGE: 0


Field	Bits	Type	Description
EXINT0	[1:0]	rw	External Interrupt 0 Trigger Select
			00 _B Interrupt on falling edge.
			01 _B Interrupt on rising edge.
			10 _B Interrupt on both rising and falling edge.
			11 _B Bypass the edge detection in SCU. The input signal directly feeds to the core.
EXINT1	[3:2]	rw	External Interrupt 1 Trigger Select
			00 _B Interrupt on falling edge.
			01 _B Interrupt on rising edge.
			10 _B Interrupt on both rising and falling edge.
			11 _B Bypass the edge detection in SCU. The input signal directly feeds to the core.

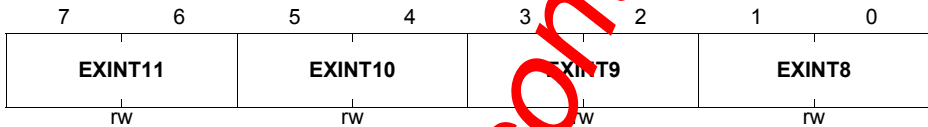
Field	Bits	Type	Description
EXINT2	[5:4]	rw	External Interrupt 2 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 2 is disabled.
EXINT3	[7:6]	rw	External Interrupt 3 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 3 is disabled.

EXICON1
External Interrupt Control Register 1 (FA_H)
Reset Value: FF_H
RMAP: 0, PAGE: 0

7	6	5	4	3	2	1	0
EXINT7		EXINT6		EXINT5		EXINT4	
rw		rw		rw		rw	

Field	Bits	Type	Description
EXINT4	[1:0]	rw	External Interrupt 4 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 4 is disabled.
EXINT5	[3:2]	rw	External Interrupt 5 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 5 is disabled.
EXINT6	[5:4]	rw	External Interrupt 6 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 6 is disabled.

Field	Bits	Type	Description
EXINT7	[7:6]	rw	External Interrupt 7 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 7 is disabled.

EXICON2
External Interrupt Control Register 2 (FB_H)
Reset Value: FF_H
RMAP: 0, PAGE: 0


Field	Bits	Type	Description
EXINT8	[1:0]	rw	External Interrupt 8 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 8 is disabled.
EXINT9	[3:2]	rw	External Interrupt 9 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 9 is disabled.
EXINT10	[5:4]	rw	External Interrupt 10 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 10 is disabled.
EXINT11	[7:6]	rw	External Interrupt 11 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 11 is disabled.

EXICON3
External Interrupt Control Register 3 (FC_H)
Reset Value: FF_H
RMAP: 0, PAGE: 0

7	6	5	4	3	2	1	0
EXINT15		EXINT14		EXINT13		EXINT12	
rw		rw		rw		rw	

Field	Bits	Type	Description
EXINT12	[1:0]	rw	External Interrupt 12 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 12 is disabled.
EXINT13	[3:2]	rw	External Interrupt 13 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 13 is disabled.
EXINT14	[5:4]	rw	External Interrupt 14 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 14 is disabled.
EXINT15	[7:6]	rw	External Interrupt 15 Trigger Select 00 _B Interrupt on falling edge. 01 _B Interrupt on rising edge. 10 _B Interrupt on both rising and falling edge. 11 _B External interrupt 15 is disabled.

TCON
Timer and Counter Control/Status Register(88_H)
Reset Value: 00_H
RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
rwh	rw	rwh	rw	rwh	rw	rwh	rw

Field	Bits	Type	Description
IT0	0	rw	External Interrupt 0 Level/Edge Trigger Control 0 _B Low level triggered external interrupt 0 is selected 1 _B Falling edge triggered external interrupt 0 is selected
IT1	2	rw	External Interrupt 1 Level/Edge Trigger Control 0 _B Low level triggered external interrupt 1 is selected 1 _B Falling edge triggered external interrupt 1 is selected

13.7.5.3 Interrupt Flag Registers

The interrupt flags for the different interrupt sources are located in several Special Function Registers. In case of software and hardware access to a flag bit at the same time, hardware will have higher priority.

IRCON0

Interrupt Request Register 0

(F2)

Reset Value: 00_H

RMAP: 0, PAGE: 0

7	6	5	4	3	2	1	0
EXINT7	EXINT6	EXINT5	EXINT4	EXINT3	EXINT2	0	
rw	rw	rw	rw	rw	rw	r	

Field	Bits	Type	Description
EXINTx (x = 2 - 7)	x	rw	Interrupt Flag for External Interrupt x This bit is set by hardware and can only be cleared by software. 0 _B Interrupt event has not occurred. 1 _B Interrupt event has occurred.

IRCON1

Interrupt Request Register 1

(F3_H)

Reset Value: 00_H

RMAP: 0, PAGE: 0

7	6	5	4	3	2	1	0	
		0				RIR	TIR	EIR
		r				rwh	rwh	rwh

Field	Bits	Type	Description
EIR	0	rwh	Error Interrupt Flag for SSC This bit is set by hardware and can only be cleared by software. 0 _B Interrupt event has not occurred. 1 _B Interrupt event has occurred.
TIR	1	rwh	Transmit Interrupt Flag for SSC This bit is set by hardware and can only be cleared by software. 0 _B Interrupt event has not occurred. 1 _B Interrupt event has occurred.
RIR	2	rwh	Receive Interrupt Flag for SSC This bit is set by hardware and can only be cleared by software. 0 _B Interrupt event has not occurred. 1 _B Interrupt event has occurred.
0	[7:3]	r	Reserved Returns 0 if read; should be written with 0.

IRCON2

Interrupt Request Register 2

(F4_H)

Reset Value: 00_H

RMAP: 0, PAGE: 0

7	6	5	4	3	2	1	0
EXINT15	EXINT14	EXINT13	EXINT12	EXINT11	EXINT10	EXINT9	EXINT8
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
EXINTx (x = 8 - 15)	x-8	rwh	Interrupt Flag for External Interrupt x This bit is set by hardware and can only be cleared by software. 0 _B Interrupt event has not occurred. 1 _B Interrupt event has occurred.

TCON
Timer and Counter Control/Status Register(88_H)
Reset Value: 00_H
RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
rwh	rw	rwh	rw	rwh	rw	rwh	rw

Field	Bits	Type	Description
IE0	1	rwh	External Interrupt 0 Flag Set by hardware when external interrupt 0 event is detected. Cleared by hardware when the processor vectors to interrupt routine. Can also be cleared by software.
IE1	3	rwh	External Interrupt 1 Flag Set by hardware when external interrupt 1 event is detected. Cleared by hardware when the processor vectors to interrupt routine. Can also be cleared by software.
TF0	5	rwh	Timer 0 Overflow Flag Set by hardware on Timer/Counter 0 overflow. Cleared by hardware when processor vectors to interrupt routine. Can also be cleared by software.
TF1	7	rwh	Timer 1 Overflow Flag Set by hardware on Timer/Counter 1 overflow. Cleared by hardware when processor vectors to interrupt routine. Can also be cleared by software.

SCON
UART Control/Status Register (98_H)
Reset Value: 00_H
RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
SM0	SM1	SM2	REN	TB8	RL8	TI	RI
rw	rw	rw	rw	rw	rwh	rwh	rwh

Field	Bits	Type	Description
RI	0	rwh	Serial Interface Receiver Interrupt Flag Set by hardware when a serial data byte has been received. Must be cleared by software.
TI	1	rwh	Serial Interface Transmitter Interrupt Flag Set by hardware at the end of a serial data transmission. Must be cleared by software.

NMISR
NMI Status Register (F5_H)
Reset Value: 00_H
RMAP: 0, PAGE: 0

7	6	5	4	3	2	1	0
0		FNMI OSCCLK	FNMI M/INC	FNMI OCDS	FNMI EXTNMI	FNMI RAMECC	FNMIWDT
r		rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
FNMIWDT	0	rwh	Watchdog Timer NMI Flag 0 _B No watchdog NMI has occurred. 1 _B WDT prewarning has occurred.
FNMI RAMECC	1	rwh	RAM Double bit ECC Error NMI Flag 0 _B No RAM ECC NMI has occurred. 1 _B RAM ECC NMI has occurred.

Field	Bits	Type	Description
FNMIEXTNMI	2	rwh	External NMI (via EXTNMI pin) Flag 0 _B No external NMI (via EXTNMI pin) has occurred. 1 _B External NMI (via EXTNMI pin) has occurred. <i>Note: It is recommended to clear the NMISR.FNMIEXTNMI flag before enabling External NMI.</i>
FNMIOCDS	3	rwh	OCDS NMI Flag 0 _B No OCDS NMI has occurred. 1 _B JTAG-receiving or user interrupt request in Monitor Mode has occurred.
FNMIMAINC	4	rwh	Main Controller NMI Flag 0 _B No Main controller NMI has occurred. 1 _B Main controller NMI has occurred.
FNMIOSCCLK	5	rwh	100 MHz Oscillator Clock NMI Flag 0 _B No 100 MHz oscillator NMI has occurred. 1 _B 100 MHz oscillator NMI has occurred.
0	7:6	r	Reserved Returns 0 if read; should be written with 0.

Note: NMISR register can only be cleared by software or reset to the default value after the Main Reset. Its value is retained on any other resets. This allows the cause of the previous NMI to be checked.

Feature Discontinued

13.7.5.4 Interrupt Priority Registers

Each interrupt node can be individually programmed to one of the four priority levels available. Two pairs of Interrupt Priority Registers are available to program the priority level of the each interrupt vector. The first pair of Interrupt Priority Registers are SFRs IP and IPH. The second pair of Interrupt Priority Registers are SFRs IP1 and IPH1.

The corresponding bits in each pair of Interrupt Priority Registers select one of the four priority levels as shown in [Table 13-26](#).

Table 13-26 Interrupt Priority Level Selection

IPH.x / IPH1.x	IP.x / IP1.x	Priority Level
0	0	Level 0 (lowest)
0	1	Level 1
1	0	Level 2
1	1	Level 3 (highest)

Note: NMI always has the highest priority (above Level 3); it does not use the priority level selection shown in [Table 13-26](#).

IP
Interrupt Priority Register (L3H)
RMAP: X, PAGE: X **Reset Value: 00_H**

7	6	5	4	3	2	1	0
0		PT2	PS	PT1	PX1	PT0	PX0
r		rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PX0	0	rw	Priority Level Low Bit for Interrupt Node XINTR0
PT0	1	rw	Priority Level Low Bit for Interrupt Node XINTR1
PX1	2	rw	Priority Level Low Bit for Interrupt Node XINTR2
PT1	3	rw	Priority Level Low Bit for Interrupt Node XINTR3
PS	4	rw	Priority Level Low Bit for Interrupt Node XINTR4
PT2	5	rw	Priority Level Low Bit for Interrupt Node XINTR5
0	[7:6]	r	Reserved Returns 0 if read; should be written with 0.

IPH
Interrupt Priority High Register
(B9_H)
Reset Value: 00_H
RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
0	PT2H	PSH	PT1H	PX1H	PT0H	PX0H	
r	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PX0H	0	rw	Priority Level High Bit for Interrupt Node XINTR0
PT0H	1	rw	Priority Level High Bit for Interrupt Node XINTR1
PX1H	2	rw	Priority Level High Bit for Interrupt Node XINTR2
PT1H	3	rw	Priority Level High Bit for Interrupt Node XINTR3
PSH	4	rw	Priority Level High Bit for Interrupt Node XINTR4
PT2H	5	rw	Priority Level High Bit for Interrupt Node XINTR5
0	[7:6]	r	Reserved Returns 0 if read; should be written with 0.

IP1
Interrupt Priority 1 Register
(F8_H)
Reset Value: 04_H
RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
PCCIP3	PCCIP2	PCCIP1	PCCIP0	PXM	PX2	PSSC	PCAN
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PCAN	0	rw	Priority Level Low Bit for Interrupt Node XINTR6
PSSC	1	rw	Priority Level Low Bit for Interrupt Node XINTR7
PX2	2	rw	Priority Level Low Bit for Interrupt Node XINTR8
PXM	3	rw	Priority Level Low Bit for Interrupt Node XINTR9
PCCIP0	4	rw	Priority Level Low Bit for Interrupt Node XINTR10
PCCIP1	5	rw	Priority Level Low Bit for Interrupt Node XINTR11

Field	Bits	Type	Description
PCCIP2	6	rw	Priority Level Low Bit for Interrupt Node XINTR12
PCCIP3	7	rw	Priority Level Low Bit for Interrupt Node XINTR13

IPH1

Interrupt Priority 1 High Register (F9_H)

Reset Value: 04_H

RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
PCCIP3H	PCCIP2H	PCCIP1H	PCCIP0H	PXMH	PX2H	PSSCH	PCANH
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PCANH	0	rw	Priority Level High Bit for Interrupt Node XINTR6
PSSCH	1	rw	Priority Level High Bit for Interrupt Node XINTR7
PX2H	2	rw	Priority Level High Bit for Interrupt Node XINTR8
PXMH	3	rw	Priority Level High Bit for Interrupt Node XINTR9
PCCIP0H	4	rw	Priority Level High Bit for Interrupt Node XINTR10
PCCIP1H	5	rw	Priority Level High Bit for Interrupt Node XINTR11
PCCIP2H	6	rw	Priority Level High Bit for Interrupt Node XINTR12
PCCIP3H	7	rw	Priority Level High Bit for Interrupt Node XINTR13

13.7.6 Interrupt Flag Overview

The interrupt events have interrupt flags that are located in different SFRs. [Table 13-27](#) shows the corresponding SFR to which each interrupt flag belongs. Detailed information on the interrupt flags is provided in the respective peripheral chapters.

Table 13-27 Location of the Interrupt Flags

Interrupt Event	Interrupt Flag	SFR
Timer 0 Overflow	TF0	TCON
Timer 1 Overflow	TF1	TCON

Table 13-27 Location of the Interrupt Flags (cont'd)

Interrupt Event	Interrupt Flag	SFR
Timer2 Overflow	TF2	T2_T2CON
Timer2 External Event	EXF2	T2_T2CON
UART Receive	RI	SCON
UART Transmit	TI	SCON
LIN End of Synch Byte	EOFSYN	LINST
LIN Synch Byte Error	ERRSYN	LINST
External Interrupt 0	IE0	TCON
External Interrupt 1	IE1	TCON
External Interrupt 2	EXINT2	IRCON0
External Interrupt 3	EXINT3	IRCON0
External Interrupt 4	EXINT4	IRCON0
External Interrupt 5	EXINT5	IRCON0
External Interrupt 6	EXINT6	IRCON0
External Interrupt 7	EXINT7	IRCON0
External Interrupt 8	EXINT8	IRCON2
External Interrupt 9	EXINT9	IRCON2
External Interrupt 10	EXINT10	IRCON2
External Interrupt 11	EXINT11	IRCON2
External Interrupt 12	EXINT12	IRCON2
External Interrupt 13	EXINT13	IRCON2
External Interrupt 14	EXINT14	IRCON2
External Interrupt 15	EXINT15	IRCON2
T2CCU CCT Overflow	CCTOVF	T2CCU_CCTCON
SSC Error	EIR	IRCON1
SSC Transmit	TIR	IRCON1
SSC Receive	RIR	IRCON1
WCAN detection interrupt	WUF	STAT_DAT1
WCAN error overflow interrupt	SYSEERR	STAT_DAT0
WCAN oscillator counter run interrupt	OSCCNTRUN	STAT_DAT0
WCAN timeout interrupt	CANTO	STAT_DAT0

Table 13-27 Location of the Interrupt Flags (cont'd)

Interrupt Event	Interrupt Flag	SFR
RTC compare/wakeup interrupt	CFRTC	RTC_RTCON
Watchdog Timer NMI	FNMIWDT	NMISR
RAM Double bit ECC Error NMI	FNMIRAMECC	NMISR
External NMI (via EXTNMI pin) NMI	FNMIEXTNMI	NMISR
OCDS NMI	FNMIOCDS	NMISR
Main Controller NMI	FNMIMCNC	NMISR
Failure of 100 MHz oscillator clock NMI	FNMIOCCCLK	NMISR

Feature Discontinued

Feature Discontinued

13.8 Debug System

This chapter describes the XC800 debug system, in particular focus on the OCDS unit which provides the hardware to support debug functionality.

13.8.1 Overview

The debug system comprises of the On-Chip Debug Support (OCDS) unit and Debug-Monitor in ROM which provides basic debug functionality for XC800-based systems, controlled directly by an external tool via debug interface pin(s).

13.8.1.1 Features

The main debug functionality supported are:

- Set breakpoints on instruction address and on address range within the Program Memory
- Set breakpoints on Internal RAM address range
- Support unlimited software breakpoints in Flash/RAM code region
- Process external breaks via debug interface
- Step through the program code
- User handling of OCDS NMI in case of ROM read/write breakpoint

13.8.1.2 Components of the Debug System

The components of the debug system are briefly described here.

Debug Interface

The debug interface allows to access the device, such as for data read or write. The debug interface supported:

- 2-pin Device Access Port (DAP)
- Single-pin DAP (SPD)

The DAP interface refers to the dual-pin Device Access Port standardized for the Infineon microcontrollers. It offers high noise immunity and robustness, and utilizes only two pins DAP0 (clock), DAP1 (serial data in/out).

The SPD interface refers to the single-pin Device Access Port standardized for the Infineon microcontrollers. It utilizes only one pin DAP1 for serial data in/out.

Monitor Program

Part of the Boot ROM is occupied by the OCDS Monitor program (or OCDS firmware). On the command from the debugger, the Monitor program executes the actual instructions for accessing the addressable locations such as memories, sfrs of the device.

On-Chip Debug System Unit (OCDS)

The OCDS hardware is the core of the debug system, controlling the debugging with interfaces to the XC800 CPU.

13.8.2 Product Specific Information

This section provides product specific information relevant to the OCDS.

13.8.2.1 Pinning

[Table 13-28](#) describes the OCDS pin function.

Table 13-28 OCDS Pin Functions

Pin	Function	Description
P02.2	DAP0_1	Dual-pin debug access port: clock
P02.6	DAP0_0	
P02.3	DAP1/SPD_1	Single-pin debug access port: data
P02.7	DAP1/SPD_0	

The pin will be set up via hardware once the OCDS mode is selected.

13.8.2.2 Clocking Configuration

The OCDS runs at the peripheral frequency, PCLK.

13.8.2.3 Interrupt Events and Assignment

Instead of entering Monitor Mode on a break event, it can be configured such that OCDS NMI is vectored to. In this configuration, the action on OCDS NMI is fully under user software control. Details of this usage is described in later sections.

[Table 13-29](#) lists the interrupt event sources from the OCDS, and the corresponding event interrupt enable bit and flag bit.

Table 13-29 OCDS Interrupt Events

Event	Event Interrupt Enable Bit	Event Flag Bit
IRAM read event	MMICR.NMIRRE	MMICR.FNMIRR
IRAM write event	MMICR.NMIRWE	MMICR.FNMIRW

shows the interrupt node assignment for each OCDS interrupt source.

13.8.2.4 Debug Suspend Control

It is enabled for selected modules to suspend operation when Monitor Mode becomes active. The module functions that can be enabled for debug suspend are:

- Watchdog timer
- Timer 2
- CCT timer
- RTC timer

By debug suspend, these module functions are frozen during the duration of the device in Monitor Mode, e.g. timers stop counting. Register MODSUSP is used to control the debug suspend function.

MODSUSP

Module Suspend Control Register (FB_H)

Reset Value: 01_H

RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
	0			CCTSUSP	T2SUSP	RTCSUSP	WDTSUSP
	r			rw	rw	rw	rw

Field	Bits	Type	Description
WDTSUSP	0	rw	SCC Watchdog Timer Debug Suspend Bit 0 _B WDT will not be suspended. 1 _B WDT will be suspended.
RTCSUSP	1	rw	Real-time Clock Debug Suspend Bit 0 _B Real-time clock will not be suspended. 1 _B Real-time clock will be suspended.
T2SUSP	2	rw	Timer2 Debug Suspend Bit 0 _B Timer2 will not be suspended. 1 _B Timer2 will be suspended.
CCTSUSP	3	rw	Compare/Capture Timer Debug Suspend Bit 0 _B CCT will not be suspended. 1 _B CCT will be suspended.
0	[7:4]	r	Reserved Returns 0 if read; should be written with 0.

13.8.2.5 JTAG ID

This is a read-only register located inside the SCR JTAG module, and it is used by the tool to identify the target connected via the SCR DAP/SPD interface. Its content is shifted out when **INSTRUCTION** register contains the **IDCODE** command (opcode 04_H). After power on and any JTAG reset, the **BYPASS** instruction (opcode FF_H) will be latched onto the **INSTRUCTION** register.

The JTAG ID for the TC2x_SCR devices is 101EC083_H.

13.8.3 Functional Overview of the Debug System

XC800 debug-operation is based on close interaction between the OCDS-hardware and the Monitor program in ROM.

These operations are elaborated in the following.

13.8.3.1 Recognizing Debug-events

The OCDS hardware takes care to startup properly the Monitor, in case some **Debug-event** happens.

The debug events (breakpoints) supported are described in following.

Hardware Breakpoints

Up to 4 hardware breakpoints can be set, causing a break in case of

- code is fetched from a defined Program Memory (IP) address - the break is set just at one instruction from the user software
- code is fetched from a defined Program Memory (IP) address-range - the break is set over a number of consecutive instructions from the user software
- Read or Write (selectable) is performed at a defined Internal RAM address - the break is caused by moving data from/to a memory location

For setting of breakpoint on instruction address,HWBPx defines the 16-bit address. For setting of breakpoint on IRAM address, HWBP2/3L and HWBP2/3H define the 8-bit IRAM address range.

The configurations supported are:

- Breakpoint 0
- Breakpoint 1
 - Two equal breakpoints on Instruction Address = HWBP0 and Instruction Address = HWBP1 or
 - One range breakpoint on HWBP0 <= Instruction Address <= HWBP1
- Breakpoint 2
 - One equal breakpoint on Instruction Address = HWBP2, or

- One range breakpoint on HWBP2L <= IRAM Read Address <= HWBP2H
- Breakpoint 3
 - One equal breakpoint on Instruction Address = HWBP3, or
 - One range breakpoint on HWBP3L <= IRAM Write Address <= HWBP3H

Setting both values for a range breakpoint to the same address leads to generation of an equal breakpoint.

Software Breakpoints

These are implemented by storing into the code XC800-specific TRAP instruction (not 8051-standard), while at the same time TRAP_EN bit within the Extended Operation (EO) register is also set to 1.

Upon fetching a TRAP instruction, a breakpoint is generated and the relevant Break Action is taken.

The software breakpoints are in fact similar in behavior to the equal breakpoints on Instruction address, except that they are raised by a program code instead of specialized (compare) logic.

An unlimited number of software breakpoints can be set by replacing the original instruction codes into the user program. Of course this is possible only at addresses, where RAM/Flash is implemented.

External Breaks

External breaks could be:

- Requested via the debug interface by OCDS command: An external Debugger can activate the Monitor and interrupt a user program, running on the target XC800.

13.8.3.2 Activating the Monitor Program

Once a Debug-event is identified and the OCDS-registers are properly programmed, the Monitor program (in ROM) is started as a primary reaction.

The OCDS hardware ensures that the Monitor is always safely started, and fully independent of the current system status at the moment when the debug action is taken. Also, interrupt requests externally raised during Monitor-entry will not disturb the Monitor firmware functioning.

Once started, the Monitor runs with its own stack- and data- memory, which guarantees that all of the core and memory resources will be found untouched when returning control back to the user program. Therefore the OCDS-debugging is fully non-destructive.

The functions of the Monitor include:

- Communication with an external Debugger via the debug interface
- Read/write access to arbitrary memory locations and Special Function Registers (SFRs), including the Instruction Pointer and password-protected bits

- Configuring OCDS and setting/removing breakpoints
- Executing a single instruction (step-mode)

Note: Details of the Monitor program functionality is not covered in this documentation.

13.8.3.3 Debug Suspend Control

Next to the basic debug functionality - setting breakpoints and halting the execution of user software - OCDS supports also for module suspending during debugging.

As long as the device is in monitor mode (i.e. while the user software is not running but in break) and if debug suspend functionality is generally enabled by on-chip software, modules or functions can be suspended in this duration.

This feature could be quite useful, especially regarding the Watchdog Timer: it prevents unintentional WDT-resets during the debug session. Usually debug suspend is provided for other timer-modules. Stopping counters provide for a complete “freeze” of the device-status during a break.

Generally, the debug suspend control bits (global enable in OCDS and individual selections in SCU) are disabled after reset, i.e. by default no module will be suspended upon a break. Normally for debugging the device will be started in OCDS mode and then the monitor will be invoked before to start any user code. The debugger should provide for user configuration of suspend-controls.

13.8.3.4 Running the Monitor

The Monitor runs with its own stack- and data- work space, therefore the complete status of the core is saved and later - restored before returning to the user code.

Once the Monitor is running, it can access for read and write all of the system resources, and data can be communicated with the Debugger.

13.8.3.5 Returning to the User Program

When a return to User-Program is requested, the Monitor sets core-registers including the Instruction Pointer (IP) as required. This can be either the same status as at the Breakpoint, or including some changes done during the Break by a host debugger. Then a jump to the location pointed by IP is performed and the target system returns to User Mode.

13.8.3.6 Single Step Execution

For this functionality OCDS operates at first as when Returning to the User Program, but also a dedicated flag is set into a control-register (MMCR.MSTEP). This way, just one user-instruction is executed before re-entering the Monitor again.

13.8.4 JTAG Module

In XC800 implementation for JTAG:

- **IO client 0** is assigned to the XC800 OCDS.

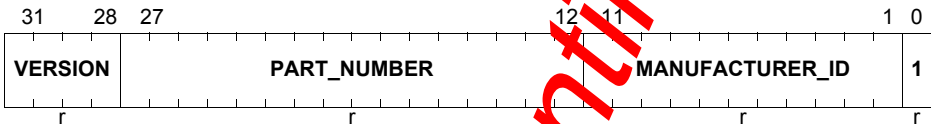
Note: Following from the last assignment, the Debugger does not need to write into IOPATH-upon reset this register is cleared and Client 0 is selected by default.

Feature Discontinued

13.8.4.1 JTAG ID Register

This is a read-only register, used to recognize the device(s) connected to the JTAG/DAP/SPD interface. Its content is shifted out when the **INSTRUCTION** register contains the **IDCODE** command (opcode 04_H). After power on and any JTAG reset, the **BYPASS** instruction (opcode FF_H) will be latched onto the **INSTRUCTION** register.

ID JTAG ID Register



Field	Bits	Typ	Description
1	0	r	Always 1
MANUFACTURER_ID	[11:1]	r	Infineon's Manufacturer ID
PART_NUMBER	[27:12]	r	Part number of the chip
VERSION	[31:28]	r	Version of the chip

The **MANUFACTURER_ID** for Infineon is 041_H, therefore the three LS nibbles of the JTAG ID Register are 083_H.

The rest - **PART_NUMBER** and **VERSION**-fields are chip-specific.

13.8.5 Interface Block

The Debug-communication channel - JTAG/DAP/SPD interface - goes through OCDS as a 16-bit shift register **MMSHR** (refer to [Figure 13-38](#)) and it is controlled by a few signals coming from the JTAG Module:

- **jm_sel_io_i**
 a select signal; any operation around the JTAG-interface within the OCDS is only possible while it is activated
- **jm_capture_ioreg_i**
 while having high level at this input, on the rising edge of tck_i:
 - four status bits from the OCDS control registers are stored into the lowest part of **MMSHR** (**MBCON**, **MMODE**, **RRF** and **TRF**) and one (**COMRST**) - as bit 12 into **MMSHR**
 - the current status of **RRF** flag which is stored into **MMSHR** is also saved into the shadow-bit **RRF_s**
 - data transmit register **MMTR** is stored into the middle of the shift-register **MMSHR**

- **jm_shift_dr_i**

as long as this signal is active, the input data from JTAG Module (tdo_i) is shifted into the MMSHR on any rising edge of tck_i (LSB first), while the data out of MMSHR goes back to JTAG Module through $jm_tdo_io0_o$

Four status bits: MBCON, MMODE, RRF and TRF are the first to be sent out via JTAG during any communication session with a Debugger, followed by the 8 bits of data from MMTR and the last status bit (within the current version of OCDS) - COMRST.

MMSHR is reset by the OCDS-reset signal.

- **jm_update_dr_i**

with this signal activated after a shift-sequence, the 16 bits last shifted into MMSHR are stored into respective OCDS-registers.

For potential update of control signals/data, there are rules defined to assure correct and reliable functionality, as follows:

1. any update is only possible, if at least one out of the five control signal requests - JENA_r, RRF_r, MMODE_r, MBCON_r and COMRST_r is active.
Therefore, sending only '0's the Debugger can check the current OCDS-status, without any further influence.
2. some of the control signals requests are meaningful only when send together with $RRF_r=0$ AND $Data=A5_H$.
The reason for this is that some functions within the OCDS are requested as dedicated commands through the Debug-interface, not combined with other (data-) transfers. This refers to:
 - a) setting the OCDS-flag for active JTAG
JENA_r - request to update JENA within MMCR2
 - b) requesting a Monitor Mode Entry
MMODE_r - request to enter Monitor Mode
Note, that MMODE_r does not set directly the MMODE flag!
 - c) requesting a Communication Restart
COMRST_r - request to set COMRST within MMICR, which causes the Monitor to restart the communication protocol.
3. The middle 8 bits from the MMSHR are stored into the data receive register MMRR, only when $RRF_r=1$ AND $RRF_s=0$.
In case the data is stored, RRF is also updated to 1.
This avoids overwriting the data within MMRR, till the Monitor reads it which will also automatically clear MMCR.RRF. Note, that RRF can not be reset via the JTAG but only by OCDS hardware.
4. TRF flag is checked by the Debugger to determine when a new data has been transmitted by the Target (XC800). This bit is fully controlled by OCDS-hardware:
 - a) set when the Monitor writes a new Byte for transfer into the transmit register MMTR
 - b) reset when MMTR is stored into MMSHR.

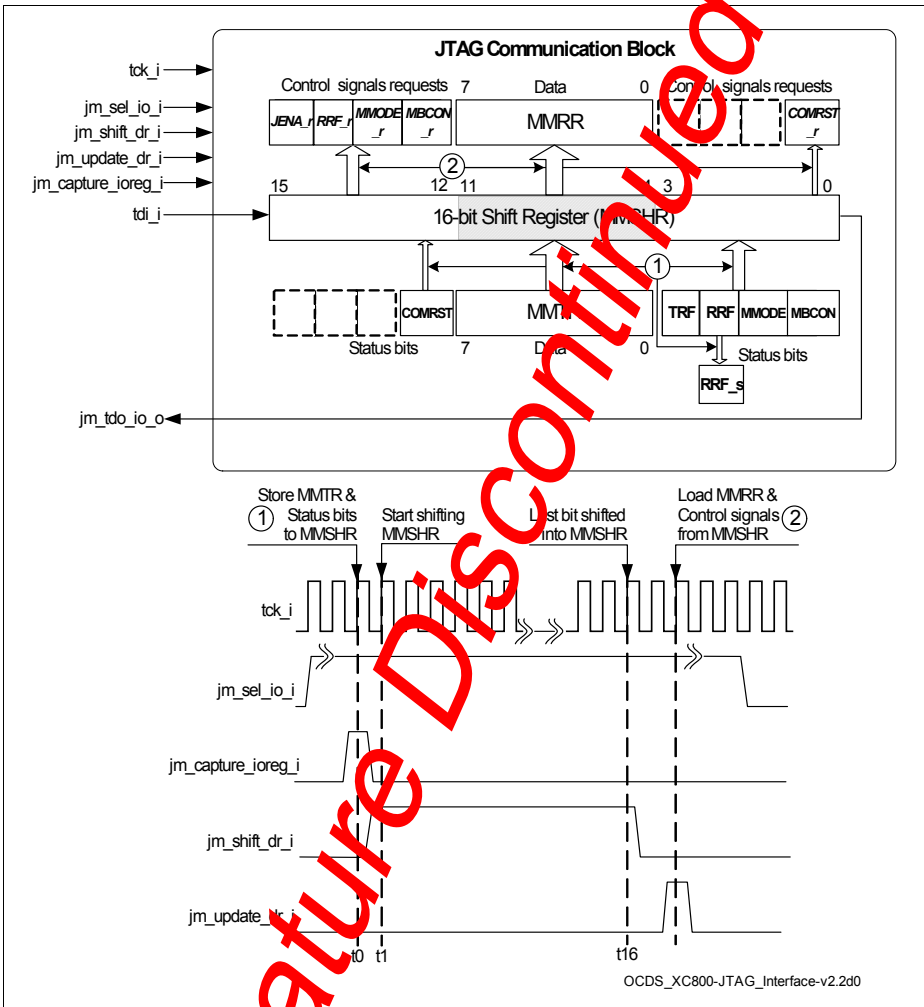


Figure 13-38 JTAG Communication: Structure and Signals

Attention: To keep backwards compatibility, the Debugger must always send only zero to the unused control bits - currently to MMSHR[3:1].

13.8.6 Program Execution Control

The basic feature of the OCDS module is to switch the code executed between user software and the Monitor program.

13.8.6.1 Entering Monitor Mode

The OCDS can enter Monitor Mode **only** when MMODE=0 and MBCON=0 within the MMCR2 register.

Main steps performed upon Monitor Entry:

1. some Debug-event happens, recognized by OCDS
2. OCDS activates a request to the XC800 core
At the same time:
 - a) MMCR.MRAMS is set to 1, to select Monitor RAM
 - b) MMCR2.MMODE is set to 1, to mark entering the Monitor Mode
3. the core starts fetching
4. MMCR2.MMEP is set to '1', signaling the Monitor is entered.

Functions supported by the first Monitor-instructions:

- save current (user) PSW and Accumulator into Monitor RAM, which is already selected by MMODE & MRAMS
- save current SYSCON0 into Monitor RAM, to prevent the user value of RMAP control bit
- set SYSCON0.RMAP to '0' allowing access to standard SFRs

Note: The core registers like SP, IE, PSW and MEX1 are accessed in both SFR spaces.

- save the Page of SCU-SFRs currently in use
- save user IE register into Monitor RAM and disable all maskable interrupts
- save current (user) Stack Pointer into Monitor RAM
- initiate the Stack to a dedicated area into Monitor RAM
- select (in Manual Page Mode) Page 0 from SCU-SFRs

Note: In case and where applicable furthermore inside the Monitor relevant PAGE-bitfields are properly controlled to access all Pages of SFRs.

- store the Break-address into dedicated Monitor RAM locations.

The Break-address will be used furthermore to determine the Return-address to User Program - if not explicitly set by the Debugger.

Note:

Feature Discontinued

13.8.6.2 Exiting Monitor Mode

For exiting Monitor Mode and returning to the User Program:

1. the Monitor sets MMCR.MEXIT to 1, while at the same time MMCR2.MMODE=1
OCDS activates a request to the XC800 core
2. the core enters Debug Mode executing the last Monitor instructions
3. MMCR2.MMODE and MMCR.MRAMS are reset by OCDS
4. MMCR.MEXIT is reset by OCDS
5. the core runs in normal mode, the system is back in User Mode.

The last Monitor instructions are doing the following.

- restore the basic core-status as saved when Entering Monitor Mode
 - interrupt enable register within the core
 - user Page of SCU-SFRs
 - SYSCON0 register (RMAP is here)
 - user Accumulator, PSW and Stack Pointer
- prepare the user MEX1 register at the top of Memory Extension Stack (has effect only in devices with Memory Extension)
- prepare the return-address on the top of stack

Note: Have in mind, the stack is still in Monitor RAM, with A7 and A6 ignored for addressing.

Feature Discontinued

13.8.6.3 Single Step Execution

In case Single Step Execution is requested, the Monitor sets MMCR.MSTEP=1 before to proceed with Exiting Monitor Mode. After executing the steps above described, the further OCDS functionality is as follows:

- the core ends current instruction - the first from user-program
- the core signals Entering Debug Mode
this is recognized as a new Debug Event - see step 1 from the sequence when Entering Monitor Mode ([Section 13.8.6.1](#)).

When the monitor mode is exited for a single step execution (with MMCR.MSTEP=1), the recognition of all breakpoints is temporary blocked by OCDS hardware to assure a correct stepping.

After (re-)entering Monitor mode, MMCR.MSTEP must be checked by the firmware to identify the single-stepping performed, and in case the return-address must be calculated accordingly (refer to [Section 13.8.9.1](#)).

Note: Single Stepping is not possible inside a MMI-servicing routine. For more information refer to [Section 13.8.11](#).

13.8.7 Breakpoint Generation Module

Breakpoint generation is only possible in Monitor Mode when MMODE=0 within the MMCR2 register.

Attention: To be able to debug via Monitor Mode, the application software should not change the TRAP_EN bit within Extended Operation (EO) register.

Attention: Upon MBCON=1 the Monitor Mode is not entered as well as no Break Flags are set into MMSR register.

Breakpoints can be classified into three types:

- Break Before Make
The break happens just before the break instruction (i.e. the instruction causing the break) is executed. Therefore, the break instruction itself will be the next instruction from the user program flow but executed only after the relevant debug action has been taken.
- Break After Make
The break happens immediately after the instruction causing it has been executed. Therefore, the break instruction itself has already been executed when the relevant debug action is taken.
- Break Now
The events of this type are asynchronous to the code execution inside the TC2x_SCR and there is no "instruction causing the debug event" in this case. The debug action is performed by OCDS "as soon as possible" once the debug event is raised.

13.8.7.1 Generating Hardware Breakpoints

This block is built around a set of comparators, dedicated to recognize two types of hardware breakpoints:

Breakpoints on Instruction Address

These IP breakpoints are generated, only when the BP-address is matched for the first byte of an instruction - i.e. just for the opcode. In case of 2- and 3- byte instructions, the break will not be generated at all for addresses of the second and third instruction-bytes.

The IP breakpoints are of Break Before Make type, therefore the instruction at the breakpoint is executed only after the proper debug action is taken.

The processing goes as follows:

1. break conditions are met
2. forces a TRAP-code $A5_H$ to be supplied to the Core, instead of a Byte currently addressed within the Program Memory

Address Comparators, Control bits and Flags

Below the comparators used for IP Breakpoints are described, together with control bits HWBxC and respective flags HWBxF:

- **CMPO**

A 16-bit comparator between the two **HWBP0** registers (A-side) and the 16-bit Program Memory Address Bus (B-side).

Two output signals are generated:

- A=B - sets HWB0F if HWB0C=01 (break on IP= **HWBP0**)
- A<B - see below.

- **CMP1**

A 16-bit comparator between the two **HWBP1** registers (A-side) and the 16-bit Program Memory Address Bus (B-side).

Two output signals are generated:

- A=B - sets HWB1F if HWB1C=1 (break on IP= **HWBP1**)
- A>=B when activated together with A<=B from CMP0 - sets HWB0F if HWB0C=11 (break on **HWBP0** =<IP=> **HWBP1**)

- **CMPIP2**

A 16-bit comparator between the two **HWBP2** registers (A-side) and the 16-bit Program Memory Address Bus (B-side).

One output signal is generated:

- A=B - sets HWB2F if HWB2C=10 (break on IP = **HWBP2**)

- **CMPIP3**

A 16-bit comparator between the two **HWBP3** registers (A-side) and the 16-bit Program Memory Address Bus (B-side).

One output signal is generated:

- A=B - sets HWB3F if HWB3C=10 (break on IP = **HWBP3**)

Table 13-30 MMBPCR [2:0]: Configuring Hardware Breakpoints 0 and 1

HWB1C	HWB0C	Break Event	Flag to set
0	00	no	---
x	01	Program Memory Address = HWBP0	HWB0F
1	xx	Program Memory Address = HWBP1	HWB1F
0	11	HWBP0 =< Program Memory Address =< HWBP1	HWB0F

Breakpoints on Internal RAM Address

These Internal RAM (IRAM) Breakpoints are recognized primary by observing the Internal Data Memory Address Buses `SOURCE_A[7:0]` and `DESTIN_A[7:0]`, respectively for Breakpoints on Read and Write addresses respectively.

The IRAM breakpoints are of Break After Make type, therefore the proper debug action is taken immediately after the operation to the breakpoint address is performed.

The OCDS supports only range breakpoints on IRAM address.

The OCDS differentiates between a breakpoint on read and a breakpoint on write operation to the IRAM.

The conditions to have a Memory Access (IRAM) Breakpoint are:

- `SOURCE_A / DESTIN_A` matches some Compare Address `HWBPxx`, **and**
- the relevant control signals are active showing a valid Read/Write operation, **and**
- the respective control field `HWBxC` is properly programmed.

The exact processing is different in case of Read and Write Breakpoints.

Break on IRAM read

The functionality is as follows:

1. the read address `SOURCE_A` matches a configured Compare Address `HWBPxx` and the respective control field `HWBxC` is set to 1
2. the Preliminary Condition - having an address-match AND `NSOURCE_RE_SGNF=0` (active low) - is registered
3. the very last clock cycle (CLP2) of the instruction reading from IRAM is executed
 - a) the respective `HWBxF` flag is set within MMSR status-register
4. Follow on:
 - a) a TRAP-code `A5H` is supplied to the Core via `PROGDI[7:0]` instead of a Byte currently addressed within the Program Memory
 - b) the registration of the Preliminary Condition is cleared.
5. This is recognized by OCDS as a Break-event (**B**), and the processing continues as described in [Section 13.8.6.1](#).

Break on IRAM write

The functionality is as follows:

1. the write address DESTIN_A matches a configured Compare address HWBPxx and the respective control field HWBxC is set to 1
2. the Preliminary Condition - having an address-match AND !DESTIN_WE=0 (active low) - is registered
3. the very last clock cycle (CLP2) of the instruction accessing the IRAM is executed
 - a) the respective HWBxF flag is set within MMSR status-register
4. Follow on:
 - a) a TRAP-code A5_H is supplied to the Core via PROGDI[7:0] instead of a Byte currently addressed within the Program Memory
 - b) the registration of the Preliminary Condition is cleared.
5. This is recognized by OCDS as a Break-event (BR), and the processing continues as described in [Section 13.8.6.1](#).

Address Comparators, Control bits and Flags

Below the comparators used for Memory Access Breakpoints are described, together with control bits HWBxC and respective flags HWBxF:

- **CMPRL2, CMPRH2**

Two 8-bit comparators:

- CMPRL2 - between HWBP2L register (A-side) and the 8-bit IRAM Source Address Bus (B-side), generating $A < B$.
- CMPRH2 - between HWBP2H register (A-side) and the 8-bit IRAM Source Address Bus (B-side), generating $A = B$.

Then:

HWB2F is set if $HWB2C = 01$,
i.e. break on $HWBP2L = \text{IRAM Read Address} \leq HWBP2H$.

- **CMPWL3, CMPWH3**

Two 8-bit comparators:

- CMPWL3 - between HWBP3L register (A-side) and the 8-bit IRAM Destination Address Bus (B-side), generating $A < B$;
- CMPWH3 - between HWBP3H register (A-side) and the 8-bit IRAM Destination Address Bus (B-side), generating $A \geq B$. Then:

HWB3F is set if $HWB3C = 01$,
i.e. break on $HWBP3L \leq \text{IRAM Write Address} \leq HWBP3H$.

Tracing changes in Break Address

The mechanism to generate Hardware Breakpoints assures, that "immediately" upon a break-condition match, the XC800 core is forced to Debug Mode. The Program Counter freezes then at a value, related to the address of instruction that would have been fetched, had the core not been in Debug Mode. This PC-value upon Monitor Mode Entry

is saved as "Break Address" and used later for [Calculating the Return Address](#) (refer to [Section 13.8.9.1](#)).

In fact, a TRAP-opcode is provided to the core instead of the next instruction from the user software-flow. In case of an IP breakpoint, this TRAP replaces just the instruction at the breakpoint and therefore the address saved is clearly connected to the address of the Break itself.

The functionality of MMICR.DVECT is as follows:

1. a shadow register is reset at the end of every instruction
 Upon an IRAM breakpoint, the TRAP comes **next** after the instruction causing the Break (break-instruction). The break-instruction itself can be an branch-instruction, performing also an IRAM-access (like CJNE, DJNZ). If so, before entering Monitor Mode the PC will be already in case updated with the branch-address. Further, the Break Address will be saved and its value will reflect properly where from the user program must continue later on (Return Address). But from the address saved in such a situation, the address where the Break Event (IRAM access) has happened **can not** be determined.
 MMICR.DVECT bit supports recognition of such situations. The core activates this signal upon any discontinuity in the program flow - when a branch instruction is executed.
2. any activation of DebugVector immediately sets the shadow register
3. an IRAM Breakpoint is recognized - the Preliminary Condition is registered, a TRAP is forced to the core; then the MMICR.DVECT bit is set to 1.

When entered, the Monitor will determine the source of the Break (flags in MMICR), and in case of Breakpoints 2 and 3 (marked by HWB2F and HWB3F respectively) - check either the respective BP has been configured as IRAM (control bits HWB2C and HWB3C in MMBPCR). If yes - the Monitor will check DVECT next and relevant information is to be forwarded to the Debugger, concerning the Break Event source (IRAM read/write):

- when DVECT=0 - the Break Event has been caused by the instruction preceding the Return Address (calculated in accordance to [Section 13.8.9.1](#))
- when DVECT=1 - the instruction causing the Break Event can not be determined.

Note: After evaluating DVECT, the Monitor must take care to reset it before returning to User Mode.

Feature Discontinued

13.8.7.2 Processing External Breaks

Possible external break-requests are generated and processed as follows:

1. a request comes from one of the possible source(s)
 - a) Debugger-request via DAP SPD - a dedicated command has been sent, with **MMODE_r=1**, **RRF_r=0** and data-Byte=A5_H, while at the same time: **MMCR2.MMODE=0** and **MMCR2.ALTDI=0**
2. a Break request is activated high to the core
 - a) OCDS sets **MMSR.EXBF** flag upon any accepted external break request
3. during the next few SCLK cycles the core ends executing current instruction, the core enters Debug Mode and activates Debug Ack. This is recognized by OCDS as a Break-event, and the processing continues as described in [Section 13.8.6.1](#).

Note: External Breaks are not possible inside a NM₀ servicing routine.

For more information refer to [Section 13.8.1](#).

Feature Discontinued

13.8.7.3 Processing Software Breakpoints

Software breakpoints are of Break Before Make type, therefore the instruction at the breakpoint is executed only after the proper debug action is taken.

Upon fetching a TRAP instruction, Software Breakpoint is effected. Due to this, it goes to Debug Mode.

Attention: The user software must not clear the TRAP_EN bit in EO register.

The OCDS reacts in two ways:

- a Break event is recognized for Entering Monitor Mode
This happens when:
(MMCR2.MMODE=0) and (MMBPCR.SWBC=1 or MMCR.MSTEP=1)
i.e. when Software Breakpoints are enabled or a Single Step has been executed.
The further processing is as described in [Section 13.8.6.1](#).
- the Software Break is ignored
This happens when the above condition is not satisfied, i.e. basically when the Software Breakpoints are disabled with MMBPCR.SWBC=0.

Upon matching a not enabled Software Breakpoint, the user software execution is suppressed for 4 clock cycles after the TRAP instruction.

Note: In order to continue user program execution after the debug event, an external Debugger must restore the original PC value at the address of the current software breakpoint.

13.8.8 Reactions on Breakpoints without Monitor Entry

In case a Break-event happens while Monitor Mode is disabled:

(MMCR2.MBCON=1) AND (MMCR2.MMODE=0)

, the Monitor program is not started (MM_no_entry) but another action is triggered by OCDS system.

13.8.8.1 Triggering a NMI request

Instead of starting the Monitor program, OCDS NMI is requested, refer [Section 13.8.10.1](#).

The flags FNMIRR/FNMIRW here set can be cleared only per software - this must be done by the user NMI-responding routine.

13.8.9 Important Notes about Break Event Processing

A few specifics must be taken into account - especially by tool vendors - when using XC800 OCDS.

13.8.9.1 Calculating the Return Address

Here two values are considered:

- the **Break Address** - the address saved while **Entering Monitor Mode**, refer to [Section 13.8.6.1](#);
- the **Return Address** - the address where from the user program will continue upon Monitor Exit, i.e. the address of the first instruction executed when back in User Mode.

The relationship between the both addresses depends on the type of event which has caused the Break. This type is to be determined by:

- Breakpoint Configuration - as programmed within MMBPCR register,
- Breakpoint Flags - as set by OCDS-hardware within the MMSR register.

The behavior required in different scenarios is described below:

1. in case of **Hardware Breakpoints** (refer to [Section 13.8.7.1](#)):
Return Address = Break Address - 2
2. in case of **External Breaks** and **Single Step Execution** (refer to [Section 13.8.7.2](#) and [Section 13.8.6.3](#)):
Return Address = Break Address - 1

3. in case of **Software Breakpoints** (refer to [Section 13.8.7.3](#))
both the above equations are applicable in the following sense:

- a) if take Return Address = Break Address - 2
this will cause upon Monitor exit an instruction to be fetched again from the same address where the Software BP has been raised - i.e. where the TRAP has been fetched from.

Therefore, if the TRAP has not been replaced by another opcode during the Monitor Mode, this will lead immediately to a new Software Break and again to Monitor entry, with no user instruction executed in between.

- b) if take Return Address = Break Address - 1
this will cause upon Monitor exit the next instruction to be fetched from the location immediately following the address where the Software BP has been matched - i.e. the next instruction after the TRAP.

Therefore, such an address-calculation is applicable when the Software BPs - i.e. the TRAPs - stay continuously within the code debugged but not dynamically swapped with user instructions.

The Return Address is also related in general to the point where a Debug Event has happened:

- upon **Breakpoints on Instruction Address** and Software Breakpoints, the events are of type "Break Before Make".

In this case the Return Address targets just the instruction which has caused the Break, and this instruction will be only executed after returning to User Mode.

- upon **Breakpoints on Internal RAM Address** the events are of type "Break After Make", i.e. the user program execution is broken only after the IRAM access configured as a breakpoint has been performed.
In this case the Return Address again targets the next instruction to be executed after returning to user mode, but the instruction which has caused the Break can be only determined, if MMICR.DVECT=0 upon monitor entry (refer to **Section**).
- the External Breaks are anyway asynchronous events - not caused by execution of an instruction.
Therefore, the only needed in this case is to have a correct continuity of the program-flow - this is assured by the Return Address properly calculated as shown above.

Feature Discontinued

13.8.9.2 Specific Behavior while XC800 is in Debug Mode

Upon a recognized and enabled Break-event, the core is immediately put into Debug Mode. There are two possibilities about the further processing:

- Monitor is to be entered
- No Monitor-Mode

The scenarios considered:

- If available, MBC pin only must be activated (MMCP2.MBCON=1),
- a TRAP-instruction is fetched from the user-code but Software Breakpoints are disabled (MMBPCR.SWBC=0).

In both these cases, the core is put into Debug Mode for 4 SCLK cycles, then released to continue user-code execution.

During these 4 SCLK cycles, **any other Break-request will be ignored** by OCDS.

In fact, only an **External Break** is possible meanwhile, as far as XC800 is in Debug Mode and all its activity is suppressed now.

Feature Discontinued

13.8.10 NMI Request and Control by OCDS

The OCDS module provides hardware allowing to generate a NMI, request itself as well as to control the general NMI-processing within the XC800 system.

13.8.10.1 NMI request from OCDS

The OCDS generates one interrupt request at the output `ocds_nmi_o`. This signal goes to the Interrupt-Management Module into XC800 SCU (System Control Unit), where it sets a flag within NMI Status Register NMISR. In case OCDS is an enabled source in NMI Control Register NMICON, a non-maskable interrupt request will be activated to the core.

The NMI-request can be activated by OCDS upon memory access performed to defined address areas (separately configurable for read/write operations) in XC800 Internal RAM. This is an advanced and fully controllable by application code feature, allowing to utilize parts of OCDS not for debugging but to support a general, user-accessible functionality.

The basics for implementation of this feature are:

- the Breakpoint Generation module is able to recognize read/write accesses to IRAM address-ranges (refer to [Section](#))
- upon a breakpoint match but while MMCR2.MBCON=1, the Monitor firmware is not started and therefore in fact no debug-session takes place.

Note: It is required to set the EO.TRAP_EN bit to enable the breakpoint generation.

To configure, trigger and handle an OCDS NMI request upon IRAM access (refer also to [Section 13.8.8.1](#)):

1. at all is possible only as long as MMCR2.MBCON=1
2. user software must execute the following sequence:
 - a) set the NMI-enable bit(s) NMIRRE/NMIRWE in MMICR, respectively for request upon IRAM read/write
 - b) write the low address(es) of the observed area into HWBP2L/HWBWP3L, respectively for NMI-request upon IRAM read/write
 - c) write the high address(es) of the observed area into HWBP2H/HWBWP3H, respectively for NMI-request upon IRAM read/write
3. after the settings according to point 2) are done and as long as MBCON=1, a NMI-request will be triggered by the OCDS upon an access to the respectively configured (for read/write) IRAM area(s).
4. in case a NMI-request has been raised by the OCDS, respective flag FNMIRR/FNMIRW is set in MMICR - refer to the register-description in [Section 13.8.8.1](#) and functional definition in [Section 13.8.8.1](#).

If the request is sent to the core - in case enabled by NMICON.NMIOCD=1 (in SCU) - then a NMI-servicing routine will be invoked by a jump to its standard location in XC800 Program Memory, and this routine:

- a) can identify OCDS as NMI-requesting source and the type of IRAM-access causing the request by evaluating NMISSR.FNMIOOCS (in SCU) and MMICR.FNMIRR/FNMIRW (in OCDS)
 - b) must clear FNMIRR/FNMIRW flags before its end, to allow a proper reaction on next NMIs.
5. any attempt to configure/activate a Breakpoint for debugging by writing into MMBPCR will reset the internal nmi_*e/nmi_*e_l flags and therefore - disable the generation of NMI-request upon IRAM access.

Attention: all the software-handling of OCDS NMI (servicing routine) is to be supported by user code!

13.8.10.2 OCDS NMI Service Routine Handling

For the correct HW behavior when the OCDS NMI is triggered due to IRAM read/write access, special handling is necessary in OCDS nmi service routine to set up a correct RETI address.

The recommended code handling is:

```

NMI_OCDS_MAIN: ;OCDS NMI service routine
    POP RETADD_H ;get the return address stored in SP
    POP RETADD_L
    ;/--user code to handle the OCDS NMI---//
    ;// USER CODE BEGIN
    ...
    ;// USER CODE END
    MOV A,RETADD_L
    CJNE A,#00H,DEC_LOW_BYTE ;low address byte under-flow?
    DEC RETADD_H
DEC_LOW_BYTE:
    DEC RETADD_L ;decrement return address by one location
    PUSH RETADD_L ;store the correct return address back to SP
    PUSH RETADD_H
    RETI
    
```

13.8.10.3 General NMI Control by OCDS

The controls are aimed to support a correct NMI-related behavior:

- all the pending NMI requests raised before Monitor Mode entry and as long as the system is in Monitor Mode will be not affected, i.e. these requests are safely kept and sent to the core when the system returns to User Mode

13.8.11 NMI-mode priority over Debug-mode

While the core is in NMI-mode (after an NMI-request has been accepted and before the RETI instruction is executed, i.e. the time during a NMI-servicing routine), certain debug functions are blocked/restricted:

1. No external break (refer to [Section 13.8.7.2](#)) is possible while the core is servicing a NMI.
External break requested inside a NMI-servicing routine will be taken only after RETI is executed.
2. A breakpoint into NMI-servicing routine is taken, but single stepping (refer to [Section 13.8.6.3](#)) is not possible afterwards.
If a step is requested, the servicing routine will run continuously and monitor mode will be invoked again only after a RETI is executed.

Hardware breakpoints and software breakpoints proceed as normal while CPU is in NMI-mode.

Feature Discontinued

13.8.12 Registers Description

The OCDS Special Function Registers are accessed from the mapped SFR area. [Table 13-31](#) lists the (direct addressable) SFRs and corresponding address.

Table 13-31 Register Map (Direct Addressable)

Address	Register
F1 _H	MMCR
F2 _H	MMSR
F3 _H	MMBPCR ⁽¹⁾
F4 _H	MMICR
E9 _H	MMCR ⁽¹⁾
F5 _H	MMDR
F6 _H	HWBPSR
F7 _H	HWBPDR ⁽¹⁾
EB _H	MMWR ⁽¹⁾
EC _H	MMVR2

1) The whole register or some of the bits are writable by software only in Monitor and Startup Mode, i.e. upon MMCR2.MMODE=1 OR MMCR2.STMODE=1 (marked also within the register description).

Additionally, there are Hardware Breakpoint Registers, which are indirect accessible via HWBPSR (register select) and HWBPDR (data) - refer to [Table 13-32](#) and [Table 13-33](#) (for more information look at [Section 13.8.12](#)).

Table 13-32 Hardware Breakpoint Registers (Indirect Accessed)

Register	Description
HWBP0L	Hardware Breakpoint 0 Low Register
HWBP0H	Hardware Breakpoint 0 High Register
HWBP1L	Hardware Breakpoint 1 Low Register
HWBP1H	Hardware Breakpoint 1 High Register
HWBP2L	Hardware Breakpoint 2 Low Register
HWBP2H	Hardware Breakpoint 2 High Register
HWBP3L	Hardware Breakpoint 3 Low Register
HWBP3H	Hardware Breakpoint 3 High Register

Table 13-33 HWBPSR [3:0]: Selecting Hardware Breakpoint Registers

BPSEL	Register Selected	BPSEL	Register Selected
0xxx	reserved - no selection		
1000	HWBP0L	1001	HWBP0H
1010	HWBP1L	1011	HWBP1H
1100	HWBP2L	1101	HWBP2H
1110	HWBP3L	1111	HWBP3H

The OCDS registers are exclusively dedicated to the on-chip Monitor program and the user should not write into them. Most of the OCDS registers or separate bits/fields are protected and can not be written by user software but only by the firmware in two modes of XC800:

- Startup mode (STMODE) - while the Bootcode is executed after reset, the user code is still not started
- Monitor mode (MMODE) - while the Monitor program is running, the user code is in break.

Therefore an unintentional access to OCDS registers by the user software cannot disturb the normal debug functionality.

Note about the bits modified both by HW and SW

- Here considered are the bits marked as (r)wh within the register descriptions below:
 - **the rule** is: if both the OCDS hardware and the software are accessing to set the same bit, the hardware has priority
 - **one exception** is the MMCF2MICON bit, where the software-access has priority

Note: Apart from the above, the implementation takes care: if a bit is modified by hardware during the execution of a read-modify-write instruction, the new value set by HW is not affected by the SW write-back operation.

Feature Discontinued

13.8.12.1 Control and Status Registers

MMCR

Monitor Mode Control Register (F1_H)

Reset value: 00_H

RMAP: 1, PAGE: X

7	6	5	4	3	2	1	0
MEXIT_P	MEXIT	MMNMI	MSTEP	MRAMS_P	MRAMS	TRF	RRF
w	rwh	rw	rw	w	rwh	rh	rh

Field	Bits	Type	Description
RRF	0	rh	Receive Register Full Flag 1 _B A New Byte in MMRR received from debug interface
TRF	1	rh	Transmit Register Ready Flag 1 _B A New Byte written by XC800 into MMTR
MRAMS	2	rwh	Internal RAM control 0 _B User Internal RAM selected 1 _B monitor RAM selected
MRAMS_P	3	w	Bit protection: 0 _B MRAMS unchangeable 1 _B MRAMS can be changed
MSTEP¹⁾	4	rw	Execute a Single Step from User Code 0 _B Normal Monitor-exit to follow 1 _B Single Step requested by the Monitor
MMNMI¹⁾	5	rw	NMI during Monitor Mode 0 _B disabled 1 _B enabled refer to Section 13.8.10.3
MEXIT	6	rw	Return to User Code 0 _B No effect 1 _B Monitor-Exit requested
MEXIT_P	7	w	Bit protection: 0 _B MEXIT unchangeable 1 _B MEXIT can be changed

1) Writable by software only in Monitor and Startup Mode, i.e. upon MMCR2.MMODE=1 OR MMCR2.STMODE=1.

RRF bit:

-set via the debug interface (refer to [Section 13.8.5](#))

-reset upon reading MMRR by XC800

TRF bit:

-set upon writing MMTR by XC800

-reset upon writing MMTR to MMSHR (refer to [Section 13.8.6.1](#))

MRAMS bit:

-set upon Monitor-entry (refer to [Section 13.8.6.1](#))

-changeable by software (Monitor, Boot code)

MEXIT bit:

-set by software (the Monitor) only

-reset by OCDS-hardware upon returning to the user-code (refer to [Section 13.8.6.2](#))

MMCR2

Monitor Mode Control Register 2

(E9_H)

Reset value: A0_H or A1_H

RMAP: 1, PAGE: X

7	6	5	4	3	2	1	0
STMODE		DSUSP	MBCON	ALTDI	MMEP	MMODE	JENA
r	rw	rw	rwh	rw	rwh	rh	rh

Field	Bits	Type	Description
JENA	0	rh	Debug Interface status 0 _B Debug interface is not available or not selected 1 _B Debug interface is available and selected from the Host
MMODE	1	rh	Current OCDS- and system- status 0 _B User Mode 1 _B Monitor running
MMEP¹⁾	2	rwh	Monitor Mode Entry Point 0 _B Safe Entry 1 _B Normal Entry
ALTDI	3	rw	Set as 0.
MBCON¹⁾	4	rwh	Do not enter Monitor Mode on Break 0 _B Disabled (Monitor Mode entered) 1 _B Enabled (Monitor Mode ignored)

Field	Bits	Type	Description
DSUSP¹⁾	5	rw	Debug-Suspend signal activation in Monitor Mode 0 _B Disabled 1 _B Enabled
STMODE	7	r	Startup Mode 0 _B Disabled 1 _B Enabled

1) Writable by software only in Monitor and Startup Mode, i.e. upon MMCR2.MMODE=1 OR MMCR2.STMODE=1.

JENA bit:

- reset by OCDS-reset
- can be set via the JTAG/DAP/SPD interface (refer to [Section 13.8.5](#))

MMODE bit:

- changeable by OCDS-hardware only
- main control bit reflecting the overall debug functionality
- controls directly the wdt_susp_o signal to the System Control Unit, which signal disables the Watchdog Timer while in Monitor Mode (refer to [Section 13.8.11](#))

MMEP bit:

- reset by CPU-reset
- set upon Monitor-entry (refer to [Section 13.8.6.1](#))
- software can only reset it

MBCON bit:

- changeable by software
- can be set/reset via the JTAG/DAP/SPD interface (refer to [Section 13.8.5](#))

DSUSP bit:

- set to 1 after device reset - supported modules are suspended by default during a debug-break.

For additional information on this functionality, refer to the System Control Unit Chapter of product documentation.

STMODE bit:

- set by any system Reset
- software can only reset it

This bit is reset by the startup software of the device.

MMSR
Monitor Mode Status Register
RMAP: 1, PAGE: X
(F2_H)
Reset value: 00_H

7	6	5	4	3	2	1	0
		EXBF	SWBF	HWB3F	HWB2F	HWB1F	HWB0F
rw	rw	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
HWB0F¹⁾	0	rwh	Hardware Breakpoint 0 Flag
HWB1F¹⁾	1	rwh	Hardware Breakpoint 1 Flag
HWB2F¹⁾	2	rwh	Hardware Breakpoint 2 Flag
HWB3F¹⁾	3	rwh	Hardware Breakpoint 3 Flag
SWBF¹⁾	4	rwh	Software Break Flag
EXBF¹⁾	5	rwh	External Break Flag

1) Anyone of these flags is set upon the occurrence of a respective event.

They can be cleared by software only, as well the software can only reset but not set them.

MMBPCR
BreakPoints Control Register
RMAP: 1, PAGE: X
(F3_H)
Reset value: 00_H

7	6	5	4	3	2	1	0
SWBC	HWB3C		HWB2C		HWB1C	HWB0C	
rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
HWB0C	[1:0]	rw	Hardware Breakpoint 0 Configuration see Table 13-30
HWB1C	2	rw	Hardware Breakpoint 1 Configuration see Table 13-30

Field	Bits	Type	Description
HWB2C	[4:3]	rw	Hardware Breakpoint 2 Configuration: 00 _B disabled 01 _B HWBP2L=< IRAM Read Address=< HWBP2H 10 _B Program Memory Address = HWBP2
HWB3C	[6:5]	rw	Hardware Breakpoint 3 Configuration: 00 _B disabled 01 _B HWBP3L =< IRAM Write Address =< HWBP3H 10 _B Program Memory Address = HWBP3
SWBC	7	rw	Software Breakpoints Control 0 _B disabled 1 _B enabled

Note: The whole MMBPCR register is writable by software only in Monitor and Startup Mode, i.e. upon MMCR2.MMODE=1 OR MMCR2.STMODE=1.

MMICR

Monitor Mode Interrupt Control Register (F1)

Reset value: 00_H

RMAP: 1, PAGE: X

7	6	5	4	3	2	1	0
DVECT		COMRST	MSTSEL	FNMI _R W	FNMI _R R	NMIRWE	NMIRRE
rwh		rwh	rh	rwh	rwh	rw	rw

Field	Bits	Type	Description
NMIRRE	0	rw	NMI upon IRAM read 0 _B Disabled 1 _B Enabled Refer to Section 13.8.10.1
NMIRWE	1	rw	NMI upon IRAM write 0 _B Disabled 1 _B Enabled Refer to Section 13.8.10.1
FNMI_RR	2	rwh	IRAM Read NMI Flag
FNMI_RW	3	rwh	IRAM Write NMI Flag

Field	Bits	Type	Description
MSTSEL	4	rh	Monitor Start Address Selection: 0 _B 0'D000 _H - Monitor in Bank 0 1 _B
COMRST	5	rwh	Communication-protocol Restart requested
DVECT	7	rwh	DebugVector signal latched

FNMIRR / FNMIRW bits:

-set by hardware in case of NMI request from OCDS activated upon read/write access to the Internal RAM (refer to [Section 13.8.8.1](#) and [Section 13.8.10.1](#))

-the software can only reset these bits

MSTSEL bit:

-set and reset by hardware only, directly following the current value of `mm_start_sel_i` input

-the software can only read it

This bit is used to determine the running Memory Scheme, in particular - by the Monitor program upon [Entering Monitor Mode](#) (refer to instruction sequences in [Section 13.8.6.1](#)).

COMRST bit:

-set by hardware

-reset by software only; the software can only reset it but not set

The bit is used to synchronize the communication protocol between the on-chip Monitor and the host Debugger. This will be needed, if some error happens and/or the data-flow can not be correctly recognized. In such a case:

-the Debugger sends a dedicated command to set COMRST=1

-the Monitor polls COMRST before starting any JTAG/DAP/SPD-transaction, i.e. together with checking RRF. Upon COMRST=1 the Monitor resets it and "resets itself" - i.e. jumps to the beginning of JTAG/DAP/SPD communication protocol (waiting for the first transaction).

DVECT bit:

-can be set by hardware only (refer to [Section](#))

-reset by software only; the software can only reset it but not set

Feature Discontinued

13.8.12.2 Data Transfer Registers

What considers the addressing, only one data-register (MMDR) is to be accessed. The real operations are performed as follows:

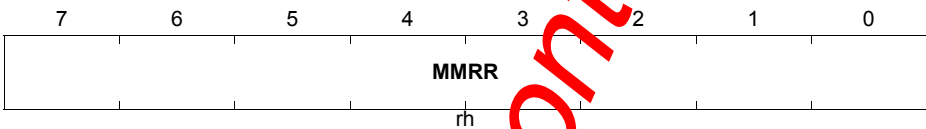
- Read: from the JTAG/DAP/SPD Receive Data register MMRR,
- Write: to the JTAG/DAP/SPD Transmit Data register MMTR.

MMDR

Monitor Mode Data Register - Receive (F5_H)

Reset value: 00_H

RMAP: 1, PAGE: X



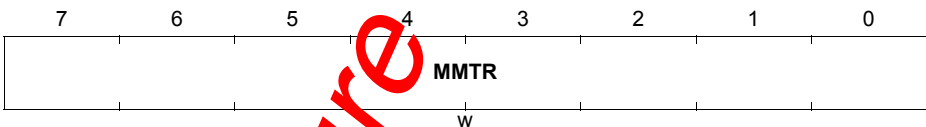
Field	Bits	Type	Description
MMRR	[7:0]	rh	Monitor Mode JTAG/DAP/SPD Receive Register

MMDR

Monitor Mode Data Register - Transmit (F5_H)

Reset value: 00_H

RMAP: 1, PAGE: X



Field	Bits	Type	Description
MMTR	[7:0]	w	Monitor Mode JTAG/DAP/SPD Transmit Register

Feature Discontinued

13.8.12.3 Hardware Breakpoint Registers

All of the Hardware Breakpoint Registers HWBP0L..HWBP3H within OCDS can be set to the desired compare-values indirectly - by writing only to one software-accessible register HWBPDR, after BPSEL bitfield within the select-register HWBPSR has been properly programmed (refer to [Table 13-33](#)).

HWBPSR

Hardware Breakpoints Select Register (F6_H)

Reset value: 00_H

RMAP: 1, PAGE: X

7	6	5	4	3	2	1	0
0	0	0	BPSEL_P			BPSEL	
r	r	r	w			rw	

Field	Bits	Type	Description
BPSEL	[3:0]	rw	BreakPoint register Select refer to Table 13-33
BPSEL_P	4	w	Bit protection 0 _B BPSEL unchangeable 1 _B BPSEL can be changed
0	[7:5]	r	Reserved

HWBPDR

Hardware Breakpoints Data Register (F7_H)

Reset value: 00_H

RMAP: 1, PAGE: X

7	6	5	4	3	2	1	0
HWBPxx							
rw							

Field	Bits	Type	Description
HWBPxx	[7:0]	rw	Data to be written into/read from a HWBPxx register, as currently selected by HWBPSR

Note: The whole HWBPDR register is generally writable by software only in Monitor and Startup Mode, i.e. upon MMCR2.MMODE=1 OR MMCR2.STMODE=1. Only in

case of MMCR2.MBCON=1, it is not required per above and HWBPDR is always writable.

HWBP0L

Hardware Breakpoint 0 Low Register written via HWBPDR Reset value: 00_H

7	6	5	4	3	2	1	0
HWBP0L							

Field	Bits	Type	Description
HWBP0L	[7:0]		The Low Byte from Compare Address HWBP0

HWBP0H

Hardware Breakpoint 0 High Register written via HWBPDR Reset value: 00_H

7	6	5	4	3	2	1	0
HWBP0H							

Field	Bits	Type	Description
HWBP0H	[7:0]		The High Byte from Compare Address HWBP0

HWBP1L

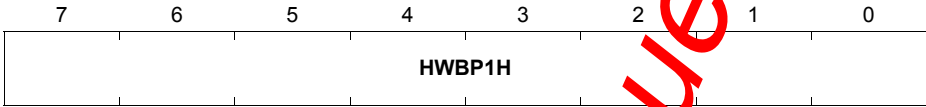
Hardware Breakpoint 1 Low Register written via HWBPDR Reset value: 00_H

7	6	5	4	3	2	1	0
HWBP1L							

Field	Bits	Type	Description
HWBP1L	[7:0]		The Low Byte from Compare Address HWBP1

HWBP1H

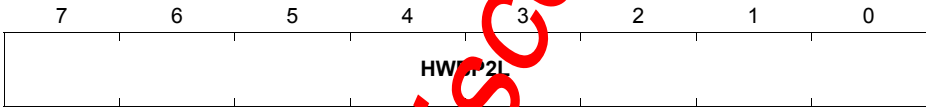
Hardware Breakpoint 1 High Register written via HWBPDR Reset value: 00_H



Field	Bits	Type	Description
HWBP1H	[7:0]		The High Byte from Compare Address HWBP1

HWBP2L

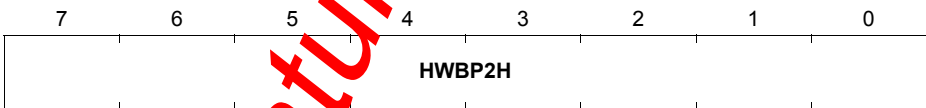
Hardware Breakpoint 2 Low Register written via HWBPDR Reset value: 00_H



Field	Bits	Type	Description
HWBP2L	[7:0]		The Low byte from Compare Address HWBP2

HWBP2H

Hardware Breakpoint 2 High Register written via HWBPDR Reset value: 00_H



Field	Bits	Type	Description
HWBP2H	[7:0]		The High Byte from Compare Address HWBP2

Feature Discontinued

HWBP3L
Hardware Breakpoint 3 Low Register written via HWBPDR Reset value: 00_H

7	6	5	4	3	2	1	0
HWBP3L							

Field	Bits	Type	Description
HWBP3L	[7:0]		The Low Byte from Compare Address HWBP3

HWBP3H
Hardware Breakpoint 3 High Register written via HWBPDR Reset value: 00_H

7	6	5	4	3	2	1	0
HWBP3H							

Field	Bits	Type	Description
HWBP3H	[7:0]		The High Byte from Compare Address HWBP3

13.8.12.4 Monitor Work Registers

These two registers are dedicated to be used as work locations by the Monitor program.

MMWR1
Monitor Work Register 1 (EB_H) Reset value: 00_H
 RMAP: 1, PAGE: X

7	6	5	4	3	2	1	0
MMWR1							
rw							

Field	Bits	Type	Description
MMWR1	[7:0]	rw	Work location 1 for the Monitor Program

Note: The whole MMWR1 register is writable by software only in Monitor and Startup Mode, i.e. upon MMCR2.MMODE=1 OR MMCR2.STMODL=1.

MMWR2 can be used for general purposes when no debug-session is possible: if the device is not started in OCDS mode and no external device is connected.

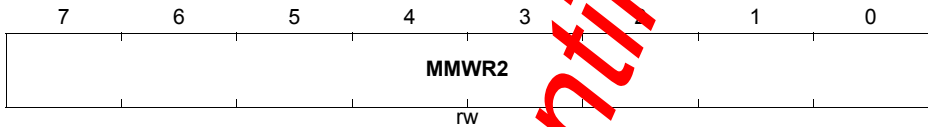
MMWR2

Monitor Work Register 1

(EC_H)

Reset value: 00_H

RMAP: 1, PAGE: X



Field	Bits	Type	Description
MMWR2	[7:0]	rw	Work location 2 for the Monitor Program

Feature Discontinued

Feature Discontinued

13.9 General Purpose I/O Ports and Peripheral I/O Lines (Ports)

The TC2x_SCR has a maximum of 16 general purpose I/O ports with 8-bit peripheral I/O lines that are shared with the 32-bit peripherals I/O lines. It is organized into two parallel ports, Port 00 (SCR_P00) and Port 02 (SCR_P02). The pin assignment of these functions are shown in [Table 13-34](#).

Upon reset, the 32-bit Ports module has the full control of all the pads. During run mode, the control of these 16 pins can be switched to TC2x_SCR if necessary via the 32-bit Ports register Pn_PCSR. For P00.2 and P00.3, SPARE[NOB]0.PULL_CTRL needs to be configured in addition to Pn_PCSR, to enable pull-up and pull-down control by the 8-bit controller. During standby mode, the control of these ports will be based on the setting of register Pn_PCSR.SELx before the device enter standby mode. If the control is from the 32-bit controller during standby mode, the pad will stay in its default reset state based on the setting of PMSWCR0.TRISTREQ.

A separate set of pad control registers as shown in [Table 13-37](#) are available in TC2x_SCR to control these ports.

Table 13-34 Pin assignment for Ports in TC2x_SCR

TC2x_SCR Ports	TC2x Ports
SCR_P00.0 - SCR_P00.7	P00.0 - P00.7
SCR_P02.0 - SCR_P02.7	P02.0 - P02.7

13.9.1 Basic Port Operation

[Figure 13-39](#) is a general block diagram of a TC2x_SCR GPIO port slice.

Feature Discontinued

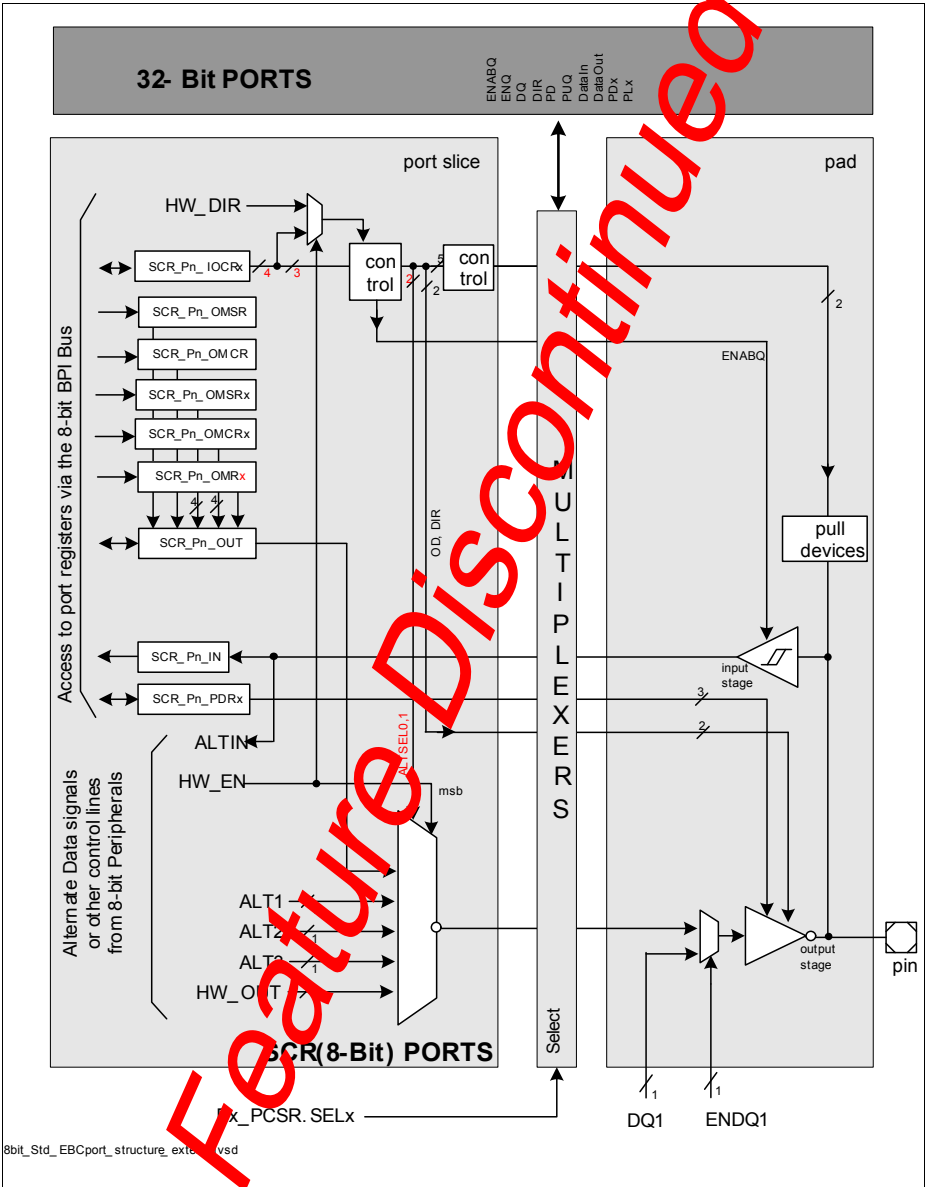


Figure 13-39 General Structure of a Port Pin

Each port line has a number of control and data bits, enabling very flexible usage of the line. Each port pin can be configured for input or output operation. In input mode (default after reset), the output driver is switched off (high-impedance). The actual voltage level present at the port pin is translated into a logical 0 or 1 via a Schmitt-Trigger device and can be read via the read-only register SCR_Pn_IN (n=00, 02). Input signals are connected directly to the various inputs of the peripheral units (AltDataIn). The function of the input line from the pin to the input register SCR_Pn_IN and to AltDataIn is independent of whether the port pin operates as input or output.

This means that when the port is in output mode, the level of the pin can be read by software via SCR_Pn_IN or a peripheral can use the pin level as an input.

In output mode, the output driver is activated and drives the value supplied through the multiplexer to the port pin. Switching between input and output mode is accomplished through the SCR_Pn_IOCR register, which enables or disables the output driver. If a peripheral unit uses a GPIO port line as a bi-directional I/O line, register SCR_Pn_IOCR has to be written for input or output selection. The SCR_Pn_IOCR register further controls the driver type of the output driver, and determines whether an internal weak pull-up, pull-down, or without input pull device is alternatively connected to the pin when used as an input. This offers additional advantages in an application.

The output multiplexer in front of the output driver selects the signal source for the GPIO line when used as output. If the pin is used as a general-purpose output, the multiplexer is switched by software (SCR_Pn_IOCR register) to the Output Data Register SCR_Pn_OUT. Software can set or clear the bit in SCR_Pn_OUT through separate SCR_Pn_OMSR or SCR_Pn_OMCR registers. The set or clear operations for the bits in SCR_Pn_OUT can also be done for up to four bits per register in SCR_Pn_OMSRx and SCR_Pn_OMCRx (x=0,4). Alternatively, the set, clear or toggle function can be achieved through SCR_Pn_OMR0 and SCR_Pn_OMR1, where adjacent pins within the same port can be set, cleared or toggled within one write operation. The manipulation of the control bits in these registers can directly influence the state of the port pin. If the on-chip peripheral units use the pin for output signals, the alternate output lines ALT1 to ALT7 can be switched via the multiplexer to the output driver. The data written into the output register SCR_Pn_OUT by software can be used as input data to an on-chip peripheral. This enables, for example, peripheral tests via software without external circuitry.

When selected as general-purpose output line, the logic state of each port pin can be changed individually by programming the pin-related bits in the Output Modification Set Register SCR_Pn_OMSR, Output Modification Set Register x SCR_Pn_OMSRx (x=0,4), Output Modification Clear Register SCR_Pn_OMCR, Output Modification Clear Register x SCR_Pn_OMCRx (x=0,4) or Output Modification Register, OMR0/OMR_1. The bits in SCR_Pn_OMSR/SCR_Pn_OMSRx and SCR_Pn_OMCR/SCR_Pn_OMCRx make it possible to set and clear the bits in the SCR_Pn_OUT register. While the bits in SCR_Pn_OMR0/SCR_Pn_OMR1 allows the bits in SCR_Pn_OUT to be set, cleared, toggled or remain unchanged.

When selected as general-purpose output line, the actual logic level at the pin can be examined through reading Pn_IN and compared against the applied output level (either applied through software via the output register SCR_Pn_OUT, or via an alternate output function of a peripheral unit). This can be used to detect some electrical failures at the pin caused through external circuitry. In addition, software-supported arbitration schemes can be implemented in this way using the open-drain configuration and an external wired-And circuitry. Collisions on the external communication lines can be detected when a high level (1) is output, but a low level (0) is seen when reading the pin value via the input register SCR_Pn_IN.

Feature Discontinued

13.9.2 Description Scheme for the Port IO Functions

The following two general building blocks can be used to describe each GPIO pin:

Table 13-35 Port x Input/Output Functions

Port Pin	I/O	Select	Connected Signal(s)	From / to Module
Px.y	Input		Signal(s)	module(s)
	Output	GPIO	Signal	module
		ALT1	Signal	module
		ALT2	Signal	module
		ALT3	Signal	module
		ALT4	Signal	module
		ALT5	Signal	module
		ALT6	Signal	module
	ALT7	Signal	module	
HW_DIR	HW_Out	Signal	module; group En	

or:

Feature Discontinued

Table 13-36 Port x Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port Pin Control Select.		
				Reg./Bit Field	Value	
Px.y	I	GPIO	Px_IN.Py	Px_IOCRz. I0	0XXX0 _B	
	O	GPIO		Px_OUT.Py		1X000 _B
						1X001 _B
						1X010 _B
						1X011 _B
						1X100 _B
					1X101 _B	
					1X110 _B	
					1X111 _B	
HW_DIR		module; group En	Signal		HW_DIR	

- **HW_DIR:**
The type Alternate Direction signal which is needed if HW_En is active:
 - Out -always output
DIRx - the pins in one port having the same DIRx (x=0, 1, 2, ...), are controlled as a group by a dedicated HW_DIR signal.
SDIR- Single DIR- the pin is controlled by its own, dedicated, single HW_DIR signal.
- grouping indicates if the respective pin is controlled by hardware:
 - ENx - the pins in one port having the same ENx (x=0, 1, 2, ...), are controlled as a group by a dedicated HW_EN signal.
 - SEN - Single EN - the pin is controlled by its own, dedicated, single HW_EN signal
- Digital port slices with HW_DIR defined are the ports described in [Figure 13-39](#).

HW_DIR signal, output case, switches the pad to push-pull output state.

HW_DIR signal, input case, switches the pad to the input state with pull-up/down setting as defined by the PCR register.

13.9.3 Port Register Overview

The individual control and data bits of each parallel port are implemented in a number of 8-bit registers. Bits with the same meaning and function are assembled together in the same register. The registers configure and use the port as general purpose I/O or alternate function input/output. For some ports, not all registers are implemented. The availability of the registers in the specific ports is described separately. The definition of these registers are similar to the 32-bit format.

Table 13-37 Port Registers

Register Short Name	Register Long Name	Description
SCR_Pn_IOCR0	Port n Input/Output Control Register 0	Page 13-196
SCR_Pn_IOCR1	Port n Input/Output Control Register 1	Page 13-198
SCR_Pn_IOCR2	Port n Input/Output Control Register 2	Page 13-199
SCR_Pn_IOCR3	Port n Input/Output Control Register 3	Page 13-200
SCR_Pn_IOCR4	Port n Input/Output Control Register 4	Page 13-201
SCR_Pn_IOCR5	Port n Input/Output Control Register 5	Page 13-202
SCR_Pn_IOCR6	Port n Input/Output Control Register 6	Page 13-203
SCR_Pn_IOCR7	Port n Input/Output Control Register 7	Page 13-204
SCR_Pn_PDR0	Port n Pad Driver Mode 0 Register	Page 13-208
SCR_Pn_PDR2	Port n Pad Driver Mode 2 Register	Page 13-208
SCR_Pn_PDR4	Port n Pad Driver Mode 4 Register	Page 13-209
SCR_Pn_PDR6	Port n Pad Driver Mode 6 Register	Page 13-209
SCR_Pn_OUT	Port n Output Register	Page 13-210

Table 13-37 Port Registers (cont'd)

Register Short Name	Register Long Name	Description
SCR_Pn_OMR0	Port n Output Modification Register 0	Page 13-21 1
SCR_Pn_OMR1	Port n Output Modification Register 1	Page 13-21 2
SCR_Pn_OMSR	Port n Output Modification Set Register	Page 13-21 3
SCR_Pn_OMSR0	Port n Output Modification Set Register 0	Page 13-21 4
SCR_Pn_OMSR4	Port n Output Modification Set Register 4	Page 13-21 4
SCR_Pn_OMCR	Port n Output Modification Clear Register	Page 13-21 6
SCR_Pn_OMCR0	Port n Output Modification Clear Register 0	Page 13-21 7
SCR_Pn_OMCR4	Port n Output Modification Clear Register 4	Page 13-21 8
SCR_Pn_IN	Port n Input Register	Page 13-21 9

The Port SFRs are located in the standard memory area (RMAP = 0) and the addresses are listed in [Table 13-38](#).

Table 13-38 SFR Address List for P0 and P2

Address	Register	Address	Register
90 _H	SCR_P00_OUT	C8 _H	SCR_P02_OUT
A0 _H	SCR_P00_IN	D8 _H	SCR_P02_IN
91 _H	SCR_P00_OMR0	CA _H	SCR_P02_OMR0
92 _H	SCR_P00_OMR1	CB _H	SCR_P02_OMR1
93 _H	SCR_P00_IOCR0	CC _H	SCR_P02_IOCR0
94 _H	SCR_P00_IOCR1	CD _H	SCR_P02_IOCR1
9A _H	SCR_P00_IOCR2	CE _H	SCR_P02_IOCR2
9B _H	SCR_P00_IOCR3	CF _H	SCR_P02_IOCR3
9C _H	SCR_P00_IOCR4	EC _H	SCR_P02_IOCR4
9D _H	SCR_P00_IOCR5	ED _H	SCR_P02_IOCR5

Table 13-38 SFR Address List for P0 and P2 (cont'd)

9E _H	SCR_P00_IOCRR6	EE _H	SCR_P02_IOCRR6
9F _H	SCR_P00_IOCRR7	EF _H	SCR_P02_IOCRR7
A1 _H	SCR_P00_PDR0	D1 _H	SCR_P02_PDR0
A3 _H	SCR_P00_PDR2	D3 _H	SCR_P02_PDR2
A4 _H	SCR_P00_PDR4	D4 _H	SCR_P02_PDR4
A5 _H	SCR_P00_PDR6	D5 _H	SCR_P02_PDR6
A9 _H	SCR_P00_OMSR0	D9 _H	SCR_P02_OMSR0
BA _H	SCR_P00_OMSR4	DA _H	SCR_P02_OMSR4
BB _H	SCR_P00_OMCR0	DB _H	SCR_P02_OMCR0
BC _H	SCR_P00_OMCR4	DC _H	SCR_P02_OMCR4
BD _H	SCR_P00_OMSR	DD _H	SCR_P02_OMSR
BE _H	SCR_P00_OMCR	DE _H	SCR_P02_OMCR

Feature Discontinued

13.9.3.1 Port Input/Output Control Registers

The port input/output control registers select the digital output and input driver functionality and characteristics of a GPIO port pin. Port direction (input or output), pull-up, pull-down, or no pull devices for inputs, and push-pull or open-drain functionality for outputs can be selected by the corresponding bit fields PCx (x = 0-7).

For P00.2 and P00.3, pull-up and pull-down can be controlled by the 8-bit controller by setting SPAREINOUT0.PULL_CTRL (see [SPAREINOUT0](#) register).

Each 8-bit wide port input/output control register controls two GPIO port lines:

- Register SCR_Pn_IOCR0 controls the Pn.0 port lines
- Register SCR_Pn_IOCR1 controls the Pn.1 port lines
- Register SCR_Pn_IOCR2 controls the Pn.2 port lines
- Register SCR_Pn_IOCR3 controls the Pn.3 port lines
- Register SCR_Pn_IOCR4 controls the Pn.4 port lines
- Register SCR_Pn_IOCR5 controls the Pn.5 port lines
- Register SCR_Pn_IOCR6 controls the Pn.6 port lines
- Register SCR_Pn_IOCR7 controls the Pn.7 port lines

The diagrams below show the register layouts of the port input/output control registers with the PCx bit fields. One PCx bit field controls exactly one port line Pn.x.

Note: The reset values of 10_H and 00_H for SCR_Pn_IOCRx registers represents input pull-up and no input pull device (tri-state mode) being activated, respectively. The switching of the intended mode of the device is controlled by HWCFG6. When a cold reset is activated and HWCFG6=1, the pins are set to input pull-up mode as long as PORST is activated. If HWCFG6=0, the pins have the default state of tri-state mode. The pad state can also be configured by software through PMSWCR0.TRISTREQ bit. In the event of a warm reset or wake-up from standby mode, PMSWCR0.TRISTREQ is not affected by reset, hence SCR_Pn_IOCRx registers have the reset values configured as per the last state of the TRISTREQ bit.

SCR_P00_IOCR0

Port 00 Input/Output Control Register 0

(93_H)

Reset Value: 10_H

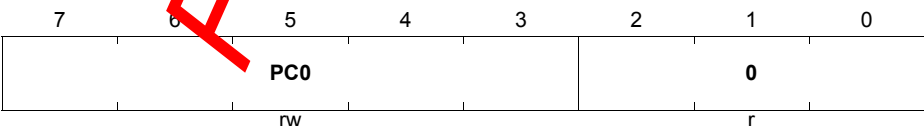
SCR_P02_IOCR0

Port 02 Input/Output Control Register 0

(CC_H)

Reset Value: 10_H

RMAP: 0, PAGE: X



Field	Bits	Type	Description
PC0	[7:3]	rw	Port Control for Port n Pin 0 This bit field determines the Port n line 0 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCRO
Port 00 Input/Output Control Register 0

 (93_H)

Reset Value: 00_H
SCR_P02_IOCRO
Port 02 Input/Output Control Register 0

 (CC_H)

Reset Value: 00_H
RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
			PC0			0	
			rw			r	

Field	Bits	Type	Description
PC0	[7:3]	rw	Port Control for Port n Pin 0 This bit field determines the Port n line 0 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

Feature Discontinued

SCR_P00_IOCR1

Port 00 Input/Output Control Register 1

 (94_H)

 Reset Value: 10_H
SCR_P02_IOCR1

Port 02 Input/Output Control Register 1

 (CD_H)

 Reset Value: 10_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC1						0	
rw						r	

Field	Bits	Type	Description
PC1	[7:3]	rw	Port Control for Port n Pin 1 This bit field determines the Port n line 1 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR1

Port 00 Input/Output Control Register

 (94_H)

 Reset Value: 00_H
SCR_P02_IOCR1

Port 02 Input/Output Control Register 1

 (CD_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC1						0	
rw						r	

Field	Bits	Type	Description
PC1	[7:3]	rw	Port Control for Port n Pin 1 This bit field determines the Port n line 1 functionality according to the coding table (see Table 13-39).

Field	Bits	Type	Description
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR2

Port 00 Input/Output Control Register 2

 (9A_H)

 Reset Value: 10_H
SCR_P02_IOCR2

Port 02 Input/Output Control Register 2

 (CE_H)

 Reset Value: 10_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC2						0	
rw						r	

Field	Bits	Type	Description
PC2	[7:3]	rw	Port Control for Port n Pin 2 This bit field determines the Port n line 2 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR2

Port 00 Input/Output Control Register 2

 (9A_H)

 Reset Value: 00_H
SCR_P02_IOCR2

Port 02 Input/Output Control Register 2

 (CE_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC2						0	
rw						r	

Field	Bits	Type	Description
PC2	[7:3]	rw	Port Control for Port n Pin 2 This bit field determines the Port n line 2 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR3

Port 00 Input/Output Control Register 3

 (9B_H)

 Reset Value: 10_H
SCR_P02_IOCR3

Port 02 Input/Output Control Register 3

 (CF_H)

 Reset Value: 10_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC3					0		
rw					r		

Field	Bits	Type	Description
PC3	[7:3]	rw	Port Control for Port n Pin 3 This bit field determines the Port n line 3 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR3

Port 00 Input/Output Control Register 3

 (9B_H)

 Reset Value: 00_H
SCR_P02_IOCR3

Port 02 Input/Output Control Register 3

 (CF_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC3					0		
rw					r		

Field	Bits	Type	Description
PC3	[7:3]	rw	Port Control for Port n Pin 3 This bit field determines the Port n line 3 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR4
Port 00 Input/Output Control Register 4

 (9C_H)

Reset Value: 10_H
SCR_P02_IOCR4
Port 02 Input/Output Control Register 4

 (EC_H)

Reset Value: 10_H
RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
			PC4			0	
			rw			r	

Field	Bits	Type	Description
PC4	[7:3]	rw	Port Control for Port n Pin 4 This bit field determines the Port n line 4 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

Feature Discontinued

SCR_P00_IOCR4

Port 00 Input/Output Control Register 4

 (9C_H)

 Reset Value: 00_H
SCR_P02_IOCR4

Port 02 Input/Output Control Register 4

 (EC_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC4						0	
rw						r	

Field	Bits	Type	Description
PC4	[7:3]	rw	Port Control for Port n Pin 4 This bit field determines the Port n line 4 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR5

Port 00 Input/Output Control Register 5

 (9D_H)

 Reset Value: 10_H
SCR_P02_IOCR5

Port 02 Input/Output Control Register 5

 (ED_H)

 Reset Value: 10_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC5						0	
rw						r	

Field	Bits	Type	Description
PC5	[7:3]	rw	Port Control for Port n Pin 5 This bit field determines the Port n line 5 functionality according to the coding table (see Table 13-39).

Field	Bits	Type	Description
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR5

Port 00 Input/Output Control Register 5

 (9D_H)

 Reset Value: 00_H
SCR_P02_IOCR5

Port 02 Input/Output Control Register 5

 (ED_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC5					0		
rw					r		

Field	Bits	Type	Description
PC5	[7:3]	rw	Port Control for Port n Pin 5 This bit field determines the Port n line 5 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR6

Port 00 Input/Output Control Register 6

 (9E_H)

 Reset Value: 10_H
SCR_P02_IOCR6

Port 02 Input/Output Control Register 6

 (EE_H)

 Reset Value: 10_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC6					0		
rw					r		

Field	Bits	Type	Description
PC6	[7:3]	rw	Port Control for Port n Pin 6 This bit field determines the Port n line 6 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR6
Port 00 Input/Output Control Register 6

 (9E_H)

Reset Value: 00_H
SCR_P02_IOCR6
Port 02 Input/Output Control Register 6

 (EE_H)

Reset Value: 00_H
RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC6					0		
rw					r		

Field	Bits	Type	Description
PC6	[7:3]	rw	Port Control for Port n Pin 6 This bit field determines the Port n line 6 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR7
Port 00 Input/Output Control Register 7

 (9F_H)

Reset Value: 10_H
SCR_P02_IOCR7
Port 02 Input/Output Control Register 7

 (EF_H)

Reset Value: 10_H
RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PC7					0		
rw					r		

Field	Bits	Type	Description
PC7	[7:3]	rw	Port Control for Port n Pin 7 This bit field determines the Port n line 7 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

SCR_P00_IOCR7

Port 00 Input/Output Control Register 7

 (9F_H)

 Reset Value: 00_H
SCR_P02_IOCR7

Port 02 Input/Output Control Register 7

 (EF_H)

 Reset Value: 00_H
RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
		PC7				0	
		rw				r	

Field	Bits	Type	Description
PC7	[7:3]	rw	Port Control for Port n Pin 7 This bit field determines the Port n line 7 functionality according to the coding table (see Table 13-39).
0	[2:0]	r	Reserved Read as 0; should be written with 0.

Port Control Coding

[Table 13-39](#) describes the coding of the PCx bit fields that determine the port line functionality.

Feature Discontinued

Table 13-39 PCx Coding

PCx[4:0]	I/O	Output Characteristics	Selected Pull-up / Pull-down / Selected Output Function
0XX00 _B	Input	–	No input pull-up device connected, tri-state mode
0XX01 _B			Input pull-down device connected
0XX10 _B			Input pull-up device connected
0XX11 _B			No input pull-down device connected, tri-state mode
10000 _B	Output	Push-pull	General-purpose output
10001 _B			Alternate output function 1
10010 _B			Alternate output function 2
10011 _B			Alternate output function 3
10100 _B			Alternate output function 4
10101 _B			Alternate output function 5
10110 _B			Alternate output function 6
10111 _B			Alternate output function 7
11000 _B		Open-drain	General-purpose output
11001 _B			Alternate output function 1
11010 _B			Alternate output function 2
11011 _B			Alternate output function 3
11100 _B			Alternate output function 4
11101 _B			Alternate output function 5
11110 _B			Alternate output function 6
11111 _B			Alternate output function 7

13.9.3.2 Pad Driver Mode Register

Overview

The pad structure of the TC26x_SCR GPIO lines offers the possibility to select the output driver strength and the slew rate. These two parameters are controlled by the PDx bit fields in the pad driver mode registers SCR_Pn_PDRy (y=0, 2, 4, 6), independently from input/output and pull-up/pull-down control functionality as programmed in the Pn_IOCR register. SCR_Pn_PDRy registers are assigned to each port.

The assignment of each port pin to one of these pad classes is shown in the port configuration figures. Further details about pad driver classes that are available in the TC26x B-Step are summarized in the Target Data Sheet/Data Sheet.

Depending on the assigned pad class, the 3-bit wide pad driver mode selection bit fields PDx in the pad driver mode registers SCR_Pn_PDRy make it possible to select the port line functionality as shown in [Table 13-40](#).

Table 13-40 Pad Driver Mode Selection

PDx.2	PDx.1	PDx.0	Functionality
X	0	0	Speed grade 1
X	0	1	Speed grade 2
X	1	0	Speed grade 3
X	1	1	Speed grade 4

Note: TC26x B-Step Data Sheet describes the DC characteristics of all pad classes.

Depending on whether it is 3.3V- or 5V-pad, the pad level is CMOS or automotive level, respectively. The TTL level can be enabled for both for 3.3V and 5V pads through the PLx bit in the SCR_Pn_PDRy. The coding is shown in [Table 13-41](#).

Table 13-41 Pad Level Selection

PLx	3.3V-Pad	5V-Pad
0	CMOS level	Automotive level
1	TTL level	TTL level

Pad Driver Mode Registers

This is the general description of the PDR registers. Each port contains its own specific PDR registers, described additionally at each port, that can contain between one and eight PDx fields for PDRy (y = 0, 2, 4, 6) registers, respectively. Each PDx field controls 1 pin. For coding of PDx, see [Table 13-40](#). Similarly, each PLx bit controls 1 pin. For coding of PLx, see [Table 13-41](#).

SCR_P00_PDR0

Port 00 Pad Driver Mode 0 Register

 (A1_H)

 Reset Value: 00_H
SCR_P02_PDR0

Port 02 Pad Driver Mode 0 Register

 (D1_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PL1	PD1		PL0		PD0		
rw	rw		rw		rw		

Field	Bits	Type	Description
PD0, PD1	[2:0], [6:4]	rw	Pad Driver Mode for Port n Pin 0 to 1
PL0, PL1	3, 7	rw	Pad Level Selection for Port n Pin 0 to 1

SCR_P00_PDR2

Port 00 Pad Driver Mode 2 Register

 (A3_H)

 Reset Value: 00_H
SCR_P02_PDR2

Port 02 Pad Driver Mode 2 Register

 (D3_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PL3	PD3		PL2		PD2		
rw	rw		rw		rw		

Field	Bits	Type	Description
PD2, PD3	[2:0], [6:4]	rw	Pad Driver Mode for Port n Pin 2 to 3
PL2, PL3	3, 7	rw	Pad Level Selection for Port n Pin 2 to 3

SCR_P00_PDR4

Port 00 Pad Driver Mode 4 Register

 (A4_H)

 Reset Value: 00_H
SCR_P02_PDR4

Port 02 Pad Driver Mode 4 Register

 (D4_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PL5	PD5		PL4		PD4		
rw	rw		rw		rw		

Field	Bits	Type	Description
PD4, PD5	[2:0], [6:4]	rw	Pad Driver Mode for Port n Pin 4 to 5
PL4, PL5	3, 7	rw	Pad Level Selection for Port n Pin 4 to 5

SCR_P00_PDR6

Port 00 Pad Driver Mode 6 Register

 (5_H)

 Reset Value: 00_H
SCR_P02_PDR6

Port 02 Pad Driver Mode 6 Register

 (D5_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PL7	PD7		PL6		PD6		
rw	rw		rw		rw		

Field	Bits	Type	Description
PD6, PD7	[2:0], [6:4]	rw	Pad Driver Mode for Port n Pin 6 to 7
PL6, PL7	3, 7	rw	Pad Level Selection for Port n Pin 6 to 7

13.9.3.3 Port Output Register

The port output register determines the value of a GPIO pin when it is selected by SCR_Pn_IOCRx as output. Writing a 0 to a SCR_Pn_OUT_Px (x = 0-7) bit position delivers a low level at the corresponding output pin. A high level is output when the corresponding bit is written with a 1. Note that the bits of SCR_Pn_OUT.Px can be individually set or cleared by writing appropriate values into the port output modification set register SCR_Pn_OMSR or port output modification clear register SCR_Pn_OMCR, respectively. The SCR_Pn_OUT.Px bits can also be set, cleared or toggled with register SCR_Pn_OMR within the same write operation.

SCR_P00_OUT

Port 00 Output Register (90_H) Reset Value: 00_H

SCR_P02_OUT

Port 02 Output Register (C8_H) Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
Px (x = 0-7)	x	rwh	<p>Port Output Bit x</p> <p>This bit determines the level at the output pin Pn.x if an output is selected as GPIO output.</p> <p>0_B The output level of Pn.x is 0.</p> <p>1_B The output level of Pn.x is 1.</p> <p>Pn.x can also be set or cleared by control bits of the SCR_Pn_OMSR, SCR_Pn_OMCR or SCR_Pn_OMR registers.</p>

Feature Discontinued

13.9.3.4 Port Output Modification Register

The port output modification registers contain control bits that make it possible to individually set, clear or toggle the logic state of a single port pin by manipulating the output register.

SCR_P00_OMR0

Port 00 Output Modification Register 0

(91_H)

Reset Value: 00_H

SCR_P02_OMR0

Port 02 Output Modification Register 0

(CA_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PCL3	PCL2	PCL1	PCL0	PS3	PS2	PS1	PS0
w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x = 0-3)	x	w	Port Pin Set Bit x Setting this bit will set or toggle the corresponding bit in the port output register SCR_Pn_OUT. Read as 0. The function of this bit is shown in Table 13-42 . 0 _B No operation 1 _B Sets or toggles SCR_Pn_OUT.Px.
PCLx (x = 0-3)	x + 4	w	Port Pin Clear Bit x Setting this bit will clear or toggle the corresponding bit in the port output register SCR_Pn_OUT. Read as 0. The function of this bit is shown in Table 13-42 . 0 _B No operation 1 _B Clears or toggles SCR_Pn_OUT.Px.

Feature Discontinued

SCR_P00_OMR1
Port 00 Output Modification Register 1

(92_H)

Reset Value: 00_H

SCR_P02_OMR1
Port 02 Output Modification Register 1

(CB_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PCL7	PCL6	PCL5	PCL4	PS7	PS6	PS5	PS4
W	W	W	W	W	W	W	W

Field	Bits	Type	Description
PSx (x = 4-7)	x - 4	w	Port n Set Bit x Setting this bit will set or toggle the corresponding bit in the port output register SCR_Pn_OUT. Read as 0. The function of this bit is shown in Table 13-42 . 0 _B No operation 1 _B Sets or toggles SCR_Pn_OUT.Px.
PCLx (x = 4-7)	x	w	Port n Clear Bit x Setting this bit will clear or toggle the corresponding bit in the port output register SCR_Pn_OUT. Read as 0. The function of this bit is shown in Table 13-42 . 0 _B No operation 1 _B Clears or toggles SCR_Pn_OUT.Px.

Table 13-42 Function of the Bits PCLx and PSx

PCLx	PSx	Function
0	0	Bit SCR_Pn_OUT.Px is not changed.
0	1	Bit SCR_Pn_OUT.Px is set.
1	0	Bit SCR_Pn_OUT.Px is reset.
1	1	Bit SCR_Pn_OUT.Px is toggled.

13.9.3.5 Port Output Modification Set Register

The port output modification set register contains control bits that make it possible to individually set the logic state of a single port line by manipulating the output register.

SCR_P00_OMSR

Port 00 Output Modification Set Register

(BD_H)

Reset Value: 00_H

SCR_P02_OMSR

Port 02 Output Modification Set Register

(DD_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0
W	W	W	W	W	W	W	W

Field	Bits	Type	Description
PSx (x = 0-7)	x	w	Port n Set Bit x Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0 _B No operation 1 _B Sets Pn_OUT.Px

13.9.3.6 Port Output Modification Set Registers x

The port output modification set register x, (x = 0, 4) contains control bits to individually set the logic state of a single port line by manipulating the output register.

Register SCR_Pn_OMSR0 sets the logic state of Pn.[3:0] port lines

Register SCR_Pn_OMSR4 sets the logic state of Pn.[7:4] port lines

SCR_P00_OMSR0

Port 00 Output Modification Set Register 0

(A9_H)

Reset Value: 00_H

SCR_P02_OMSR0

Port 02 Output Modification Set Register 0

(D9_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
0				PS3	PS2	PS1	PS0
r				w	w	w	w

Field	Bits	Type	Description
PSx (x = 0-3)	x	w	Port n Set Bit x Setting this bit will set the corresponding bit in the port output register SCR_Pn_OUT. Read as 0. 0 _B No operation 1 _B Sets SCR_Pn_OUT.Px
0	[7:4]	r	Reserved Read as 0; should be written with 0.

SCR_P00_OMSR4

Port 00 Output Modification Set Register 4

(BA_H)

Reset Value: 00_H

SCR_P02_OMSR4

Port 02 Output Modification Set Register 4

(DA_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PS7	PS6	PS5	PS4	0	0	0	0
w	w	w	w	r	r	r	r

Field	Bits	Type	Description
PSx (x = 7-4)	x	w	Port n Set Bit x Setting this bit will set the corresponding bit in the port output register SCR_Pn_OU. Read as 0. 0 _B No operation 1 _B Sets SCR_Pn_OU[x] Px
0	[3:0]	r	Reserved Read as 0; should be written with 0.

Feature Discontinued

13.9.3.7 Port Output Modification Clear Register

The port output modification clear register contains control bits that make it possible to individually clear the logic state of a single port line by manipulating the output register.

SCR_P00_OMCR

Port 00 Output Modification Clear Register
(BE_H)

Reset Value: 00_H

SCR_P02_OMCR

Port 02 Output Modification Clear Register
(DE_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PCLx (x = 0-7)	x	w	Port n Clear Bit x Setting this bit will clear the corresponding bit in the port output register SCR_Pn_OUT. Read as 0. 0 _B No operation 1 _B Clears SCR_Pn_OUT.Px.

Feature Discontinued

13.9.3.8 Port Output Modification Clear Registers x

The port output modification clear register x, (x = 0, 4) contains control bits to individually clear the logic state of a single port line by manipulating the output register.

Register SCR_Pn_OMCR0 clears the logic state of Pn.[3:0] port lines

Register SCR_Pn_OMCR4 clears the logic state of Pn.[7:4] port lines

SCR_P00_OMCR0

Port 00 Output Modification Clear Register 0
(BB_H)

Reset Value: 00_H

SCR_P02_OMCR0

Port 02 Output Modification Clear Register 0
(DB_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
	0			PCL3	PCL2	PCL1	PCL0
	r			w	w	w	w

Field	Bits	Type	Description
PCLx (x = 0-3)	x	w	Port n Clear Bit x Setting this bit will clear the corresponding bit in the port output register SCR_Pn_OUT. Read as 0. 0 No operation 1 Clears SCR_Pn_OUT.Px
0	[7:4]	r	Reserved Read as 0; should be written with 0.

Feature Discontinued

SCR_P00_OMCR4

Port 00 Output Modification Clear Register 4

 (BC_H)

 Reset Value: 00_H
SCR_P02_OMCR4

Port 02 Output Modification Clear Register 4

 (DC_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
PCL 7	PCL 6	PCL 5	PCL 4	0			
w	w	w	w	r			

Field	Bits	Type	Description
PCLx (x = 7-4)	x	w	Port n Clear Bit x Setting this bit will clear the corresponding bit in the port output register SCR_Pn_OUT. Read as 0. 0 _B No operation 1 _B Clears SCR_Pn_OUT.Px
0	[3:0]	r	Reserved Read as 0; should be written with 0.

Feature Discontinued

13.9.3.9 Port Input Register

The logic level of a GPIO pin can be read via the read-only port input register SCR_Pn_IN. Reading the SCR_Pn_IN register always returns the current logical value at the GPIO pin independently whether the pin is selected as input or output.

SCR_P00_IN

Port 00 Input Register

(A0_H)

Reset Value: 00_H

SCR_P02_IN

Port 02 Input Register

(D8_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0
rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
Px (x = 0-7)	x	rh	Port Input Bit x This bit indicates the level at the input pin Pn.x. 0 _B The input level of Pn.x is 0. 1 _B The input level of Pn.x is 1.

Feature Discontinued

13.9.4 T2X_SCR functions in Port 00

This section describes the TC2X_SCR functions in Port 00.

13.9.4.1 T2X_SCR Port 00 Configuration

TC2X_SCR has functions in Port 00.0 - P00.7. It can be alternatively used for UART, WCAN, SSC, T0/T1, T2CCU and External Interrupt input and output functions.

13.9.4.2 Port 00 Function Table

Table 13-43 summarizes the TC2X_SCR I/O control selection functions of the 8 Port 00 lines.

Table 13-43 Port 00 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.0	I	General-purpose input	SCR_P00_IN.P0	SCR_P00_ IOCR0.PC0	0XXXX _B
		SSC input	MTSRB		
		External Interrupt 3/ T2CCU input	T2CC0_EXINT3B		
	O	General-purpose output	SCR_P00_OUT.P0	1X000 _B	
		UART output	TXD	1X001 _B	
		SSC output	MTSR	1X010 _B	
		T2CCU output	T2CC0	1X011 _B	
		Reserved	-	1X100 _B	
		Reserved	-	1X101 _B	
		Reserved	-	1X110 _B	
		Reserved	-	1X111 _B	

Feature Discontinued

Table 13-43 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.1	I	General-purpose input	SCR_P00_IN.P1	SCR_P00_IOCR1.PC1	0XXXX _B
		UART input	RXDD		
		WCAN input	WCANRXDD		
		SSC input	MRSTB		
		External Interrupt 4/ T2CCU input	T2CC1EXINT4P		
		T2CCU input	T2EXC		
	O	General-purpose output	SCR_P00_OUT.P1		1X000 _B
		UART output	RXDO		1X001 _B
		SSC output	MRST		1X010 _B
		T2CCU output	T2CC1		1X011 _B
		Reserved	-		1X100 _B
		Reserved	-		1X101 _B
		Reserved	-		1X110 _B
		Reserved	-		1X111 _B
P00.2	I	General-purpose input	SCR_P00_IN.P2	SCR_P00_IOCR2.PC2	0XXXX _B
		SSC input	SCLKB		
		External interrupt 5/ T2CCU input	T2CC2EXINT5B		
		T2CCU input	T2C		
		Reserved	-		
	O	General-purpose output	SCR_P00_OUT.P2		1X000 _B
		UART output	TXD		1X001 _B
		SSC output	SCLK		1X010 _B
		T2CC output	T2CC2		1X011 _B
		Reserved	-		1X100 _B
		Reserved	-		1X101 _B
		Reserved	-		1X110 _B
		Reserved	-		1X111 _B

Table 13-43 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
P00.3	I	General-purpose input	SCR_P00_IN.P3	SCR_P00_IOCR3.PC3	0XXXX _B	
		UART input	RXDE			
		WCAN input	WCANRXDE			
		External interrupt 6/ T2CCU input	T2CC3EXINT6F			
	O	General-purpose output	SCR_P00_OUT.P3			1X000 _B
		UART output	RXDO			1X001 _B
		T2CCU output	T2CC3			1X010 _B
		Reserved	-			1X011 _B
		Reserved	-	1X100 _B		
		Reserved	-	1X101 _B		
		Reserved	-	1X110 _B		
		Reserved	-	1X111 _B		
	P00.4	I	General-purpose input	SCR_P00_IN.P4	SCR_P00_IOCR4.PC4	0XXXX _B
			External interrupt 11	EXINT11		
External interrupt 0			EXINT0C			
T1 input			T1C			
O		General-purpose output	SCR_P00_OUT.P4	1X000 _B		
		T2CCU output	T2CC4	1X001 _B		
		Reserved	-	1X010 _B		
		Reserved	-	1X011 _B		
		Reserved	-	1X100 _B		
		Reserved	-	1X101 _B		
		Reserved	-	1X110 _B		
		Reserved	-	1X111 _B		

Feature Discontinued

Table 13-43 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.				
				Reg./Bit Field	Value			
P00.5	I	General-purpose input	SCR_P00_IN.P5	SCR_P00_IOCR5.PC5	0XXXX _B			
		External interrupt 12	EXINT12					
		External interrupt 1	EXINT1C					
		T0 input	T0C					
	O	General-purpose output	SCR_P00_OUT.P5			1X000 _B		
		T2CCU output	T2CC5			1X001 _B		
		Reserved	-			1X010 _B		
		Reserved	-			1X011 _B		
		Reserved	-			1X100 _B		
		Reserved	-			1X101 _B		
		Reserved	-			1X110 _B		
		Reserved	-			1X111 _B		
	P00.6	I	General-purpose input			SCR_P00_IN.P6	SCR_P00_IOCR6.PC6	0XXXX _B
			External interrupt 5/T2CCU input			T2CC2EXINT5C		
External interrupt 13			EXINT13					
External interrupt 2			EXINT2C					
O		General-purpose output	SCR_P00_OUT.P6	1X000 _B				
		T2CCU output	T2CC2	1X001 _B				
		Reserved	-	1X010 _B				
		Reserved	-	1X011 _B				
		Reserved	-	1X100 _B				
		Reserved	-	1X101 _B				
		Reserved	-	1X110 _B				
		Reserved	-	1X111 _B				

Feature Discontinued

Table 13-43 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Port I/O Control Select. Reg./Bit Field	Value
P00.7	I	General-purpose input	SCR_P00_IN.P7	SCR_P00_1PCR7.PC7	0XXXX _B
		External interrupt 6/ T2CCU input	T2CC3EXINT6C		
		External interrupt 14	EXINT14		
		T2CCU input	T2EXD		
	O	General-purpose output	SCR_P00_OUT.P7		1X000 _B
		T2CCU output	T2CC3		1X001 _B
		Reserved	-		1X010 _B
		Reserved	-		1X011 _B
		Reserved	-		1X100 _B
		Reserved	-		1X101 _B
		Reserved	-		1X110 _B
		Reserved	-		1X111 _B
		Reserved	-		

Feature Discontinued

13.9.4.3 Port 00 Registers

The following registers are available on Port 00.

The Port SFRs are located in the standard memory area (RMAP 7.0).

Table 13-44 Port 00 Registers

Register Short Name	Register Long Name	Address	Description see
SCR_P00_IOCRO	Port 00 Input/Output Control Register 0	83 _H	Page 13-196
SCR_P00_IOCRR1	Port 00 Input/Output Control Register 1	94 _H	Page 13-198
SCR_P00_IOCRR2	Port 00 Input/Output Control Register 2	9A _H	Page 13-199
SCR_P00_IOCRR3	Port 00 Input/Output Control Register 3	9B _H	Page 13-200
SCR_P00_IOCRR4	Port 00 Input/Output Control Register 4	9C _H	Page 13-201
SCR_P00_IOCRR5	Port 00 Input/Output Control Register 5	9D _H	Page 13-202
SCR_P00_IOCRR6	Port 00 Input/Output Control Register 6	9E _H	Page 13-203
SCR_P00_IOCRR7	Port 00 Input/Output Control Register 7	9F _H	Page 13-204
SCR_P00_PDR0	Port 00 Pad Driver Mode 0 Register	A1 _H	Page 13-208
SCR_P00_PDR2	Port 00 Pad Driver Mode 2 Register	A3 _H	Page 13-208
SCR_P00_PDR4	Port 00 Pad Driver Mode 4 Register	A4 _H	Page 13-209
SCR_P00_PDR6	Port 00 Pad Driver Mode 6 Register	A5 _H	Page 13-209
SCR_P00_OUT	Port 00 Output Register	90 _H	Page 13-210
SCR_P00_OMR0	Port 00 Output Modification Register 0	91 _H	Page 13-211
SCR_P00_OMR1	Port 00 Output Modification Register 1	92 _H	Page 13-212

Table 13-44 Port 00 Registers (cont'd)

Register Short Name	Register Long Name	Address	Description see
SCR_P00_OMSR	Port 00 Output Modification Set Register	BD _H	Page 13-213
SCR_P00_OMSR0	Port 00 Output Modification Set Register 0	AS _H	Page 13-214
SCR_P00_OMSR4	Port 00 Output Modification Set Register 4	BA _H	Page 13-214
SCR_P00_OMCR	Port 00 Output Modification Clear Register	BE _H	Page 13-216
SCR_P00_OMCR0	Port 00 Output Modification Clear Register 0	BB _H	Page 13-217
SCR_P00_OMCR4	Port 00 Output Modification Clear Register 4	BC _H	Page 13-218
SCR_P00_IN	Port 00 Input Register	A0 _H	Page 13-219

Feature Discontinued

13.9.5 T2X_SCR functions in Port 02

This section describes the TC2X_SCR functions in Port 02.

13.9.5.1 T2X_SCR Port 02 Configuration

TC2X_SCR has functions in Port 02.0 - P02.7. It can be alternatively used for Debug, UART, WCAN, SSC, T0/T1, T2CCU and External Interrupt input and output functions.

13.9.5.2 Port 02 Function Table

Table 13-43 summarizes the TC2X_SCR I/O control selection functions of the 8 Port 02 lines.

Table 13-45 Port 02 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
P02.0	I	General-purpose input	SCR_P02_IN.P0	SCR_P02_IOCR0.PC0	0XXXX _B	
		SSC input	MTSR0			
		External Interrupt 3/ T2CCU input	T2CCU_EXINT3A			
		External interrupt 0	EXTINT0A			
		T1 input	T1A			
		External NMI	EXTNMI			
	O	General-purpose output	SCR_P02_OUT.P0			1X000 _B
		UART output	TXD			1X001 _B
		SSC output	MTSR	1X010 _B		
		T2CCU output	T2CC0	1X011 _B		
		Reserved	-	1X100 _B		
		Reserved	-	1X101 _B		
		Reserved	-	1X110 _B		
		Reserved	-	1X111 _B		

Feature Discontinued

Table 13-45 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.1	I	General-purpose input	SCR_P02_IN.P1	SCR_P02_ICCR1.PC1	0XXXX _B
		UART input	RXDA		
		WCAN input	WCANRXDA		
		SSC input	MRSTC		
		External Interrupt 4/ T2CCU input	T2CC1EXINT4A		
		External interrupt 1	EXINT1A		
		T0 input	T0A		
		T2CCU input	T2EXA		
	O	General-purpose output	SCR_P02_OUT.P1	1X000 _B	
		UART output	RXDA	1X001 _B	
		SSC output	MRSTC	1X010 _B	
		T2CCU output	T2CC1	1X011 _B	
		Reserved	-	1X100 _B	
		Reserved	-	1X101 _B	
		Reserved	-	1X110 _B	
Reserved	-	1X111 _B			

Feature Discontinued

Table 13-45 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
P02.2	I	General-purpose input	SCR_P02_IN.P2	SCR_P02_ IOCR2.PC2	0XXXX _B	
		External interrupt 5/ T2CCU input	T2CC2EXINT5A			
		External interrupt 2	EXINT2A			
		Debug input 0	DAP0_1			
	O	General-purpose output	SCR_P02_OUT.P2		1X000 _B	
		UART output	TXD		1X001 _B	
		T2CC output	T2CC2		1X010 _B	
		T2CC output	EXF2		1X011 _B	
		Reserved	-		1X100 _B	
		Reserved	-		1X101 _B	
		Reserved	-		1X110 _B	
		Reserved	-		1X111 _B	
	P02.3	I	General-purpose input	SCR_P02_IN.P3	SCR_P02_ IOCR3.PC3	0XXXX _B
			UART input	RXDB		
WCAN input			WCANRXDB			
External interrupt 6/ T2CCU input			T2CC3EXINT6A			
T2CCU input			T2A			
Debug input 1			DAP1/SPD_1			
O			General-purpose output	SCR_P02_OUT.P3		
		UART output	RXDO		1X001 _B	
		T2CCU output	T2CC3		1X010 _B	
		Reserved	-		1X011 _B	
		Reserved	-		1X100 _B	
		Reserved	-		1X101 _B	
		Reserved	-		1X110 _B	
Reserved		-		1X111 _B		
DIR	Debug output 1; ENx	DAP1/SPD_1		HW_OUT		

Table 13-45 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.4	I	General-purpose input	SCR_P02_IN.P4	SCR_P02_ICCR4.PC4	0XXXX _B
		UART input	RXDC		
		WCAN input	WCANRXDC		
		SSC input	SCLKC		
		External Interrupt 7	EXINT7		
		External interrupt 0	EXINT0B		
		T1 input	T1B		
	O	General-purpose output	SCR_P02_OUT.P4		1X000 _B
		UART output	RXDO		1X001 _B
		SSC output	SCLC		1X010 _B
		T2CCU output	T2PC4		1X011 _B
		Reserved	-		1X100 _B
		Reserved	-		1X101 _B
		Reserved	-		1X110 _B
Reserved	-		1X111 _B		

Feature Discontinued

Table 13-45 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.5	I	General-purpose input	SCR_P02_IN.P5	SCR_P02_IOCR5.PC5	0XXXX _B
		WCAN input	SYNCA		
		SSC input	MRSTA		
		External interrupt 8	EXINT8		
		External interrupt 1	EXINT1B		
		T0 input	T0B		
		T2CCU input	T2EXB		
	O	General-purpose output	SCR_P02_OUT.P5	1X000 _B	
		UART output	TXD	1X001 _B	
		SSC output	MRST	1X010 _B	
		T2CCU output	T2PC5	1X011 _B	
		Reserved	-	1X100 _B	
		Reserved	-	1X101 _B	
		Reserved	-	1X110 _B	
Reserved	-	1X111 _B			

Feature Discontinued

Table 13-45 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.6	I	General-purpose input	SCR_P02_IN.P6	SCR_P02_ICCR6.PC6	0XXXX _B
		SSC input	MTSRA		
		External interrupt 3/ T2CCU input	T2CC0EXINT3C		
		External interrupt 9	EXINT9		
		External interrupt 2	EXINT2B		
		Debug input	DAP0_0		
	O	General-purpose output	SCR_P02_OUT.P6	1X000 _B	
		SSC output	MTSR	1X001 _B	
		T2CCU output	T2CC0	1X010 _B	
		T2CCU output	EXE2	1X011 _B	
		Reserved	-	1X100 _B	
		Reserved	-	1X101 _B	
		Reserved	-	1X110 _B	
		Reserved	-	1X111 _B	

Feature Discontinued

Table 13-45 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.7	I	General-purpose input	SCR_P02_IN.P7	SCR_P02_ICCR7.PC7	0XXXX _B
		WCAN input	SYNCB		
		SSC input	SCLKA		
		External interrupt 4/ T2CCU input	T2CC1EXINT10C		
		External interrupt 10	EXINT10		
		T2CCU input	T2B		
		Debug input	DAP1/SPD_0		
	O	General-purpose output	SCR_P02_OUT.P7		1X000 _B
		SSC input	SCLKA		1X001 _B
		T2CCU output	T2CC1		1X010 _B
		Reserved	-		1X011 _B
		Reserved	-		1X100 _B
		Reserved	-		1X101 _B
		Reserved	-		1X110 _B
	Reserved	-		1X111 _B	
DIR	Debug output 1; ENX	DAP1/SPD_0	HW_OUT		

Feature Discontinued

13.9.5.3 Port 02 Registers

The following registers are available on Port 02.

The Port SFRs are located in the standard memory area (RMAP 7.0).

Table 13-46 Port 02 Registers

Register Short Name	Register Long Name	Address	Description see
SCR_P02_IOCRO	Port 02 Input/Output Control Register 0	0C _H	Page 13-196
SCR_P02_IOCR1	Port 02 Input/Output Control Register 1	0D _H	Page 13-198
SCR_P02_IOCR2	Port 02 Input/Output Control Register 2	0E _H	Page 13-199
SCR_P02_IOCR3	Port 02 Input/Output Control Register 3	0F _H	Page 13-200
SCR_P02_IOCR4	Port 02 Input/Output Control Register 4	10 _H	Page 13-201
SCR_P02_IOCR5	Port 02 Input/Output Control Register 5	11 _H	Page 13-202
SCR_P02_IOCR6	Port 02 Input/Output Control Register 6	12 _H	Page 13-203
SCR_P02_IOCR7	Port 02 Input/Output Control Register 7	13 _H	Page 13-204
SCR_P02_PDR0	Port 02 Pad Driver Mode 0 Register	D1 _H	Page 13-208
SCR_P02_PDR2	Port 02 Pad Driver Mode 2 Register	D3 _H	Page 13-208
SCR_P02_PDR4	Port 02 Pad Driver Mode 4 Register	D4 _H	Page 13-209
SCR_P02_PDR6	Port 02 Pad Driver Mode 6 Register	D5 _H	Page 13-209
SCR_P02_OUT	Port 02 Output Register	C8 _H	Page 13-210
SCR_P02_OMR0	Port 02 Output Modification Register 0	CA _H	Page 13-211
SCR_P02_OMR1	Port 02 Output Modification Register 1	CB _H	Page 13-212

Table 13-46 Port 02 Registers (cont'd)

Register Short Name	Register Long Name	Address	Description see
SCR_P02_OMSR	Port 02 Output Modification Set Register	DD _H	Page 13-213
SCR_P02_OMSR0	Port 02 Output Modification Set Register 0	D0 _H	Page 13-214
SCR_P02_OMSR4	Port 02 Output Modification Set Register 4	DA _H	Page 13-214
SCR_P02_OMCR	Port 02 Output Modification Clear Register	DE _H	Page 13-216
SCR_P02_OMCR0	Port 02 Output Modification Clear Register 0	DB _H	Page 13-217
SCR_P02_OMCR4	Port 02 Output Modification Clear Register 4	DC _H	Page 13-218
SCR_P02_IN	Port 02 Input Register	D8 _H	Page 13-219

Feature Discontinued

13.10 Timer 0 and Timer 1

13.10.1 Overview

Timer 0 and Timer 1 can function as both timers or counters. When functioning as a timer, Timer 0 and Timer 1 are incremented every machine cycle, i.e. every 2 input clocks (or 2 PCLKs). When functioning as a counter, Timer 0 and Timer 1 are incremented in response to a 1-to-0 transition (falling edge) at their respective external input pins, T0 or T1.

They are useful in many timing applications such as measuring the time interval between events, counting events and generating signals at regular intervals. In particular, Timer 1 can be used as the baud-rate generator for the on-chip serial port.

Features:

- Four operational modes :
 - Mode 0: 13-bit timer/counter
 - Mode 1: 16-bit timer/counter
 - Mode 2: 8-bit timer/counter with auto-reload
 - Mode 3: Two 8-bit timers/counters

13.10.2 System Information

This section provides system information relevant to the Timer 0 and 1.

13.10.2.1 Pinning

Table 13-47 Timer 0 and Timer 1 Pin Functions

Pin	Function	Description	Selected By
P02.0	T1A	Timer 1 Input A	MODPISEL5.T1IS = 00 _B SCR_P02_IOCRO.PC0 = 0XX10 _B
P02.1	T0A	Timer 0 Input A	MODPISEL5.T0IS = 00 _B SCR_P02_IOC1R.PC1 = 0XX10 _B
P02.4	T1B	Timer 1 Input B	MODPISEL5.T1IS = 01 _B SCR_P02_IOC4R.PC4 = 0XX10 _B
P02.5	T0B	Timer 0 Input B	MODPISEL5.T0IS = 01 _B SCR_P02_IOC5R.PC5 = 0XX10 _B

Table 13-47 Timer 0 and Timer 1 Pin Functions (cont'd)

Pin	Function	Description	Selected By
P00.4	T1C	Timer 1 Input C	MODPISEL5.T1IS = 10 _B SCR_P00_00CR4.PC4 = 0XX10 _B
P00.5	T0C	Timer 0 Input C	MODPISEL5.T0IS = 10 _B SCR_P00_10CR5.PC5 = 0XX10 _B

The bit field PAGE of SCU_PAGE register must be programmed before accessing the MODPISEL5 register.

MODPISEL5
Peripheral Input Select Register 5 (F7_H)
Reset Value: 00_H
RMAP: 0, PAGE: 3

7	6	5	4	3	2	1	0
0			T0IS		T1IS		
r			rw		rw		

Field	Bits	Type	Description
T1IS	[1:0]	rw	Timer 1 Input Select 00 _B Timer 1 Input T1A is selected. 01 _B Timer 1 Input T1B is selected. 10 _B Timer 1 Input T1C is selected. 11 _B Reserved
T0IS	[3:2]	rw	Timer 0 Input Select 00 _B Timer 0 Input T0A is selected. 01 _B Timer 0 Input T0B is selected. 10 _B Timer 0 Input T0C is selected. 11 _B Reserved
0	[7:4]		Reserved Returns 0 if read; should be written with 0.

13.10.2.2 Clocking Configuration

The Timer 0 and 1 runs on the PCLK.

13.10.2.3 Interrupt Events and Assignment

Table 13-48 lists the interrupt event sources from the Timer 0 and 1, and the interrupt node assignment for each Timer 0 and 1 interrupt source.

Table 13-48 Timer 0 and 1 Events' Interrupt Node Control

Event	Interrupt Node Enable Bit	Interrupt Node Flag Bit	Vector Address
Timer 0 Overflow	IEN0.ET0	TCON.TF0	0BH
Timer 1 Overflow	IEN0.ET1	TCON.TF1	1BH

13.10.3 Basic Timer Operations

The operations of the two timers are controlled using the Special Function Registers (SFRs) TCON and TMOD. To enable a timer, i.e., allow the timer to run, its control bit TCON.TRx is set. To select the timer input to be either from internal system clock or external pin, the input selector bit TMOD is used.

Note: The "x" (e.g., TCON.TRx) in this chapter denotes either 0 or 1.

Each timer consists of two 8-bit registers, TLx (low byte) and THx (high byte) which defaults to 00_H on reset. Setting or clearing TCON.TRx does not affect the timer registers.

Timer Overflow

When a timer overflow occurs, the timer overflow flag, TCON.TFx, is set, and an interrupt may be raised if the interrupt enable control bit, IEN0.ETx, is set. The overflow flag is automatically cleared when the interrupt service routine is entered.

When Timer 0 operates in mode 3, the Timer 1 control bits, TR1, TF1 and ET1 are reserved for TH0, see [Section 13.10.4.4](#).

External Control

In addition to pure software control, the timers can also be enabled or disabled through external port control. When external port control is used, SFR EXICON0 must first be configured to bypass the edge detection circuitry for EXINTx to allow direct feed-through. When the timer is enabled (TCON.TRx = 1) and TMOD.GATEx is set, the respective timer will only run if the corresponding external interrupt EXINTx = 1. This facilitates pulse width measurements. However, this is not applicable for Timer 1 in mode 3.

If TMOD.GATEx is cleared, the timer reverts to pure software control.

13.10.4 Timer Modes

Timers 0 and 1 are fully compatible and can be configured in four different operating modes, as shown in [Table 13-49](#). The bit field TxM in register TMOD selects the operating mode to be used for each timer.

In modes 0, 1 and 2, the two timers operate independently, but in mode 3, their functions are specialized.

Table 13-49 Timer 0 and Timer 1 Modes

Mode	Operation
0	13-bit timer/counter The timer is essentially an 8-bit counter with a divide-by-32 prescaler. This mode is included solely for compatibility with Intel 8048 devices.
1	16-bit timer/counter The timer registers, TLx and THx, are concatenated to form a 16-bit timer/counter.
2	8-bit timer/counter with auto-reload The timer register TLx is reloaded with a user-defined 8-bit value in THx upon overflow.
3	Timer 0 operates as two 8-bit timers/counters The timer registers, TL0 and TH0, operate as two separate 8-bit counters. Timer 1 is halted and retains its count even if enabled.

Feature Discontinued

13.10.4.1 Mode 0

Putting either Timer 0 or Timer 1 into mode 0 configures it as an 8-bit timer/counter with a divide-by-32 prescaler. **Figure 13-40** shows the mode 0 operation.

In this mode, the timer register is configured as a 13-bit register. As the count rolls over from all 1s to all 0s, it sets the timer overflow flag TFX. The overflow flag TFX can then be used to request an interrupt. The counted input is enabled for the timer when TRx = 1 and either GATEx = 0 or EXINTx = 1 (setting GATEx = 1 allows the timer to be controlled by external input EXINTx to facilitate pulse width measurements). TRx is a control bit in the register TCON; bit GATEx is in register TMOD.

The 13-bit register consists of all the 8 bits of THx and the lower 5 bits of TLx. The upper 3 bits of TLx are indeterminate and should be ignored. Setting the run flag (TRx) does not clear the registers..

Mode 0 operation is the same for Timer 0 and Timer 1.

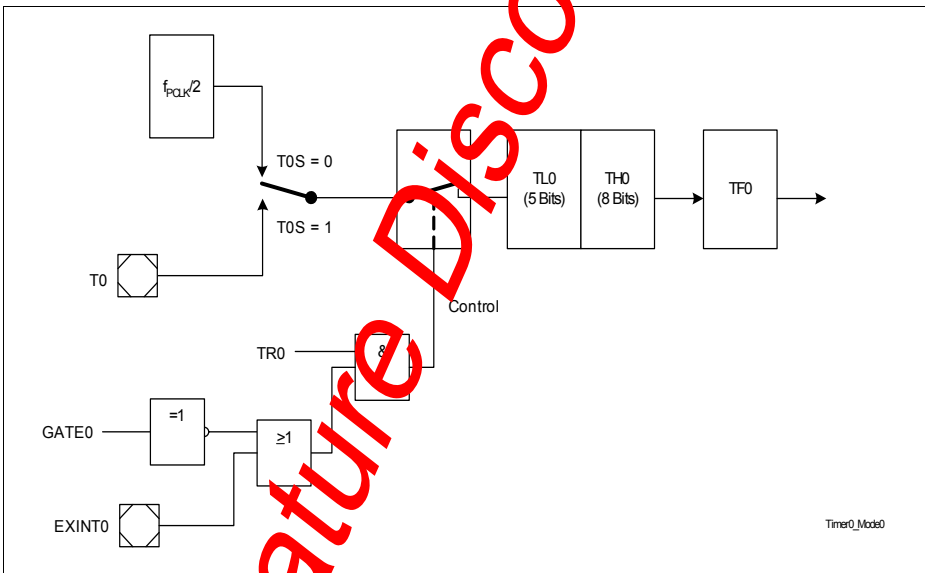


Figure 13-40 Timer 0, Mode 0: 13-Bit Timer/Counter

13.10.4.2 Mode 1

Mode 1 operation is similar to that of mode 0, except that the timer register runs with all 16 bits. Mode 1 operation for Timer 0 is shown in [Figure 13-41](#).

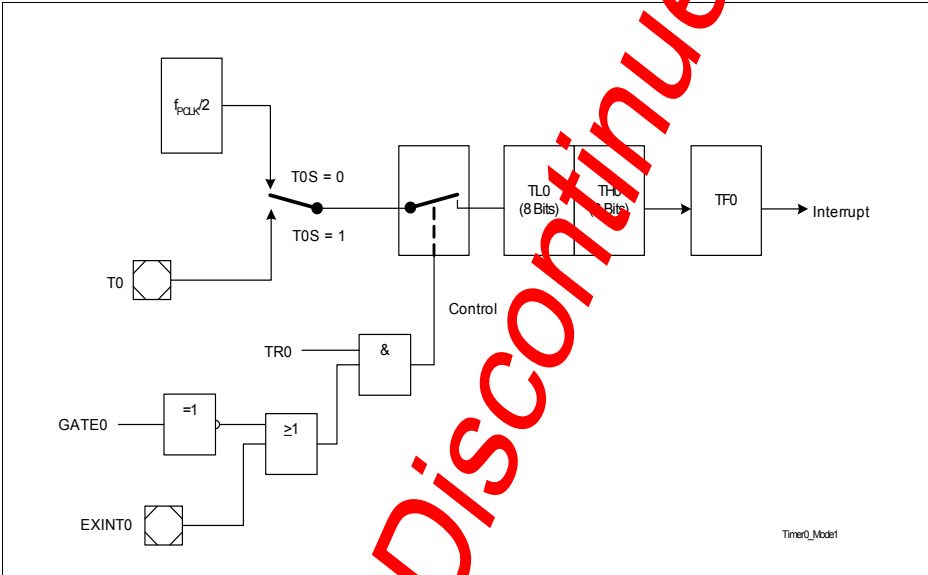


Figure 13-41 Timer 0, Mode 1: 16-Bit Timer/Counter

13.10.4.3 Mode 2

In Mode 2 operation, the timer is configured as an 8-bit counter (TLx) with automatic reload, as shown in **Figure 13-42** for Timer 0.

An overflow from TLx not only sets TFx, but also reloads TLx with the contents of THx that has been preset by software. The reload leaves THx unchanged.

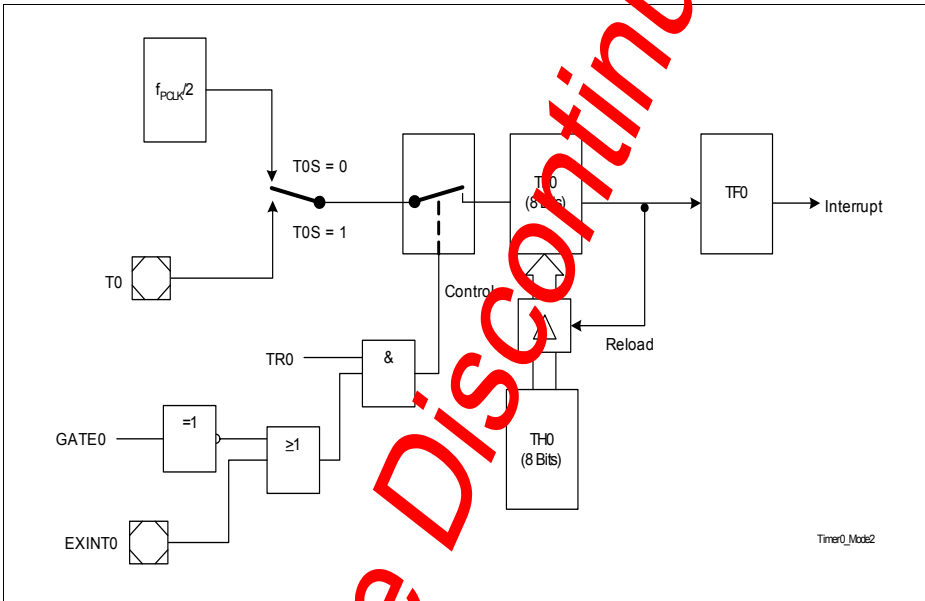


Figure 13-42 Timer 0, Mode 2: 8-bit Timer/Counter with Auto-Reload

Feature Discontinued

13.10.4.4 Mode 3

In mode 3, Timer 0 and Timer 1 behave differently. Timer 0 in mode 3 establishes TL0 and TH0 as two separate counters. Timer 1 in mode 3 simply holds its count. The effect is the same as setting $TR1 = 0$

The logic for mode 3 operation for Timer 0 is shown in [Figure 13-43](#). TL0 uses the Timer 0 control bits GATE0, TR0 and TF0, while TH0 is locked into a timer function (counting machine cycles) and takes over the use of TR1 and TF1 from Timer 1. Thus, TH0 now sets TF1 upon overflow and generates an interrupt if ET1 is set.

Mode 3 is provided for applications requiring an extra 8-bit timer. When Timer 0 is in mode 3 and TR1 is set, Timer 1 can be turned on by switching it to any of the other modes and turned off by switching it into mode 3.

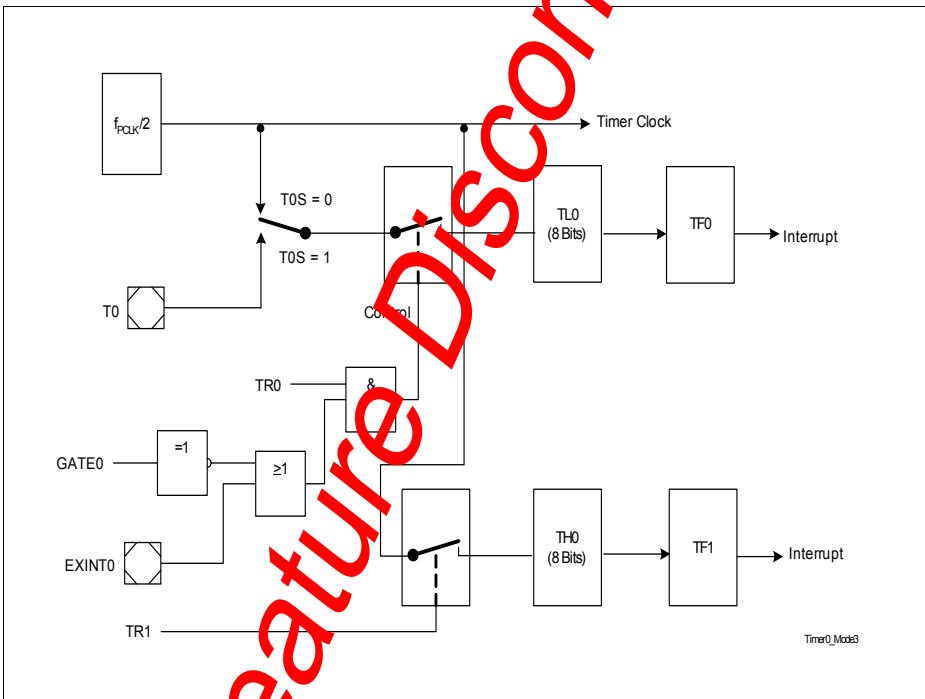


Figure 13-43 Timer 0, Mode 3: Two 8-Bit Timers/Counters

13.10.5 Register Map

Seven SFRs control the operations of Timer 0 and Timer 1. They can be accessed from both the standard (non-mapped) and mapped SFR area.

Table 13-50 lists the addresses of these SFRs.

Table 13-50 Register Map

Address	Register
88 _H	TCON
89 _H	TMOD
8A _H	TL0
8B _H	TL1
8C _H	TH0
8D _H	TH1
A8 _H	IEN0

Feature Discontinued

13.10.6 Register Description

The low bytes (TL0, TL1) and high bytes (TH0, TH1) of both Timer 0 and Timer 1 can be combined to a one-timer configuration depending on the mode used. Register TCON controls the operations of Timer 0 and Timer 1. The operating modes of both timers are selected using register TMOD. Register IEN0 contains bits that enable interrupt operations in Timer 0 and Timer 1.

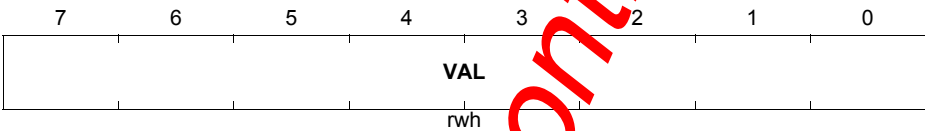
TLx (x = 0 - 1)

Timer x, Low Byte

(8A_H + x)

Reset Value: 00_H

RMAP: X, PAGE: X



Field	Bits	Type	Description
VAL	[7:0]	rwh	Timer 0/1 Low Register 00 _B TLx holds the 5-bit prescaler value. 01 _B TLx holds the lower 8-bit part of the 16-bit timer value. 10 _B TLx holds the 8-bit timer value. 11 _B TL0 holds the 8-bit timer value; TL1 is not used.

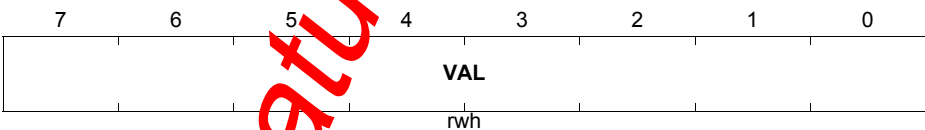
THx (x = 0 - 1)

Timer x, High Byte

(8C_H + x)

Reset Value: 00_H

RMAP: X, PAGE: X



Feature Discontinued

Field	Bits	Type	Description
VAL	[7:0]	rwh	Timer 0/1 High Register 00 _B THx holds the 8-bit timer value. 01 _B THx holds the higher 8-bit part of the 16-bit timer value. 10 _B THx holds the 8-bit reload value. 11 _B TH0 holds the 8-bit timer value; TH1 is not used.

TCON
Timer 0/1 Control Registers
RMAP: X, PAGE: X
(88_H)
Reset Value: 00_H

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
rwh	rw	rwh	rw	rwh	rw	rwh	rw

Field	Bits	Type	Description
TR0	4	rw	Timer 0 Run Control 0 _B Timer is halted 1 _B Timer runs
TF0	5	rwh	Timer 0 Overflow Flag Set by hardware when Timer 0 overflows. Cleared by hardware when the processor calls the interrupt service routine.
TR1	6	rw	Timer 1 Run Control 0 _B Timer is halted 1 _B Timer runs <i>Note: Timer 1 Run Control affects TH0 also if Timer 0 operates in Mode 3</i>
TF1	7	rwh	Timer 1 Overflow Flag Set by hardware when Timer 1 ¹⁾ overflows. Cleared by hardware when the processor calls the interrupt service routine.

1) TF1 is set by TH0 instead if Timer 0 operates in Mode 3.

TMOD
Timer Mode Register
(89_H)
Reset Value: 00_H
RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
GATE1	T1S	T1M		GATE0	T0S	T0M	
rw	rw	rw		rw	rw	rw	

Field	Bits	Type	Description
T0M	[1:0]	rw	Mode Select Bits 00 _B 13-bit timer (M8048 compatible mode) 01 _B 16-bit timer 10 _B 8-bit auto-reload timer 11 _B Timer 0 is split into two halves. TL0 is an 8-bit timer controlled by the standard Timer 0 control bits, and TH0 is the other 8-bit timer controlled by the standard Timer 1 control bits. TH1 and TL1 of Timer 1 are held (Timer 1 is stopped).
T0S	2	rw	Timer 0 Selector 0 _B Input is from internal system clock 1 _B Input is from T0 pin
GATE0	3	rw	Timer 0 Gate Flag 0 Timer 0 will only run if TCON.TR0 = 1 (software control) 1 _B Timer 0 will only run if EXINT0 pin = 1 (hardware control) and TCON.TR0 is set
T1M	[5:4]	rw	Mode Select Bits 00 _B 13-bit timer (M8048 compatible mode) 01 _B 16-bit timer 10 _B 8-bit auto-reload timer 11 _B Timer 0 is split into two halves. TL0 is an 8-bit timer controlled by the standard Timer 0 control bits, and TH0 is the other 8-bit timer controlled by the standard Timer 1 control bits. TH1 and TL1 of Timer 1 are held (Timer 1 is stopped).

Field	Bits	Type	Description
T1S	6	rw	Timer 1 Selector 0 _B Input is from internal system clock 1 _B Input is from T1 pin
GATE1	7	rw	Timer 1 Gate Flag 0 _B Timer 1 will only run if TCON.TR1 = 1 (software control) 1 _B Timer 1 will only run if EXINT0 pin = 1 (hardware control) and TCON.TR1 is set

IEN0
Interrupt Enable Register
(A8_H)
Reset Value: 00_H
RMAP: X, PAGE: X

7	6	5	4	2	1	0	
EA	0	ET2	ES	ET1	EX1	ET0	EX0
rw	r	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ET0	1	rw	Timer 0 Overflow Interrupt Enable 0 _B Timer 0 interrupt is disabled 1 _B Timer 0 interrupt is enabled
ET1	3	rw	Timer 1 Overflow Interrupt Enable 0 _B Timer 1 interrupt is disabled 1 _B Timer 1 interrupt is enabled <i>Note: When Timer 0 operates in Mode 3, this interrupt indicates an overflow in the Timer 0 register, TH0.</i>

Feature Discontinued

Feature Discontinued

13.11 Timer 2

13.11.1 Overview

Timer 2 is a 16-bit general purpose timer which is functionally compatible with the Timer 2 in the C501 product family. Timer 2 can function as a timer or counter in each of its modes. As a timer, it counts with an input clock of PCLK/42 (if prescaler is disabled). As a counter, Timer 2 counts 1-to-0 transitions on pin T2. In the counter mode, the maximum resolution for the count is PCLK/24 (if prescaler is disabled).

Features

- 16-bit auto-reload mode
- selectable up or down counting
- One channel, 16-bit capture mode

13.11.2 System Information

This section provides system information relevant to the Timer 2.

13.11.2.1 Pinning

Timer 2 can be used as an event counter which count 1-to-0 transitions at the external input pin, T2. These pins are selected from three different sources, T2A, T2B and T2C. This selection is performed by the SFR bits MODPISEL3.T2IS. In addition, there are eight sources of T2EX pin for Timer 2. They can be selected by MODPISEL3.T2EXIS.

The Timer 2 pin assignment for T2EX, SCR is shown in [Table 13-51](#).

Table 13-51 Timer 2 Pin Functions

Pins	Function	Description	Selected By
P02.3	T2A	Timer 2 Input	MODPISEL3.T2IS = 00 _B
P02.7	T2B		MODPISEL3.T2IS = 01 _B
P00.2	T2C		MODPISEL3.T2IS = 10 _B

Table 13-51 Timer 2 Pin Functions (cont'd)

Pins	Function	Description	Selected By
P02.1	T2EXA	Timer 2 External Trigger Input	MODPISEL3.T2EXIS = 000 _B
P02.5	T2EXB		MODPISEL3.T2EXIS = 001 _B
P00.1	T2EXC		MODPISEL3.T2EXIS = 010 _B
P00.7	T2EXD		MODPISEL3.T2EXIS = 011 _B
ESR0 ¹⁾	T2EXE		MODPISEL3.T2EXIS = 100 _B
ESR1 ¹⁾	T2EXF		MODPISEL3.T2EXIS = 101 _B
P14.1 ²⁾	T2EXG		MODPISEL3.T2EXIS = 110 _B
P15.1 ²⁾	T2EXH		MODPISEL3.T2EXIS = 111 _B
P02.2	EXF2		Timer 2 Overflow Flag output
P02.6		SCR_P02_IOCR6.PC6 = 1X011 _B	

- 1) The pad control of this pin is in the ESR sub-module (part of 32-bit SCU module). The input function of the pad must be enabled before this pin can be used as T2EX input.
- 2) The pad control of this pin is in the 32-bit Ports module. The input function of the pad must be enabled before this pin can be used as T2EX input.

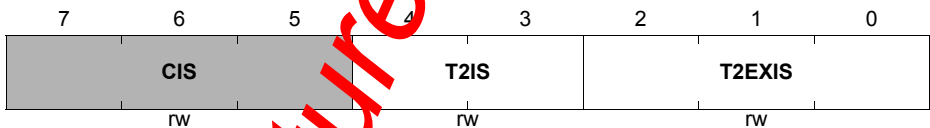
The bit field PAGE of SCU_PAGE register must be programmed before accessing the MODPISEL3 register.

MODPISEL3

Peripheral Input Select Register 3 (r5_H)

Reset Value: 00_H

RMAP: 0, PAGE: 3



Feature Discontinued

Field	Bits	Type	Description
T2EXIS	[2:0]	rw	Timer 2 External Input Select 000 _B Timer 2 Input T2EXA is selected. 001 _B Timer 2 Input T2EXB is selected. 010 _B Timer 2 Input T2EXC is selected. 011 _B Timer 2 Input T2EXD is selected. 100 _B Timer 2 Input T2EXE is selected. 101 _B Timer 2 Input T2EXF is selected. 110 _B Timer 2 Input T2EXG is selected. 111 _B Timer 2 Input T2EXH is selected.
T2IS	[4:3]	rw	Timer 2 Input Select 00 _B Timer 2 Input T2A is selected. 01 _B Timer 2 Input T2B is selected. 10 _B Timer 2 Input T2C is selected. 11 _B Reserved.

13.11.2.2 Clocking Configuration

The Timer 2 runs on the PCLK.

If the Timer 2 functionality is not required at all, it can be completely disabled by gating off its clock input for maximal power reduction. This is done by setting bit T2CCU_DIS in register PMCON1 as described below.

PMCON1

Peripheral Management Control Register 1(F7_H)

Reset Value: 00_H

RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
0				WCAN _DIS	RTC_DIS	T2CCU _DIS	SSC _DIS
r				rw	rw	rw	rw

Field	Bits	Type	Description
T2CCU_DIS	1	rw	T2CCU Disable Request. Active high. 0 _B T2CCU is in normal operation. 1 _B Request to disable the T2CCU. (default)
0	[7:4]	r	Reserved Returns 0 if read; should be written with 0.

13.11.2.3 Interrupt Events and Assignment

Table 13-52 lists the interrupt event sources from the Timer 2, and the corresponding event interrupt enable bit and flag bit.

Table 13-52 Timer 2 Interrupt Events

Event	Event Interrupt Enable Bit	Event Flag Bit
Timer 2 Overflow	T2_T2CON1.TF2EN	T2_T2CON.TF2
Timer 2 External	T2_T2CON1.EXF2EN	T2_T2CON.EXF2

Table 13-53 shows the interrupt node assignment for each Timer 2 interrupt source.

Table 13-53 Timer 2 Events' Interrupt Node Control

Event	Interrupt Node Enable Bit	Interrupt Node Flag Bit	Vector Address
Timer 2 Overflow	IEN0.ET2	–	2BH
Timer 2 External			

13.11.2.4 Module Suspend Control

When the On-Chip Debug Support (OCDS) is in Monitor Mode (MMCR2.MMODE = 1) and the Debug-Suspend signal is active (MMCR2.DSUSP = 1), the timer/counter in Timer 2 module in TC2x_SCR can be suspended based on the settings of their corresponding module suspend bits in register MODSUSP. When suspended, only the timer stops counting as the counter input clock is gated off. The module is still clocked so that module registers are accessible.

13.11.3 Basic Timer Operations

Timer 2 can be started by using TR2 bit by hardware or software. Timer 2 can be started by setting TR2 bit by software. If bit T2RHEN is set, Timer 2 can be started by hardware. Bit T2REGS defines the event on pin T2EX, falling edge or rising edge, that can set the run bit TR2 by hardware. Timer 2 can only be stopped by resetting TR2 bit by software.

13.11.4 Auto-Reload mode

The auto-reload mode is selected when the bit $\overline{CP/RL2}$ in register T2CON is zero. In this mode, Timer 2 counts to an overflow value and then reloads its register contents with a 16-bit start value for a fresh counting sequence. The overflow condition is indicated by setting bit TF2 in the T2CON register. At the same time, an interrupt request to the core will be generated (if interrupt is enabled). The overflow flag TF2 must be cleared by software.

The auto-reload mode is further classified into two categories depending upon the DCEN control bit in register T2MOD.

13.11.4.1 Up/Down Count Disabled

If DCEN = 0, the up-down count selection is disabled. The timer, therefore, functions as a pure up counting timer only. The operational block diagram is shown in [Figure 13-44](#).

If the T2CON register bit EXEN2 = 0, the timer starts to count up to a maximum of $FFFF_H$ once the timer is started by setting the bit TR2 in register T2CON to 1. Upon overflow, bit TF2 is set and the timer register is reloaded with the 16-bit reload value of the RC2 register. This reload value is chosen by software, prior to the occurrence of an overflow condition. A fresh count sequence is started and the timer counts up from this reload value as in the previous count sequence.

If EXEN2 = 1, the timer counts up to a maximum of $FFFF_H$ once TR2 is set. A 16-bit reload of the timer registers from register RC2 is triggered either by an overflow condition or by a negative/positive edge (chosen by the bit EDGESEL in register T2MOD) at input pin T2EX. If an overflow caused the reload, the overflow flag TF2 is set. If a negative/positive transition at pin T2EX caused the reload, bit EXF2 in register T2CON is set. In either case, an interrupt is generated to the core if the related interrupt enable bit EXF2EN/TF2EN in register T2CON1 is enabled and the timer proceeds to its next count sequence. The EXF2 flag, similar to the TF2, must be cleared by software.

If bit T2RHEN is set, Timer 2 is started by first falling edge/rising edge at pin T2EX, which is defined by bit T2REGS. If bit EXEN2 is set, bit EXF2 is also set at the same point when Timer 2 is started with the same falling edge/rising edge at pin T2EX, which is defined by bit EDGESEL. The reload will happen with the following negative/positive transitions at pin T2EX, which is defined by bit EDGESEL.

Note: In counter mode, if the reload via T2EX and the count clock T2 are detected simultaneously, the reload takes precedence over the count. The counter increments its value with the following T2 count clock.

Feature Discontinued

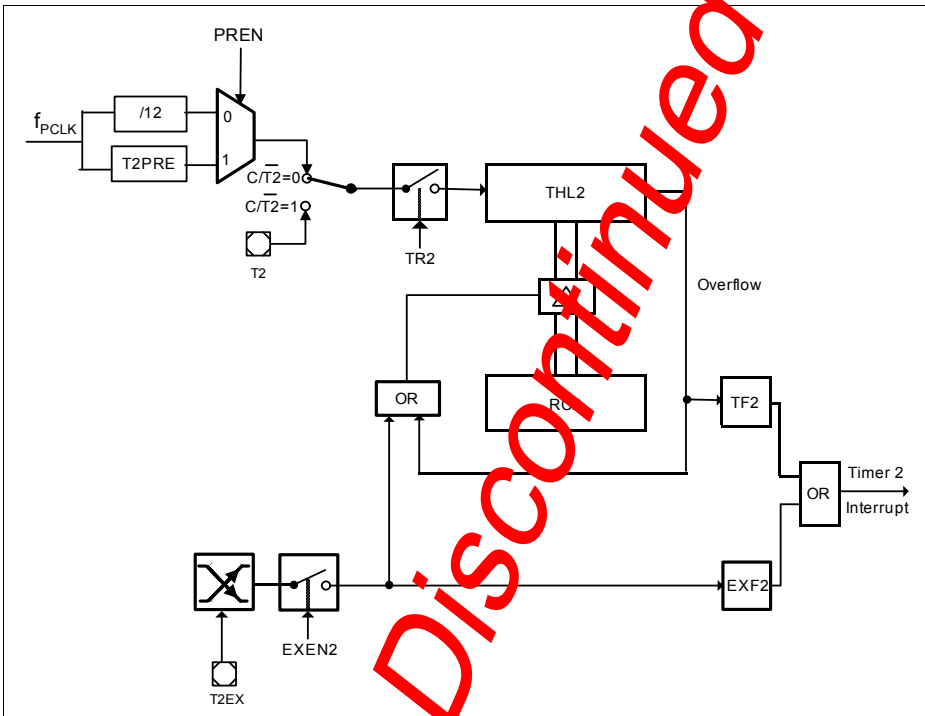


Figure 13-44 Auto-Reload Mode (DCEN = 0)

13.11.4.2 Up/Down Count Enable

If DCEN = 1, the up-down count selection is enabled. The direction of count is determined by the level at input pin T2EX. The operational block diagram is shown in [Figure 13-45](#).

A logic 1 at pin T2EX sets the Timer 2 to up counting mode. The timer, therefore, counts up to a maximum of $FFFF_H$. Upon overflow, bit TF2 is set and the timer register is reloaded with a 16-bit reload value of the RC2 register. A fresh count sequence is started and the timer counts up from this reload value as in the previous count sequence. This reload value is chosen by software, prior to the occurrence of an overflow condition.

A logic 0 at pin T2EX sets the Timer 2 to down counting mode. The timer counts down and underflows when the THL2 value reaches the value stored at register RC2. The underflow condition sets the TF2 flag and causes $FFFF_H$ to be reloaded into the THL2 register. A fresh down counting sequence is started and the timer counts down as in the previous counting sequence.

If bit T2RHEN is set, Timer 2 can only be started either by rising edge (T2REGS = 1) at pin T2EX and then proceed with the up counting, or be started by falling edge (T2REGS = 0) at pin T2EX and then proceed with the down counting.

In this mode, bit EXF2 toggles whenever an overflow or a underflow condition is detected. This flag, however, does not generate an interrupt request.

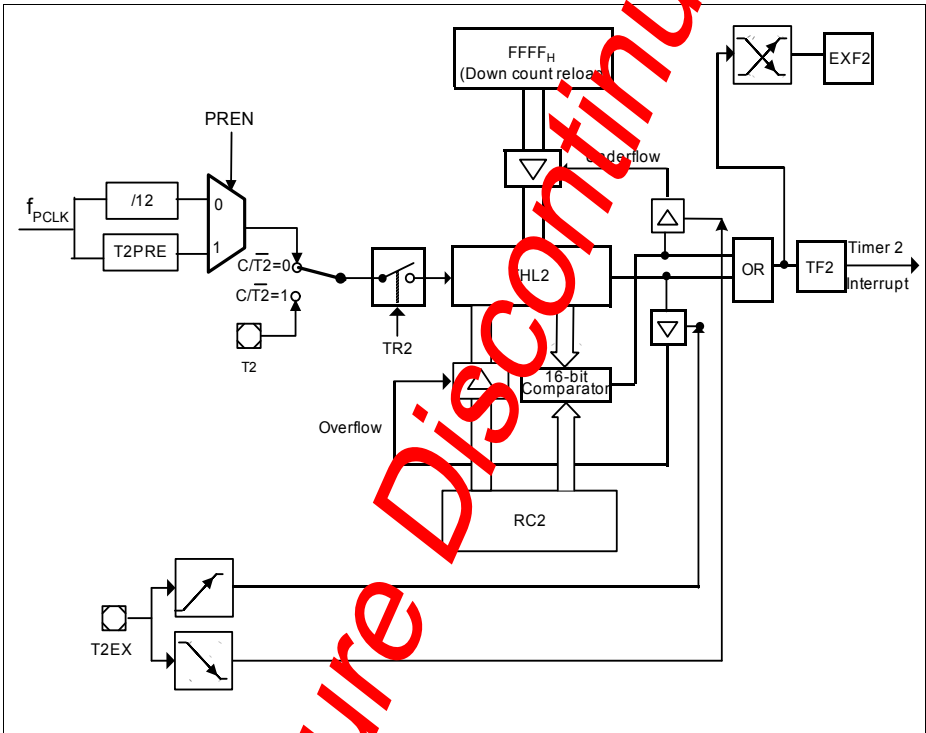


Figure 13-45 Auto-Reload Mode (DCEN = 1)

Feature Discontinued

13.11.5 Capture Mode

In order to enter the 16-bit capture mode, bits CP/RL2 and EXEN2 in register T2CON must be set. In this mode, the down count function must remain disabled. The timer functions as a 16-bit timer and always counts up to $FFFF_H$, after which, an overflow condition occurs. Upon overflow, bit TF2 is set and the timer reloads its registers with 0000_H . The setting of TF2 generates an interrupt request to the core.

Additionally, with a falling/rising edge (chosen by T2MOD_EDGE_SEL) on pin T2EX, the contents of the timer register (THL2) are captured into the RC2 register. The external input is sampled in every PCLK clock cycle. When a sampled input shows a low (high) level in one PCLK clock cycle and a high (low) in the next PCLK clock cycle, a transition is recognized. If the capture signal is detected while the counter is being incremented, the counter is first incremented before the capture operation is performed. This ensures that the latest value of the timer register is always captured.

If bit T2RHEN is set, Timer 2 is started by first falling edge/rising edge at pin T2EX, which is defined by bit T2REGS. If bit EXEN2 is set, bit EXF2 is also set at the same point when Timer 2 is started with the same falling edge/rising edge at pin T2EX, which is defined by bit EDGESEL. The capture will happen with the following negative/positive transitions at pin T2EX, which is defined by bit EDGESEL.

When the capture operation is completed, bit EXF2 is set and can be used to generate an interrupt request. [Figure 13-46](#) describes the capture function of Timer 2.

Feature Discontinued

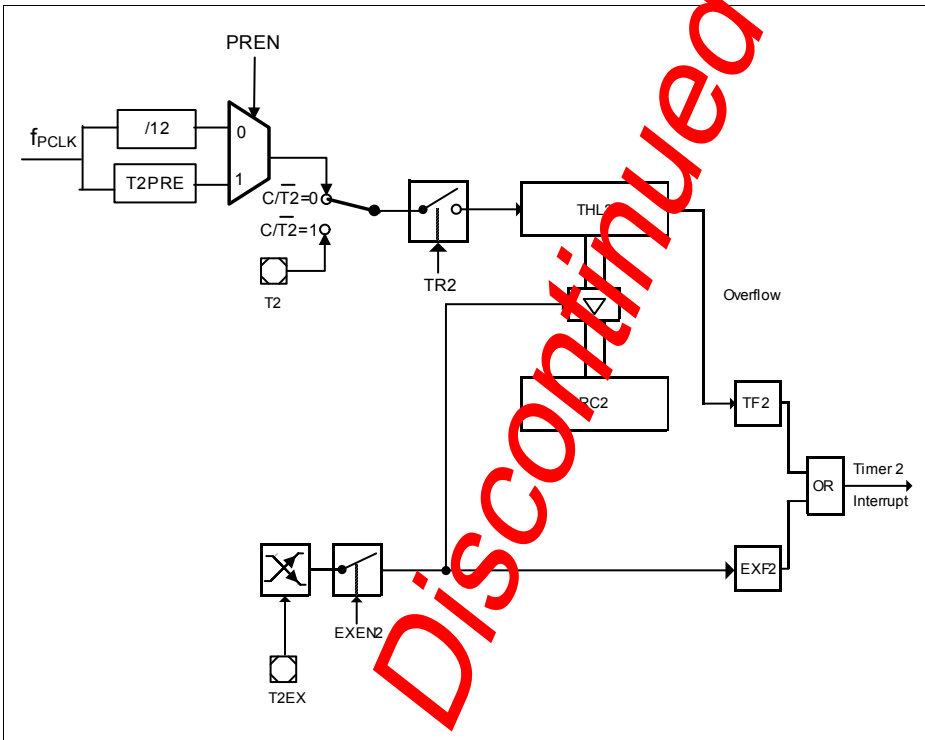


Figure 13-46 Capture Mode

13.11.6 Count Clock

The count clock for the auto-reload mode is chosen by the bit $\overline{C/T2}$ in register T2CON. If $\overline{C/T2} = 0$, a count clock of $f_{PCLK}/12$ (if prescaler is disabled) is used for the count operation.

If $\overline{C/T2} = 1$, Timer 2 behaves as a counter that counts 1-to-0 transitions of input pin T2. The counter samples pin T2 over 2 PCLK clock cycles. If a 1 was detected during the first clock and a 0 was detected in the following clock, then the counter increments by one. Therefore, the input levels should be stable for at least 1 clock.

If bit T2RHEN is set, Timer 2 can be started by the falling edge/rising edge on pin T2EX, which is defined by bit T2REGS.

Note: The C501 compatible feature requires a count resolution of at least 24 clocks.

13.11.7 External Interrupt Function

While the timer/counter function is disabled ($TR2 = 0$), it is still possible to generate a Timer 2 interrupt to the core via an external event at T2EX, as long as Timer 2 remains enabled ($PMCON1.T2CCU_DIS = 0$). To achieve this, bit EXF2 in register T2CON must be set. As a result, any transition on T2EX will cause either a dummy reload or a dummy capture, depending on the CP/ RL2 bit selection.

By disabling the timer/counter function, T2EX can be alternatively used to provide an edge-triggered (rising or falling) external interrupt function, with bit EXF2 serving as the external interrupt flag.

Feature Discontinued

13.11.8 Register Map

All Timer 2 register names described in the following sections are referenced in other chapters of this document with the module name prefix "T2_", e.g. T2_T2CON.

The Timer 2 SFRs are located in the standard (non-mapped) SFR area. [Table 13-54](#) lists these addresses.

Table 13-54 SFR Address List

Address	Register
C0 _H	T2CON
C1 _H	T2MOD
C2 _H	RC2L
C3 _H	RC2H
C4 _H	T2L
C5 _H	T2H
C6 _H	T2CON1

Feature Discontinued

13.11.9 Register Description

13.11.9.1 Mode Register

The T2MOD is used to configure Timer 2 for various modes of operation.

T2_T2MOD

Timer 2 Mode Register

(C1_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
T2REGS	T2RHEN	EDGESEL	PREN		T2PRE		DCEN
rw	rw	rw	rw		rw		rw

Field	Bit	Type	Description
DCEN	0	rw	Up/Down Counter Enable 0 _B Up/Down Counter function is disabled 1 _B Up/Down Counter function is enabled and controlled by pin T2EX (Up = 1, Down = 0)
T2PRE	[3:1]	rw	Timer 2 Prescaler Bit Selects the input clock for Timer 2 which is derived from the peripheral clock. 000 _B $f_{T2} = f_{PCLK}$ 001 _B $f_{T2} = f_{PCLK} / 2$ 010 _B $f_{T2} = f_{PCLK} / 4$ 011 _B $f_{T2} = f_{PCLK} / 8$ 100 _B $f_{T2} = f_{PCLK} / 16$ 101 _B $f_{T2} = f_{PCLK} / 32$ 110 _B $f_{T2} = f_{PCLK} / 64$ 111 _B $f_{T2} = f_{PCLK} / 128$
PREN	4	rw	Prescaler Enable 0 _B Prescaler is disabled and the 2 or 12 divider takes effect. 1 _B Prescaler is enabled (see T2PRE bit) and the 2 or 12 divider is bypassed.
EDGESEL	5	rw	Edge Select in Capture Mode/Reload Mode 0 _B The falling edge at Pin T2EX is selected. 1 _B The rising edge at Pin T2EX is selected.

Field	Bit	Type	Description
T2RHEN	6	rw	Timer 2 External Start Enable 0 _B Timer 2 External Start is disabled. 1 _B Timer 2 External Start is enabled.
T2REGS	7	rw	Edge Select for Timer 2 External Start 0 _B The falling edge at Pin T2EX is selected. 1 _B The rising edge at Pin T2EX is selected.

Feature Discontinued

13.11.9.2 Control Register

Control register T2CON is used to control the operating modes and interrupt of Timer 2. In addition, it contains the status flags for interrupt generation.

T2_T2CON

Timer 2 Control Register

(C0_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
TF2	EXF2	0	0	EXEN2	TR2	C_T2	CP_RL2
rwh	rwh	r	r	rw	rwh	rw	rw

Field	Bit	Type	Description
CP_RL2	0	rw	Capture/Reload Select 0 _B Reload upon overflow or upon negative/positive transition at pin T2EX (when EXEN2 = 1). 1 _B Capture Timer 2 data register contents on the negative/positive transition at pin T2EX, provided EXEN2 = 1. The negative or positive transition at Pin T2EX is selected by bit TR2GESEL.
C_T2	1	rw	Timer or Counter Select 0 _B Timer function selected. 1 _B Count upon negative edge at pin T2.
TR2	2	rwh	Timer 2 Start/Stop Control 0 _B Stop Timer 2. 1 _B Start Timer 2.
EXEN2	3	rw	Timer 2 External Enable Control 0 _B External events are disabled. 1 _B External events are enabled in Capture/Reload Mode.

Feature Discontinued

Field	Bit	Type	Description
EXF2	6	rwh	Timer 2 External Flag In Capture/Reload Mode, this bit is set by hardware when a negative/positive transition occurs at pin T2EX, if bit EXEN2 = 1. This bit must be cleared by software. <i>Note: When bit DCEN = 1 in auto-reload mode, no interrupt request to the core is generated.</i>
TF2	7	rwh	Timer 2 Overflow/Underflow Flag Set by a Timer 2 overflow/underflow. Must be cleared by software.
0	[5:4]	r	Reserved Returns 0 if read; must be written with 0.

Register T2CON1 is used to enable the external interrupt and the overflow interrupt.

T2_T2CON1

Timer 2 Control Register 1
 RMAP: 0, PAGE: X

(06)

Reset Value: 03_H

7	6	5	4	3	2	1	0
0						TF2EN	EXF2EN
r						rw	rw

Field	Bit	Type	Description
EXF2EN	0	rw	External Interrupt Enable 0 _B External interrupt is disabled. 1 _B External interrupt is enabled.
TF2EN	1	rw	Overflow/Underflow Interrupt Enable 0 _B Overflow/underflow interrupt is disabled. 1 _B Overflow/underflow interrupt is enabled.
0	[7:2]	r	Reserved Returns 0 if read; should be written with 0.

13.11.9.3 Timer 2 Reload/Capture Register

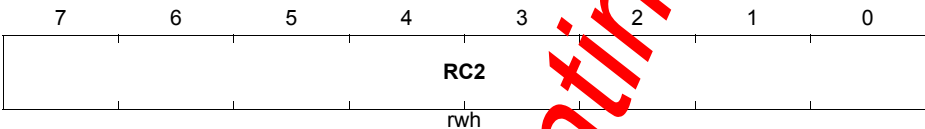
The RC2 register is used for a 16-bit reload of the timer count upon overflow or a capture of current timer count depending on the mode selected.

T2_RC2L

Timer 2 Reload/Capture Register, Low Byte(C2_H)

Reset Value: 00_H

RMAP: 0, PAGE: X



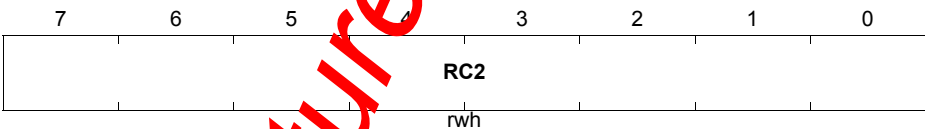
Field	Bit	Type	Description
RC2	[7:0]	rwh	Reload/Capture Value [7:0] If CP/RL2 = 0, these contents are loaded into the timer register upon an overflow condition. If CP/RL2 = 1, this register is loaded with the current timer count upon a negative/positive transition at pin T2EX when EXEN2 = 1.

T2_RC2H

Timer 2 Reload/Capture Register, Low Byte(C3_H)

Reset Value: 00_H

RMAP: 0, PAGE: X



Field	Bit	Type	Description
RC2	[7:0]	rwh	Reload/Capture Value [15:8] If CP/RL2 = 0, these contents are loaded into the timer register upon an overflow condition. If CP/RL2 = 1, this register is loaded with the current timer count upon a negative/positive transition at pin T2EX when EXEN2 = 1.

13.11.9.4 Timer 2 Count Register

Register T2L and T2H hold the current 16-bit value of the Timer 2 counter.

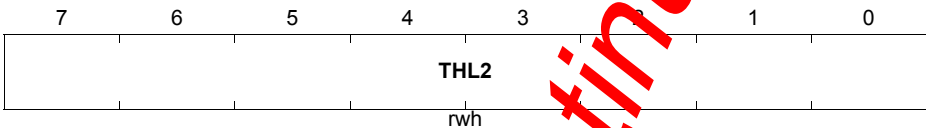
T2_T2L

Timer 2, Low Byte

(C4_H)

Reset Value: 00_H

RMAP: 0, PAGE: X



Field	Bit	Type	Description
THL2	[7:0]	rwh	Timer 2 Value [7:0] These bits indicate the current 16-bit timer value.

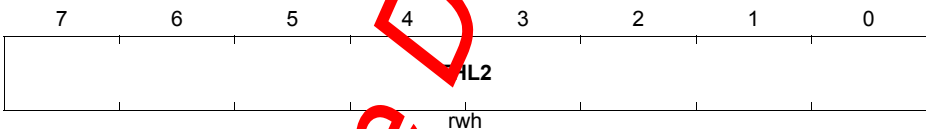
T2_T2H

Timer 2, High Byte

(C5_H)

Reset Value: 00_H

RMAP: 0, PAGE: X



Field	Bit	Type	Description
THL2	[7:0]	rwh	Timer 2 Value [15:8] These bits indicate the current 16-bit timer value.

Feature Discontinued

13.12 T2CCU

The block diagram of the Timer 2 Capture/Compare Unit module is shown in [Figure 13-47](#). It consists of the standard Timer 2 and a Capture/compare unit (CCU). The Capture/Compare Timer (CCT) is part of the CCU. Control is available in the CCU to select individually for each of its 16-bit capture/compare channel, either the Timer 2 or its own Capture/Compare Timer (CCT) as the time base. Both timers have a resolution of 16 bits. The clock frequency of T2CCU, f_{T2CCU} , is set at PCLK frequency.

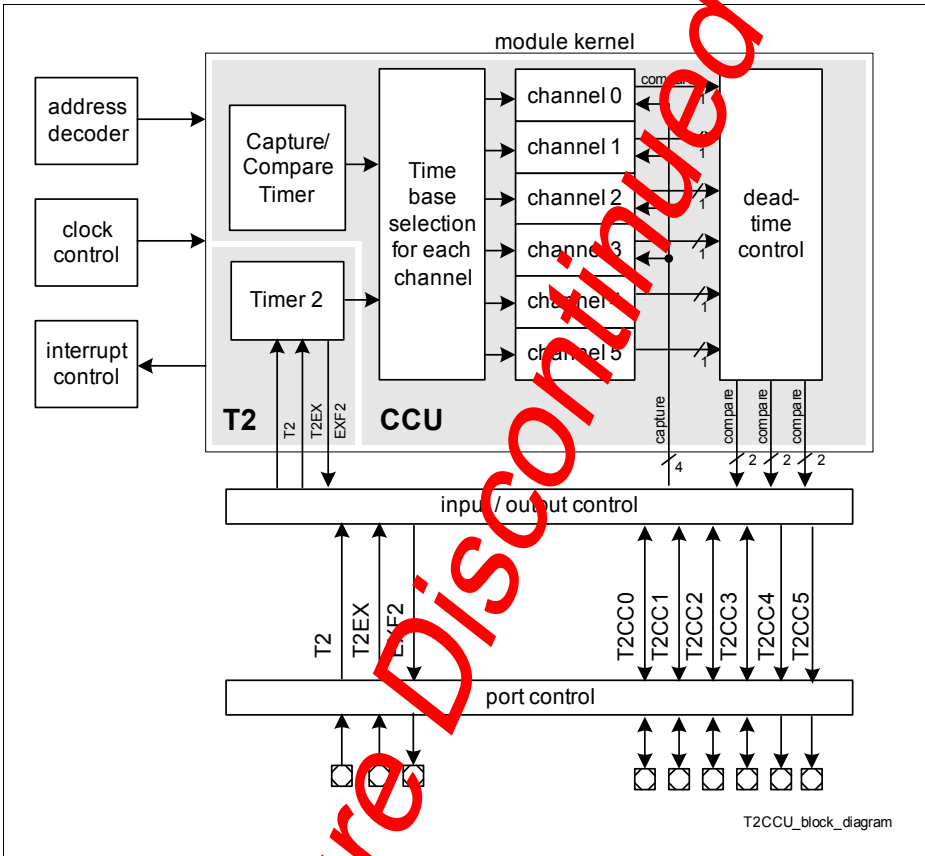
The T2CCU can be used for various digital signal generation and event capturing like pulse generation, pulse width modulation, pulse width measuring etc. Target applications include various automotive control as well as industrial (frequency generation, digital-to-analog conversion, process control etc.).

13.12.1 Features

Timer 2 Capture/Compare Unit has the following features:

- Option to select individually for each channel, either Timer 2 or Capture/Compare Timer as time base
- Extremely flexible Capture/Compare Timer count rate by cascading with Timer 2
- Capture/Compare Timer may be 'reset' immediately by triggering overflow event
- 16-bit resolution
- Six compare channels in total
- Four capture channels multiplexed with the compare channels, in total
- Shadow register for each compare register
 - Transfer via software control or on timer overflow.
- Compare Mode 0: Compare output signal changes from the inactive level to active level on compare match. Returns to inactive level on timer overflow.
 - Active level can be defined by register bit for channel groups A and B.
 - Support of 0% to 100% duty cycle in compare mode 0.
- Compare Mode 1: Full control of the software on the compare output signal level, for the next compare match.
- Concurrent Compare Mode with channel 0
- Capture Mode 0: Capture on any external event (rising/falling/both edge) at the 4 pins T2CC0 to T2CC3.
- Capture Mode 1: Capture upon writing to the low byte of the corresponding channel capture register.
- Capture mode 0 or 1 can be established independently on the 4 capture channels.

Figure 13-47 Timer 2 Capture/Compare Unit Block Diagram



13.12.2 Requirements of Timer 2

By default, all T2CCU channels select the 16-bit Timer 2 as the time base. Timer 2 can be started in timer function (C/T2), by setting bit TR2 or by external start via T2EX. Prior to this, the Timer 2 reload value on overflow should be initialized in the registers RC2H/L. The Up/Down Counter function of Timer 2 must be disabled (DCEN = 0), so that Timer 2 functions as a pure up timer only. T2PRE should be set to select the timer operating frequency, this setting depends on the application requirement.

Timer 2 supports operational frequencies of f_{PCLK} , $f_{PCLK}/2$, $f_{PCLK}/4$, $f_{PCLK}/8$, $f_{PCLK}/16$, $f_{PCLK}/32$, $f_{PCLK}/64$, or $f_{PCLK}/128$. Thus, the 16-bit timer is incremented in the fastest case, twice in every machine cycle and in the slowest case, once in every sixty-four machine cycles.

The timer overflow flag TF2 is set when there is an overflow of the timer contents, and can generate an interrupt if enabled.

Timer 2 may be stopped any time by clearing bit TR2.

Note: Only T2CCU related functions of Timer 2 are discussed in this section. For more details on Timer 2, refer to the corresponding chapter/section.

13.12.3 Capture/Compare Timer of T2CCU

The Capture/Compare Timer (CCT) is part of the T2CCU and is a dedicated time base for the T2CCU capture/compare operations.

CCT is basically a free-running counter that is started by setting bit T2CCU_CCTCON.CCTST. When this bit is set, CCT will start counting from the value in the CCTH/L timer register. The timer counts up to $FFFF_H$ and on overflow, the timer will be reloaded with the value initialized in the registers CCTRELH/L, and count up from this value. CCT can be clocked at frequencies of f_{PCLK} , $f_{PCLK}/2$, $f_{PCLK}/4$, $f_{PCLK}/8$, $f_{PCLK}/16$, $f_{PCLK}/32$, $f_{PCLK}/64$, $f_{PCLK}/128$, $f_{PCLK}/256$, $f_{PCLK}/512$, $f_{PCLK}/1024$, or $f_{PCLK}/2048$. Thus, CCT is incremented in the fastest case, twice in every machine cycle and in the slowest case, once in every 1024 machine cycles.

The timer overflow flag CCTOVF is set on overflow of the timer contents, and can generate an interrupt if enabled by bit CCTOVEN.

The CCT timer registers CCTH/L may be written any time to (re-)initialize the timer value, even when it is running. When CCT is not running, write to CCTL/H updates the corresponding byte of timer. However, when CCT is running, CCTH has to be written first. On write to CCTL, the values in CCTH and CCTL will be updated to the timer which starts counting from this value.

CCT may be stopped any time by clearing bit CCTST. The timer registers will hold the last timer count.

Triggered CCT Timer Overflow

A special feature of the CCT timer is the possibility to 'reset' the timer immediately by a Triggered Timer Overflow event. By 'reset', CCT timer is reloaded and start counting from the value of the reload registers, while all actions associated with the operation mode of the timer changes on base timer overflow will be executed. On setting bit T2CCU_CCTBSEL.CCTFOW the CCT timer overflow event is triggered asynchronously to accomplish all the above. An example usage for a channel operating in compare mode 0 is illustrated in [Figure 13-48](#). For more details on compare mode 0, refer to [Section 13.12.6](#).

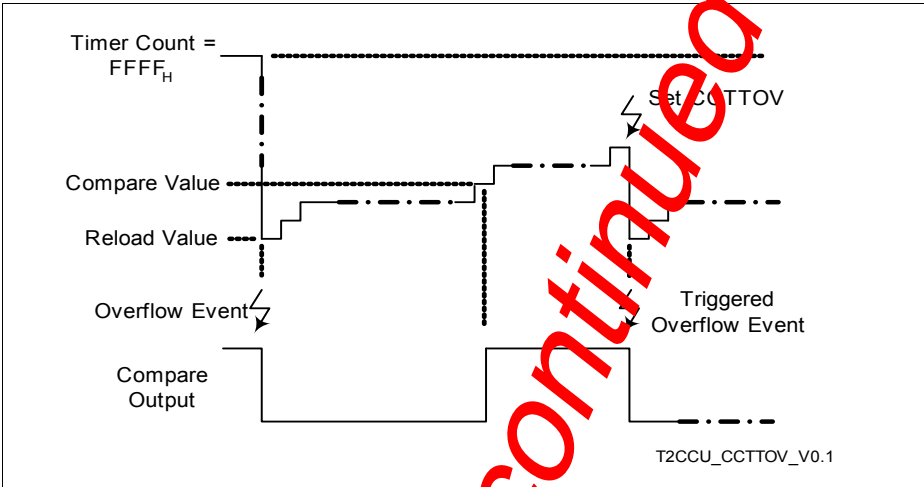


Figure 13-48 Example of Channel in Compare Mode 0 on 'Triggered Timer Overflow' Event

13.12.3.1 Synchronized Timer Start

As described in above sections, it is possible to individually start/stop the timers: Timer 2 by TR2 bit; Capture/Compare Timer by CCTST bit. If the user wants to synchronize the timer starts, then the procedure required is:

1. Ensure CCT timer is not already running,
2. Clear bit TR2 (stop Timer 2) and ensure timers are properly initialized,
3. Set bit TIMSYN to enable synchronized timer starts,
4. Set bit CCTST.

Step 3 and 4 can be combined in a single write operation to the sfr T2CCU_CCTCON.

Synchronized timer start is defined for the case where CCT timer is not already running. When T2CCU_CCTCON bit TIMSYN = 1 and CCTST = 0, setting CCTST sets also the Timer 2 T2CON.TR2 bit. Therefore, both timers are started simultaneously. In the case Timer 2 is already running when CCTST is set, Timer 2 is not affected and continues normal counting operation.

To stop the timers, the respective timer bits can be cleared independently.

13.12.3.2 Cascading Timers For Flexible Count Rate

Especially for applications requiring the base timer to run at flexible count rate, it is allowed to cascade the Timer 2 with the CCT timer. If the bit T2CCU_CCTBSEL.CASC

is set, CCT timer will count at the overflow rate of Timer 2. Refer to [Figure 13-49](#) where Timer 2 is configured as a timer.

An example usage could modify the reload value of Timer 2 for the next overflow, whenever required by the application. If it fits the application, Timer 2 may still be used by the channels as the time base for capture or compare operations.

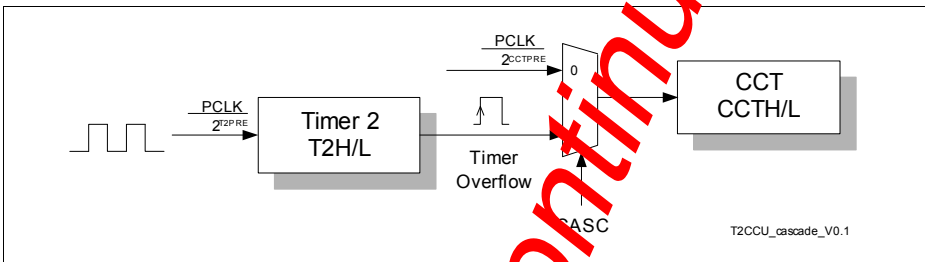


Figure 13-49 Cascading Timer 2 to CCT Timer

Note: Timer 2 may be configured in timer mode or as an external event counter via pin T2.

13.12.3.3 Dead-time with CCT Timer

The generation of (complementary) signals with dead-time for the highside and the lowside switches of one power inverter phase is based on twin grouping of the consecutive compare channels – one from group A and the other from group B. The three compare channel pairs therefore comprise of channel 0 with 1, channel 2 with 3, and channel 4 with 5. All channels must be enabled to run in compare mode 0 with CCT timer as the time base. The compare value must be set the same for the channels of a channel pair. It is also required that all the highside and lowside switches have the same active polarity. If the highside switch should be active while the CCT timer value is above the compare value, then the lowside switch should be active while the CCT timer value is below the compare value. The polarity on compare match is controlled via the bits T2CCU_COCON.POLA and PCPLB for compare group A and B respectively – settings of which must be inverted.

Feature Discontinued

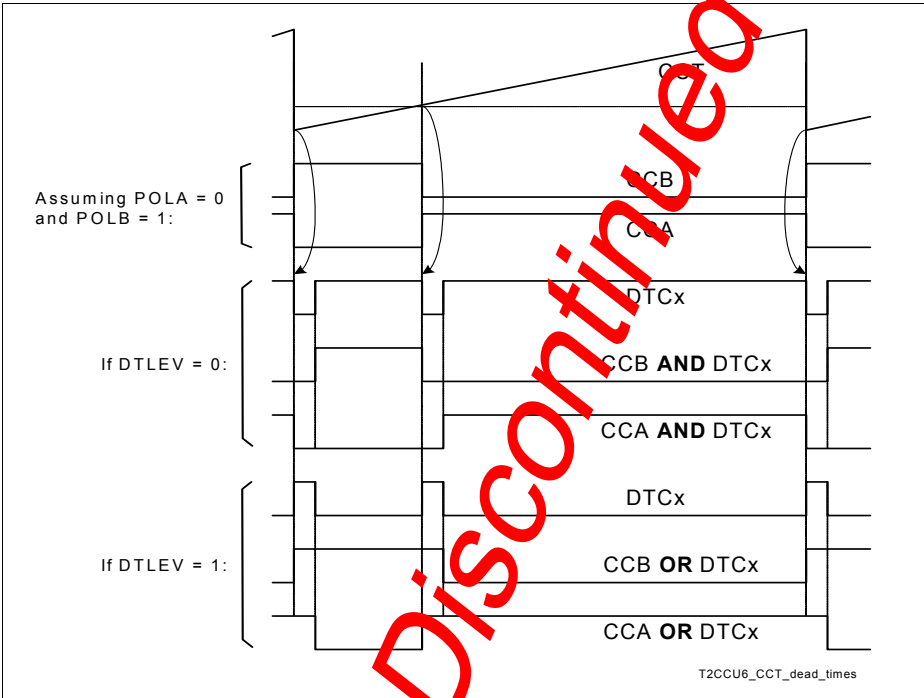


Figure 13-50 PWM-signals with Dead-time Generation (Illustrated for One Channel Pair)

In most cases, the switching behavior of the connected power switches is not symmetrical concerning the times needed to switch on and to switch off. A general problem arises if the time taken to switch on is less than the time to switch off the power device. This leads to a short-circuit in the inverter bridge leg, which may damage the complete system. In order to solve this problem by hardware, T2CCU contains a programmable dead-time counter, which delays the passive to active edge of the switching signals (the active to passive edge is not delayed); see [Figure 13-50](#).

The dead-time generation logic (see [Figure 13-51](#)) is built in a similar way for all three channel pairs. Each instance of the CCT timer overflow or compare match event (of channel from group A) triggers the corresponding dead-time counter (8-bit down counter, clocked at CCT_count_freq) and toggles DTCx to the active level. The trigger pulse (DTCx_rl) leads to a reload of the dead-time counter with the value programmed in bit field T2CCU_CCTDTCL.DTM. This reload can only take place if the dead-time feature is enabled by bit T2CCU_CCTDTCH.DTEx and while the counter is zero.

While counting down (zero is not yet reached), the control signal DTCx stays at active level. On reaching the counter value zero, the dead-time counter stops counting and the signal DTCx toggles again to the inactive level. The DTCx signal is logically manipulated with the CCT modulation signals CCA and CCB, leading to a delay of the passive to active edge of the resulting compare output signal, which is shown in **Figure 13-50**. The active level of the DTCx and the manipulation of the modulation signals is defined and handled differently depending on setting of T2CCU_CCTDTCx.DTLEV (defines the passive level of the switches).

Note: Signal DTCx toggles only if dead-time is enabled by T2CCU_CCTDTCH.DTEX. While DTEX is enabled, DTCx toggles only when dead-time counter is reloaded or when the counter value changed to zero (counter was actively down-counting). When DTEX is set from the clear state, DTCx is held at inactive level according to setting of DTLEV until the next dead-time event.

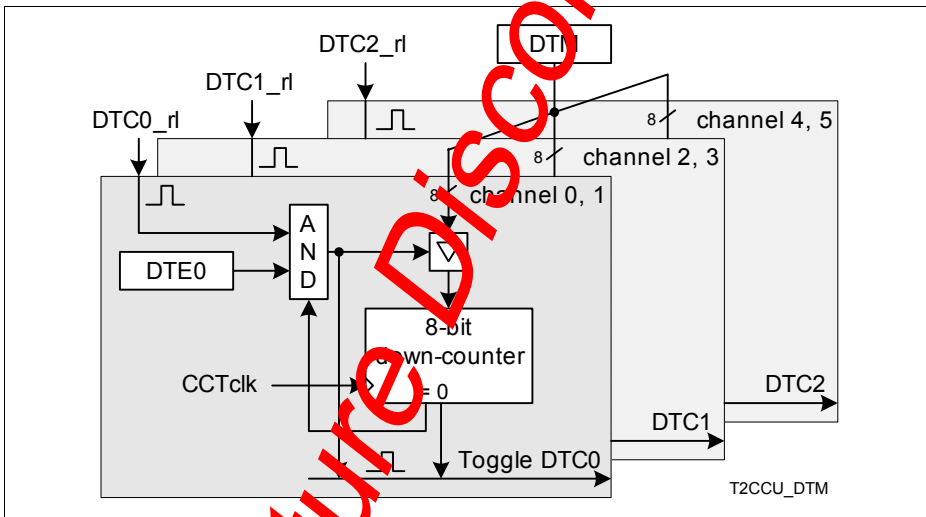


Figure 13-51 Dead-time Counter

Each of the three channel pairs works independently with its own dead-time counter, the trigger and enable signals. The value of bit field DTM is valid for all three channel pairs.

13.12.4 T2CCU Capture/Compare Channels

The T2CCU allows the following operating modes:

- Compare: Up to 4 PWM output signals with 65536 steps at maximum
- Capture: Up to 4 high speed capture inputs

The capture/compare resolution for each channel is per the timer base operating frequency.

Figure 13-52 shows a block diagram of Timer 2 with the T2CCU Capture/Compare Timer (CCT) and capture/compare function for a channel. The I/O pins are generally assigned as alternate function to general purpose I/O port. As shown in **Table 3**, the I/Os T2CC0, 1, 2, 3, 4 always share the same pin as the external interrupts EXT_INT3, 4, 5, 6 respectively. The corresponding external interrupt request flags are set for the capture/compare function. The respective timer overflow flags are also set on timer overflow.

Table 3 I/Os of T2CCU

Pins	Function
T2CC0 / EXINT3	Compare output/Capture input for channel 0
T2CC1 / EXINT4	Compare output/Capture input for channel 1
T2CC2 / EXINT5	Compare output/Capture input for channel 2
T2CC3 / EXINT6	Compare output/Capture input for channel 3
T2CC4	Compare output for channel 4
T2CC5	Compare output for channel 5

Note: The pins T2CCU4, 5 are not available on devices which support up to 4 compare channels.

Feature Discontinued

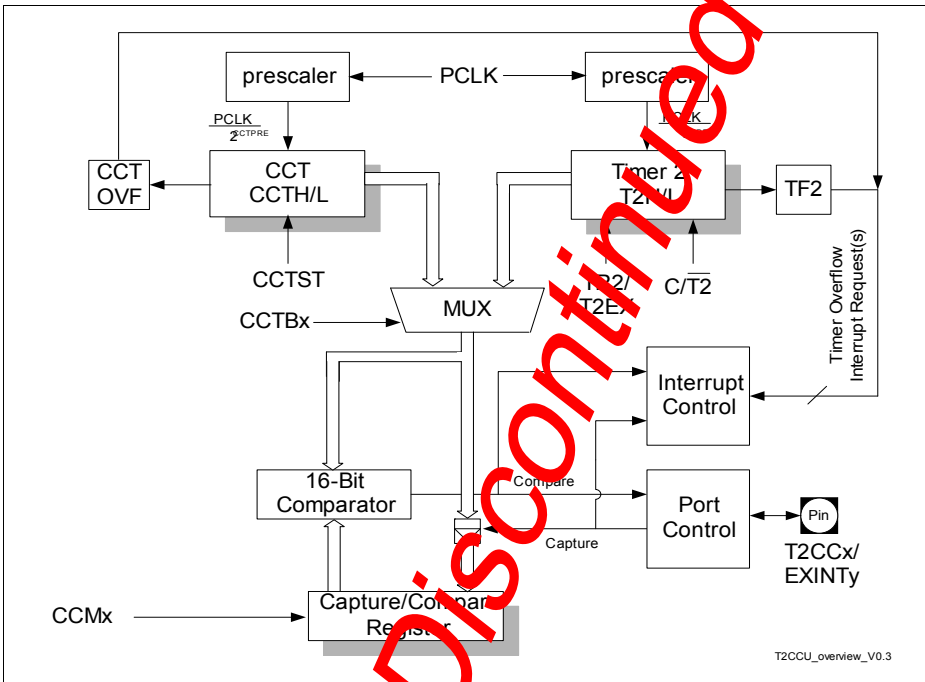


Figure 13-52 Timer 2 with T2CCU CC and Capture/Compare Function for Channel x

13.12.5 Capture/Compare Operation

There are altogether six compare channels and altogether four capture channels, which can be individually enabled. The capture channels can be independently selected for capture mode 0 or 1. A general selection of the capture/compare function for each channel is done in register T2CCU_CCEN.

In case of compare mode selected, the register T2CCU_COCON allows to configure the compare mode for all compare channels. The compare channels are further divided into two groups - A and B. If the channels are configured for compare mode respectively, group A consists of the channels CC0, CC2 and CC4 while group B consists of the channels CC1, CC3 and CC5. In this way, the level of the output signal on compare match till timer overflow (active level) can be configured separately for the channels of the two groups.

In case of capture mode selected, the external event for the capture can be configured by the external interrupt SFR bits. The active trigger could be selected individually for each channel for rising, falling or both edge triggered.

Note: For defined behavior, it is required of the user to initialize properly all the T2CCU related SFRs before enabling any T2CCU capture or compare mode.

13.12.6 Compare Function

The T2CCU compare output is only actively driven on enabling the compare function. When the compare function is enabled by writing to a channel's CCMx bits, the respective compare output is latched (as initial value) based on the settings to the channel SFR bits.

In compare mode, the 16-bit value stored in a compare register is compared with the contents of the timer. When the timer count matches the compare register value, the corresponding channel's output signal is toggled at the pin and an interrupt is requested if enabled.

The contents of a compare register can be regarded as a "time stamp" at which the corresponding output reacts in a predefined way (with either a positive or negative transition). Variation of this "time stamp" changes the waveform of a rectangular output signal at the pin. As a variation of the duty cycle of a periodic signal, this may be used for pulse width modulation as well as for a continually controlled generation of any kind of square waveforms. Two compare modes plus one concurrent compare mode are available to cover a wide range of possible applications.

The active compare mode is selected by bit T2CCU_COCON.COMOD, and applies to all enabled compare channels. In all compare modes, the new output value arrives at the port pin within the same PCLK cycle in which the internal compare signal is activated.

The compare match signal stays high for as long as the comparator inputs are equal (at least one PCLK). The rising edge of the compare match signal is the active edge for all compare match operations.

Shadow Registers for Compare Registers

The compare registers of the T2CCU cannot be accessed directly by user code. Update of channel compare value is by shadow transfer from the corresponding SFR T2CCU_CCx. The user may choose to select one out of two events to start a shadow transfer to the internal compare registers.

1. By setting the SFR bit T2CCU_COSHDW.TXOV = 1, shadow transfer is started for the compare channels on the overflow of their corresponding timer base. In this case, the update of the compare registers is controlled by hardware and is therefore independent from any service delay.
2. By setting the SFR bit T2CCU_COSHDW.ENSHDW = 1, shadow transfer is started for each channel when the corresponding SFR bit COOUTx is set under user control. COOUTx will be cleared automatically by hardware at the next PCLK, after being set.

In both cases, the rising edge of the event pulse signal is the active edge for the shadow transfer of the compare register(s). The advantage of the shadow transfer mechanism is

that it allows glitch-free pulse width modulation, which can occur due to the two instructions needed to update the 16-bit compare registers.

Note that it is not allowed to set both bits TXOV and ENSHDW at the same time. If bit TXOV is set to enable shadow transfer on timer overflow, bit ENSHDW will be cleared by internal hardware to disable shadow transfer by software.

Figure 13-53 illustrates the shadow transfer mechanism.

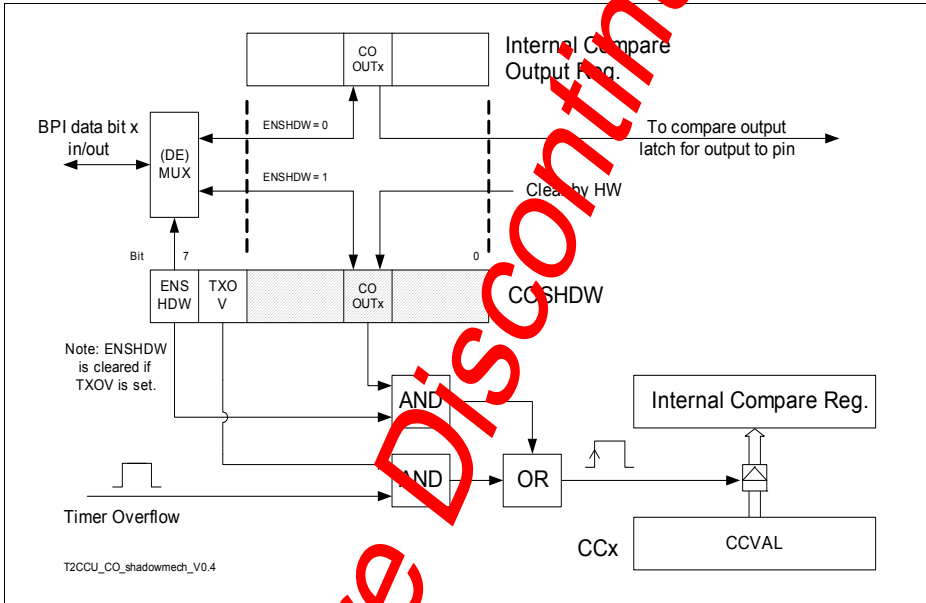


Figure 13-53 Compare Value Shadow Transfer Mechanism and The Dual Function of Bits COOUTx

Mapped Compare Output Bits

To enable dual function on the COOUTx bits of SFR T2CCU_COSHDW, bit ENSHDW is used for the control of access to these bits. When ENSHDW = 1, the SFR bits COOUTx can be directly accessed for control of shadow transfer of compare value. When ENSHDW = 0, writing or reading to COOUTx bits actually access an internal compare output register instead of the COSHDW bits. The corresponding bits of this internal compare output register control the output polarity of the compare channels in compare mode and concurrent compare mode.

Figure 13-53 illustrates the dual function of the bits COOUTx.

13.12.6.1 Compare Mode 0

When compare mode 0 is selected, on setting CCMx to enable compare function, the compare output is latched,

- To the active level if the base timer content are equal or bigger than the compare register content, or,
- To the inactive level if the base timer content is less than the compare register content.

The active level is defined for each channel by its compare group A or B, via the SFR bit T2CCU_COCON.POLA or POLB.

In compare mode 0, when the timer and compare register contents match, the output signal changes from the inactive level to active level. The compare output signal goes back to inactive level on timer overflow.

Compare mode 0 is ideal for generating pulse width modulated output signals, which in turn can be used for digital-to-analog conversion via a filter network or by the controlled device (e.g. the inductance of a DC or AC motor). Mode 0 may also be used for providing output clocks with initially defined period and duty cycle. This usage needs the least CPU time. Once set up, the output goes on oscillating without any CPU intervention.

Figure 13-54 illustrates the function of compare mode 0. In this function, the compare output signal is directly controlled by the setting of SFR bit T2CCU_COCON.POLA/B and the events: timer overflow and compare match.

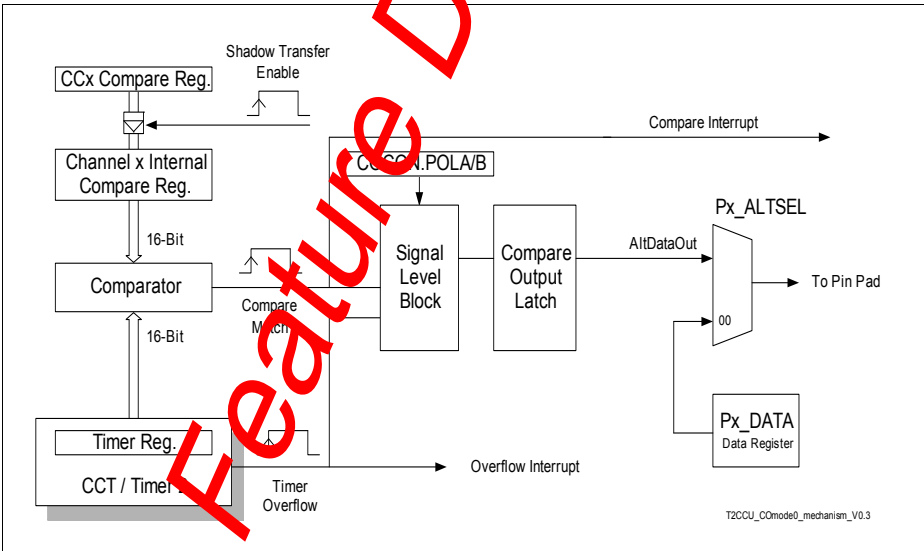


Figure 13-54 Functional Diagram of Compare Mode 0

Figure 13-55 illustrates the channel output when POLA/B is set to 0. Interrupts may be generated on timer overflow or compare match, and update the next compare value etc., in the interrupt service routine.

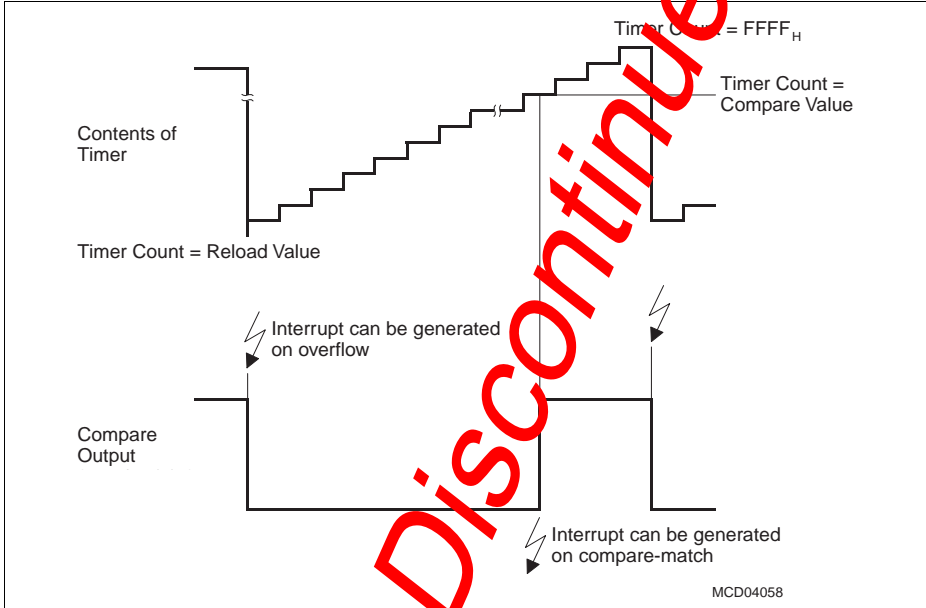


Figure 13-55 Channel Output in Compare Mode 0 (POLA/B = 0)

PWM Signal from 0% to 100% Duty Cycle in Compare Mode 0

The active level of a PWM signal refers to the compare output signal level from the time of compare match till the next timer overflow.

To support a 100% duty cycle for a compare channel with T2CCU_COCON.POLA/B = 0 (active level = high), the 16-bit channel x compare value must be set equal to the timer overflow reload value. In this case, the internal hardware will not cause the compare output signal to the inactive level on timer overflow, and the output stays at the active level until the next timer overflow when the compare value is no more equal to the timer reload value.

To support a 0% duty cycle for a compare channel with T2CCU_COCON.POLA/B = 0, the channel x compare value must be smaller than the timer overflow reload value. In this case, the timer reload value must not be zero. The compare match event will not happen and the compare output stays at the inactive level (low).

The above mechanism applies for compare channel with T2CCU_COCON.POLA/B = 1 correspondingly.

13.12.6.2 Compare Mode 1

When compare mode 1 is selected, on setting CCMx to enable compare function, the compare output is latched with the value initialized to bit COOUTx of the internal compare output register.

In compare mode 1, the software adaptively determines the transition of the output signal. It is commonly used when output signals are not related to a constant signal period (as in a standard PWM Generation) but must be controlled very precisely with high resolution and without jitter. In compare mode 1, the compare output signal level is under the full control of the software, for the next compare match.

If compare mode 1 is enabled and the software writes to COOUTx bit(s) of the internal compare output register (in this case, bit T2CCU_C0SHDW.ENSHDW must be cleared), the new value will not appear at the output pin until the next compare match occurs. Thus, one can choose independently for each compare channel, whether the output signal is to toggle (1-to-0 or 0-to-1) or to remain at the same level, when the channel timer count matches the contents of the channel internal compare register.

Figure 13-56 illustrates the function of compare mode 1. In this function, the bit COOUTx of the internal compare output register will only be transferred to the compare output latch as alternate data out, on the next compare match event.

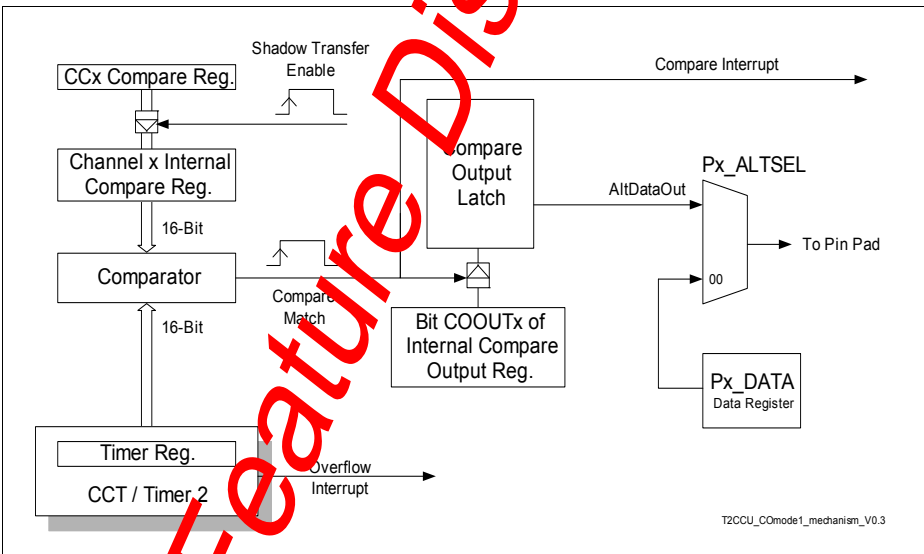


Figure 13-56 Functional Diagram of Compare Mode 1

13.12.6.3 Concurrent Compare Mode

When concurrent compare mode is selected, on setting CCMx to enable compare function, the compare outputs are latched with the corresponding values initialized to the COOUTx bits of the internal compare output register.

Concurrent Compare is similar to compare mode 1, with the additional capability to manipulate the output at more than one compare channel. Like in compare mode 1, the software updates the COOUTx bits of the internal compare output register (bit T2CCU_COSHDW.ENSHPDW must be cleared), so that up to six compare channel outputs are updated concurrently when channel 0 compare match occurs. When this happens, all enabled compare channel outputs will be updated with the corresponding value in the internal compare output register.

Concurrent compare is an ideal and effective mode for applications where more than one synchronous output signal is to be generated. Such applications could include for e.g., a multiple-phase stepper motor control, or control of ignition coils of a car engine. These applications generally require predefined bit-patterns to be put to an output port at precisely predefined moment.

Figure 13-57 illustrates the function of concurrent compare mode. Compare channel 0 must be enabled and initialized for proper operation.

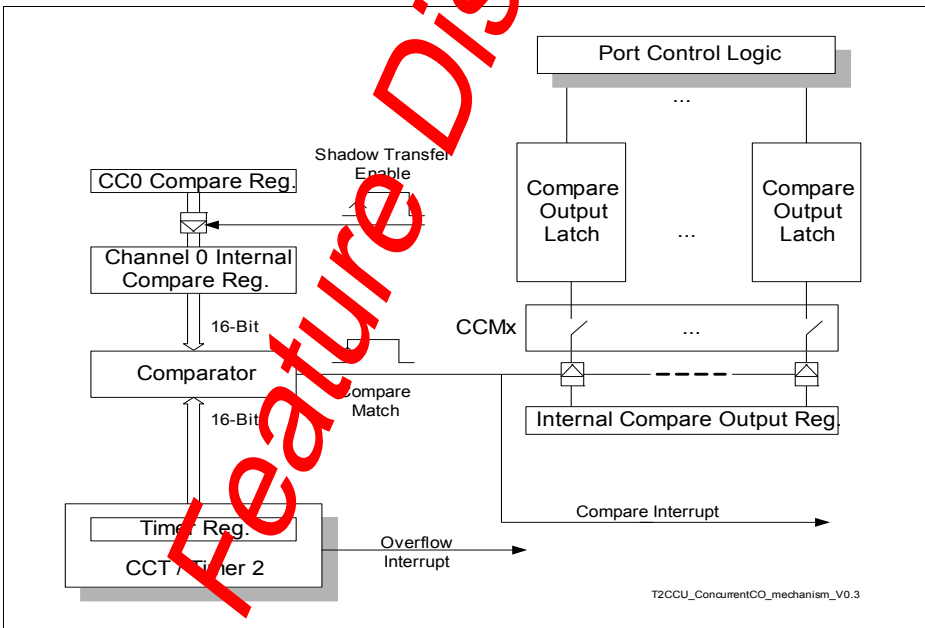


Figure 13-57 Functional Diagram of Concurrent Compare Mode

Figure 13-58 illustrates two examples of how four rectangular waveforms can be generated at the port using a pattern table. The patterns are written to the internal compare output register (with T2CCU_COSHDW.ENSHDW = 0) before the corresponding timer count is reached. The (future) timer count at which the pattern shall be available at the port must be loaded to the compare shadow registers T2CCU_CC0H and T2CCU_CC0L, and transferred to the internal compare registers. To manage the waveform generation, one example uses a time schedule table with list of compare values, the other example makes use of a fixed compare value (such as FFF0_H) with shorter timer period. In these ways, each channel can be controlled at predefined time(s) by the user on whether to toggle or not.

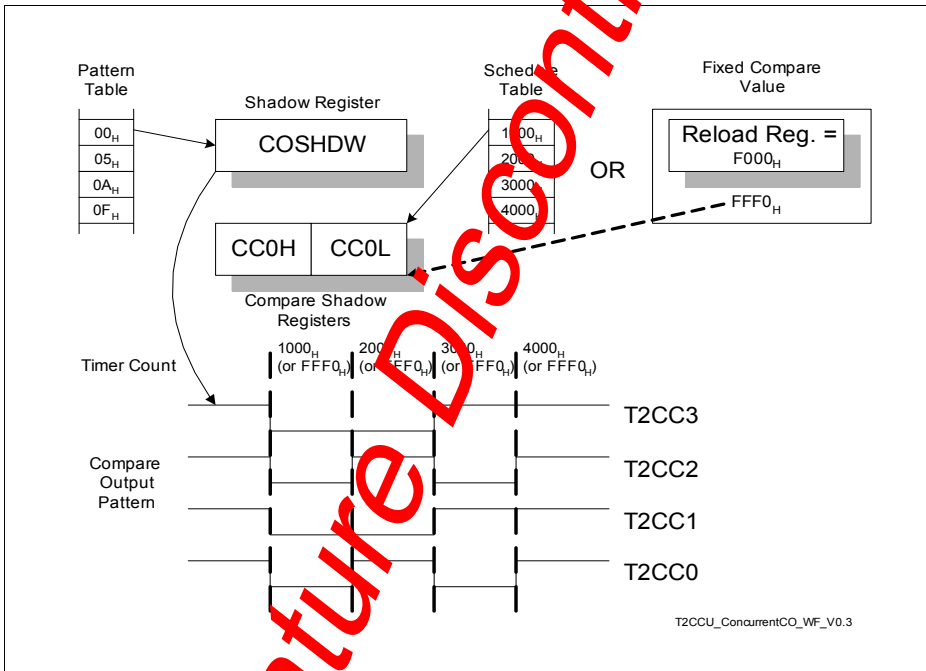


Figure 13-58 Two Examples of Waveform Generation in Concurrent Compare Mode (by Schedule Table of Compare Values or Fixed Compare Value)

13.12.6.4 Using Interrupts in Combination with the Compare Function

Reserved External Interrupts

Each of the compare channels T2CCU_CC0-3 share the same port pin with an external interrupt as alternate functions at port pins. When compare mode is enabled and the corresponding port pin is set to output mode for use as compare output, the input driver of the port pin is disabled. Therefore the corresponding external interrupt inputs cannot be used for any purpose.

However, the interrupt function of the external interrupts can still be enabled for the compare match interrupts. In this case, the corresponding external interrupt lines are connected directly to the internal compare match signal.

Compare Match Interrupt

The compare match interrupt (including that for channels 4 and 5) is enabled together with the capture interrupt by the IEN1.EXM bit for all channels. A channel compare match occurs when timer count matches the channel register contents. Besides manipulating the corresponding channel compare output, the corresponding compare match flag IRCON0.EXINTx or T2CCU_CCW00.CMxF is set for the interrupt request. For this, the interrupt edge trigger for compare match must be set to rising edge in the corresponding EXICON register. The compare match interrupt service routine can be used to set up the parameters for the next channel compare event, such as change the contents of the compare registers or to determine the level of the compare port outputs for the next “compare match”.

The main advantage in using compare match interrupts is to avoid unintentional overwriting of compare/compare output registers before a match has been reached. This could happen when the CPU writes to the compare registers without knowing about the actual timer count.

Note: As an alternative, the timer overflow interrupt may be used in place of the compare match interrupt.

13.12.7 Capture Function

This mode allows the measurement of external signal pulse widths through the Capture/Compare registers CC0-3. Each of the 16-bit registers can be used to latch the current 16-bit count of the timer base of the corresponding channel. Two capture modes are provided for this function.

In mode 0, any external event (rising/falling/both edge) at pins T2CC0 to T2CC3 latches the channel timer contents to the corresponding channel capture register. The capture edge is defined by the corresponding external interrupt set up (each capture input share the same pin with an external interrupt). The external input(s) is sampled in every CCLK

cycle. When a sampled input shows a low (high) level in one CCLK cycle and a high (low) in the next CCLK cycle, a transition is recognized. The corresponding external interrupt request flag(s) (EXINT3 to EXINT6) is set. If the interrupt(s) is enabled, the CPU will vector to the appropriate interrupt service routine. The timer contents are latched to the corresponding capture register in the PCLK cycle following the one in which the transition was identified.

In mode 1, a capture will occur upon writing to the low byte of the corresponding channel capture register. The software is able to read the contents of the channel timer “on-the-fly”. The value written to the capture register is irrelevant for this function. The timer contents will be latched into the channel capture register following the write instruction. In this mode no interrupt request will be generated.

Figure 13-59 shows the capture operation of the Capture/Compare channels.

The two capture modes can be established individually for each capture channel by bits in SFR T2CCU_CCEN (Capture/Compare enable register). That means, in contrast to the compare modes, it is possible to simultaneously select mode 0 for one capture channel and mode 1 for another channel.

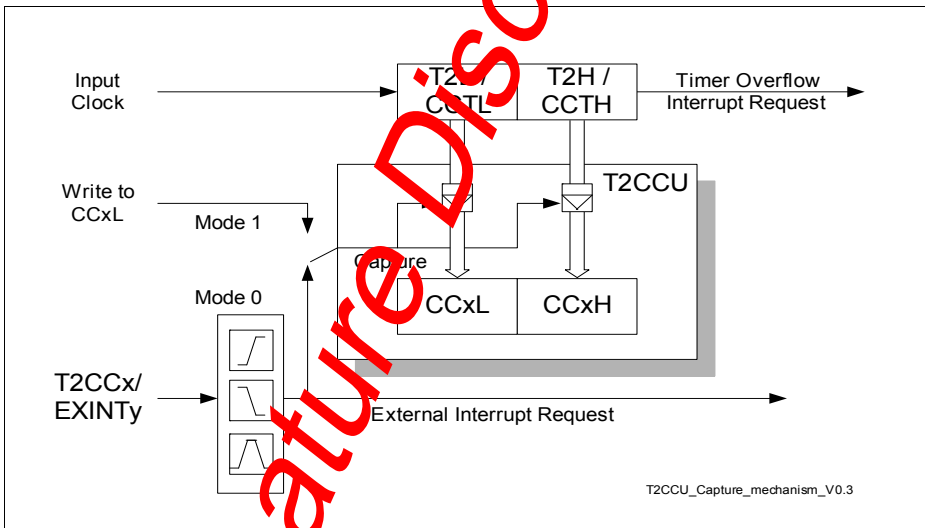


Figure 13-59 Functional Diagram of Capture Mode

13.12.8 T2CCU Registers

This section describes all T2CCU special function registers. The Timer 2 and interrupt related SFRs are also listed, as shown in [Table 4](#).

Note: All T2CCU register names shall be referenced fully with the module name prefix "T2CCU_".

Table 4 Registers Overview

Name	Description
T2CCU Special Function Registers	
CCTBSEL	Capture/Compare Timer Base Select Register
CCTRELL	Capture/Compare Timer Reload Register Low
CCTRELH	Capture/Compare Timer Reload Register High
CCTL	Capture/Compare Timer Register Low
CCTH	Capture/Compare Timer Register High
CCTCON	Capture/Compare Timer Control Register
CCTDTCL	Capture/Compare Timer Dead-Time Control Register Low
CCTDTCH	Capture/Compare Timer Dead-Time Control Register High
CCEN	Capture/Compare Enable Register
COSHDW	Compare Shadow Register
COCON	Compare Control Register
CC0L	Capture/Compare Register 0 Low
CC0H	Capture/Compare Register 0 High
CC1L	Capture/Compare Register 1 Low
CC1H	Capture/Compare Register 1 High
CC2L	Capture/Compare Register 2 Low
CC2H	Capture/Compare Register 2 High
CC3L	Capture/Compare Register 3 Low
CC3H	Capture/Compare Register 3 High
CC4L	Compare Register 4 Low
CC4H	Compare Register 4 High

Table 4 Registers Overview (cont'd)

Name	Description
CC5L	Compare Register 5 Low
CC5H	Compare Register 5 High
Related Timer 2 Special Function Registers	
T2_T2CON	Timer 2 Control Register
T2_T2CON1	Timer 2 Control Register 1
T2_T2MOD	Timer 2 Mode Register
T2_RC2L	Timer 2 Reload Register, Low Byte
T2_RC2H	Timer 2 Reload Register, High Byte
T2_T2L	Timer 2, Low Byte
T2_T2H	Timer 2, High Byte
Related Interrupt Special Function Registers	
IEN0	Interrupt Enable Register 0
IEN1	Interrupt Enable Register 1
EXICON0	External Interrupt Control Register 0
EXICON1	External Interrupt Control Register 1
IRCON0	Interrupt Request Register 0

13.12.8.1 Register Mapping

The BPI of the Timer 2 with T2CCU supports local address extension mechanism. The SFRs of the Timer 2 are accessed from Page 0, while SFRs of the T2CCU are accessed from Page 1 to Page 4.

The addresses (non-mapped) of the SFRs are listed in [Table 5](#).

Table 5 SFR Address List for Pages 0-4

Address	Page 0	Page 1	Page 2	Page 3	Page 4
C0 _H	T2CON	CCEN	COSHDW	COCON	
C1 _H	T2MOD	CCTBSEL	CC0L	CC3L	
C2 _H	RC2L	CCTRELL	CC0H	CC3H	CCTDTCL
C3 _H	RC2H	CCTRELH	CC1L	CC4L	CCTDTCH
C4 _H	T2L	CCTL	CC1H	CC4H	
C5 _H	T2H	CCTH	CC2L	CC5L	
C6 _H	T2CON1	CCTCON	CC2H	CC5H	

Note: For Timer 2 with T2CCU, the Timer 2 SFRs must be accessed via the paging mechanism at Page 0.

SFR T2_PAGE, at address C7H, contains the page value and the page control information.

T2CCU_PAGE
Page Register for T2CCU
(C7_H)
Reset Value: 00_H
RMAP: 0, PAGE: X

	7	6	5	4	3	2	1	0
	OP		STNR		0	PAGE		
	w		w		r	rw		

Field	Bits	Type	Description
PAGE	[2:0]	rw	<p>Page Bits</p> <p>When written, the value indicates the new page address.</p> <p>When read, the value indicates the currently active page = addr [y:x+1]</p>
STNR	[5:4]	w	<p>Storage Number</p> <p>This number indicates which storage bit field is the target of the operation defined by bit OP.</p> <p>If OP = 10_B, the contents of PAGE are saved in STx before being overwritten with the new value.</p> <p>If OP = 11_B, the contents of PAGE are overwritten by the contents of STx. The value written to the bit positions of PAGE is ignored.</p> <p>00_B ST0 is selected. 01_B ST1 is selected. 10_B ST2 is selected. 11_B ST3 is selected.</p>

Feature Discontinued

Field	Bits	Type	Description
OP	[7:6]	w	Operation 0x _B Manual page mode. The value of STNR is ignored and PAGE is directly written. 10 _B New page programming with automatic page saving. The value written to the bit positions of PAGE is stored. In parallel, the former contents of PAGE are saved in the storage bit field STx indicated by STNR. 11 _B Automatic restore page action. The value written to the bit positions PAGE is ignored and instead, PAGE is overwritten by the contents of the storage bit field STx indicated by STNR.
0	3	r	Reserved Returns 0 if read, should be written with 0.

13.12.8.2 T2CCU Registers Description

The following describes the Special Function Registers of the T2CCU kernel.

T2CCU_CCTBSEL

T2CCU Capture/Compare Time Base Select Register(C1_H)

Reset Value: 00_H

RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
CASC	CCTTOV	CCTB5	CCTB4	CCTB3	CCTB2	CCTB1	CCTB0
rw	rwh	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CCTBx (x = 0-5)	x	rw	Channel x Time Base Select 0 _B Selects Timer 2 1 _B Selects Capture/Compare Timer (CCT) <i>Note: In case the CCT is not implemented, the signals corresponding to CCTBx must be connected to GND.</i>

Field	Bits	Typ	Description
CCTTOV	6	rwh	Trigger CCT Timer Overflow Event 0 _B No action 1 _B CCT overflow event is triggered immediately On set, this bit is held for one PCLK, then cleared by hardware. Reading this bit always returns 0.
CASC	7	rw	Cascade Timers 0 _B Timer 2 and CCT timer operate independently 1 _B Timer 2 is cascaded with CCT timer

T2CCU_CCTRELL
T2CCU Capture/Compare Timer Reload Register Low (C2_H) **Reset Value: 00_H**
RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
CCTREL[7:0]							
rw							

Field	Bits	Typ	Description
CCTREL[7:0]	[7:0]	rw	Capture/Compare Timer Low Byte Reload Value The contents of the registers are loaded into the CCT timer low byte register upon an overflow condition.

T2CCU_CCTRELH
T2CCU Capture/Compare Timer Reload Register High (C3_H) **Reset Value: 00_H**
RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
CCTREL[15:8]							
rw							

Field	Bits	Typ	Description
CCTREL[15:8]	[7:0]	rw	Capture/Compare Timer High Byte Reload Value The contents of the registers are loaded into the CCT timer high byte register upon an overflow condition.

T2CCU_CTL
T2CCU Capture/Compare Timer Register Low(C_{4H})
RMAP: 0, PAGE: 1
Reset Value: 00_H

7	6	5	4	3	2	1	0
CCT[7:0]							
rwh							

Field	Bits	Typ	Description
CCT[7:0]	[7:0]	rwh	Capture/Compare Timer Low Byte Value This register represents the low byte counter value of the Capture/Compare Timer. When a CCT is not running, write to CCTL updates the lower byte of timer. To (re-)initialize CCT when it is running, CCTH has to be written first. On write to CCTL, the values of CCTH and CCTL will be updated to the timer which counts from this value. <i>Note: In case of write and count update occurs at the same time, the written value takes effect.</i>

T2CCU_CCTH
T2CCU Capture/Compare Timer Register High(C_{5H})
RMAP: 0, PAGE: 1
Reset Value: 00_H

7	6	5	4	3	2	1	0
CCT[15:8]							
rwh							

Field	Bits	Typ	Description
CCT[15:8]	[7:0]	rw	<p>Capture/Compare Timer High Byte Value</p> <p>The register represents the high byte counter value of the Capture/Compare Timer.</p> <p>When CCT is not running, write to CCTH updates the upper byte of timer.</p> <p>To (re-)initialize CCT when it is running, CCTH has to be written first. Or write to CCTL, the values CCTH and CCTL will be updated to the timer which counts from this value.</p> <p><i>Note: In case of write and count update occurs at the same time, the written value takes effect.</i></p>

T2CCU_CCTCON
T2CCU Capture/Compare Timer Control Register (C6_H)
Reset Value: 00_H
RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
CCTPRE			CCTOVF		CCTOVEN	TIMSYN	CCTST
rw			rwh		rw	rw	rw

Field	Bits	Typ	Description
CCTST	0	rw	<p>Capture/Compare Timer Start/Stop Control</p> <p>0_B Stop Capture/Compare Timer</p> <p>1_B Start Capture/Compare Timer.</p> <p>If bit TIMSYN = 1 and CCTST was 0, start also Timer 2 (if it was not already running).</p> <p><i>Note: Setting CCTST while the bit was already set will not disturb the running CCT timer operation.</i></p> <p><i>Note: In case the CCT is not implemented, the corresponding start signal to Timer 2 must be disabled.</i></p>
TIMSYN		rw	<p>Enable Synchronized Timer Starts</p> <p>0_B Disable synchronized timer starts</p> <p>1_B Enable synchronized timer starts</p>

Field	Bits	Typ	Description
CCTOVEN	2	rw	Capture/Compare Timer Overflow Interrupt Enable Set to enable CCT overflow interrupt.
CCTOVF¹⁾	3	rwh	Capture/Compare Timer Overflow Flag Set by hardware on CCT overflow. This bit has to be cleared by software.
CCTPRE	[7:4]	rw	Compare/Compare Timer Prescaler Bit Selects the input clock for CCT, which is derived from the peripheral clock. 0000 _B $f_{CCT} = f_{PCLK}$ 0001 _B $f_{CCT} = f_{PCLK}/2$ 0010 _B $f_{CCT} = f_{PCLK}/4$ 0011 _B $f_{CCT} = f_{PCLK}/8$ 0100 _B $f_{CCT} = f_{PCLK}/16$ 0101 _B $f_{CCT} = f_{PCLK}/32$ 0110 _B $f_{CCT} = f_{PCLK}/64$ 0111 _B $f_{CCT} = f_{PCLK}/128$ 1000 _B $f_{CCT} = f_{PCLK}/256$ 1001 _B $f_{CCT} = f_{PCLK}/512$ 1010 _B $f_{CCT} = f_{PCLK}/1024$ 1011 _B $f_{CCT} = f_{PCLK}/2048$ 1100 _B Reserved 1101 _B Reserved 1110 _B Reserved 1111 _B Reserved Note: It is not recommended to change CCTPRE when CCTST = 1.

1) In case of software and hardware access to a flag bit at the same time, hardware will have higher priority.

T2CCU_CCTDTCL

T2CCU Capture/Compare Timer Dead-Time Control Register Low(C2_H)

Reset Value: 00_H

RMAP: 0, PAGE: 4

7	6	5	4	3	2	1	0
DTM							
rw							

Field	Bits	Typ	Description
DTM	[7:0]	rw	Dead-Time This 8-bit value defines the programmable delay between switching from the passive state to the active state of the selected outputs. The switching from the active state to the passive state is not delayed.

T2CCU_CCTDTCH
T2CCU Capture/Compare Timer Dead-Time Control Register High(C3_H)
Reset Value: 00_H
RMAP: 0, PAGE: 4

7	6	5	4	3	2	1	0
DTRES	DTR2	DTR1	DTR0	DTLEV	DTE2	DTE1	DTE0
rw	rh	rh	rh	rw	rw	rw	rw

Field	Bits	Typ	Description
DTE_x (x = 0-2)	x	rw	Dead-Time Enable for Channel Pair x 0 _B Dead-time generation is disabled. 1 _B Dead-time generation is enabled for channel pair x. The corresponding outputs switch from the passive state to the active state (according to compare status) with the delay programmed in DTM. <i>Note: This bit shall only be set after enabling the corresponding channels in compare mode 0 with CCT timer as the time base.</i>
DTLEV	3	rw	Dead-Time Level This bit defines the active level of all DTC _x during dead-time counting. 0 _B Low (switches are active high). 1 _B High (switches are active low).

Field	Bits	Typ	Description
DTRx (x = 0-2)	x+4	rh	Dead-Time Run Indication Bit 0 _B The value of the corresponding dead-time counter is 0. 1 _B The value of the corresponding dead-time counter is not 0.
DTRES	7	rwh	Reset Dead-Time Counters 0 _B No action. 1 _B The three dead-time counters are reset to 0. On set, this bit is held for one PCLK, then cleared by hardware. Reading this bit always return 0.

T2CCU_CCEN
T2CCU Capture/Compare Enable Register (CC_{EN})
Reset Value: 00_H
RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
CCM3		CCM2		CCM1		CCM0	
rw		rw		rw		rw	

Field	Bits	Typ	Description
CCM0	[1:0]	rw	Capture/Compare Enable for Channel 0 00 _B Capture/Compare disabled 01 _B Capture on active edge at pin T2CC0 10 _B Compare enabled 11 _B Capture on write operation into register CC0L
CCM1	[3:2]	rw	Capture/Compare Enable for Channel 1 00 _B Capture/Compare disabled 01 _B Capture on active edge at pin T2CC1 10 _B Compare enabled 11 _B Capture on write operation into register CC1L
CCM2	[5:4]	rw	Capture/Compare Enable for Channel 2 00 _B Capture/Compare disabled 01 _B Capture on active edge at pin T2CC2 10 _B Compare enabled 11 _B Capture on write operation into register CC2L

Field	Bits	Typ	Description
CCM3	[7:6]	rw	Capture/Compare Enable for Channel 3 00 _B Capture/Compare disabled 01 _B Capture on active edge at 0 in T2CC3 10 _B Compare enabled 11 _B Capture on write operation into register CC3L

Note: The T2CCU output is only actively driven on enabling the compare function.

T2CCU_COSHDW
T2CCU Compare Shadow Register (C0_H)
Reset Value: 00_H
RMAP: 0, PAGE: 2

7	6	5	4	3	2	1	0
ENSHDW	TXOV	COOUT5	COOUT4	COOUT3	COOUT2	COOUT1	COOUT0
rw	rw	rw	rw	rw	rw	rw	rw

Feature Discontinued

Field	Bits	Typ	Description
COOUTx (x = 0-5)	x	rwh	<p>Compare Channel x Output</p> <p><u>ENSHDW = 0</u> Any access to bit COOUTx accesses the corresponding bit of the internal 6-bit compare output register. Write to bit COOUTx will change the corresponding bit of the internal compare output register. Read to bit COOUTx returns the value of the bit in the internal compare output register. The content of the internal compare output register is output on the corresponding T2CCx pins when a valid compare match occurs for a channel in compare mode 1 or concurrent compare mode.</p> <p><i>Note: A concurrent write operation to clear bit ENSHDW and set/clear bit(s) COOUTx is effective for initializing the internal compare output register.</i></p> <p><u>ENSHDW = 1</u> In this case, accessing bit COOUTx accesses the actual SR bits. If COOUTx is set, the contents of register CCx will be transferred to corresponding channel x internal compare register. On set, COOUTx is held for one PCLK, then cleared automatically by hardware.</p> <p><i>Note: A concurrent write operation to set bit ENSHDW and bit(s) COOUTx (and clear bit TXOV) is effective for starting shadow transfer(s).</i></p>
TXOV	6	r/w	<p>Enable Shadow Transfer On Timer Overflow</p> <p>Set to enable for all compare channels, transfer of shadow value in T2CCU_CCx to the corresponding internal compare registers on overflow of their respective timer base.</p>

Field	Bits	Typ	Description
ENSHDW	7	rwh	Enable Shadow Transfer With COOUTx Clear to enable access of the internal compare output register via the COOUTx bits. Set to enable use of COOUTx bits for transfer of shadow value in T2CCU_C0Cx to the corresponding channel x internal compare register. <i>Note: When setting bit TXOV, bit ENSHDW will be cleared by internal hardware. To ensure bit ENSHDW is set, user must ensure bit TXOV is cleared.</i>

T2CCU_COCON
T2CCU Compare Control Register
(C0_H)
Reset Value: 00_H
RMAP: 0, PAGE: 3

7	6	5	4	3	2	1	0
CCM5	CCM4	CM5F	CM4F	POLB	POLA	COMOD	
rw	rw	rwh	rwh	rw	rw	rw	

Field	Bits	Typ	Description
COMOD	[1:0]	rw	Compare Mode Control (for compare channels) 0 _B Compare Mode 0 selected 01 _B Compare Mode 1 selected 10 _B Concurrent Compare mode selected 11 _B Reserved
POLA	2	rw	Compare Active Level for Channel Group A In compare mode 0, the bit defines the active level of the outputs of the compare channels belonging to the group. 0 _B High on Compare Match 1 _B Low on Compare Match
POLB	3	rw	Compare Active Level for Channel Group B In compare mode 0, the bit defines the active level of the outputs of the compare channels belonging to the group. 0 _B High on Compare Match 1 _B Low on Compare Match

Field	Bits	Typ	Description
CM4F¹⁾	4	rwh	Compare Channel 4 Interrupt Flag Set on a compare match event on compare channel 4. This bit has to be cleared by software.
CM5F¹⁾	5	rwh	Compare Channel 5 Interrupt Flag Set on a compare match event on compare channel 5. This bit has to be cleared by software.
CCM4	6	rw	Compare Enable for Channel 4 0 _B Compare disabled 1 _B Compare enabled See note below
CCM5	7	rw	Compare Enable for Channel 5 0 _B Compare disabled 1 _B Compare enabled See note below

1) In case of software and hardware access to a flag bit at the same time, hardware will have higher priority.

*Note: Compare channels 4 and 5 should be enabled in a separate step after setting the compare mode via bits **COMOD** (and active level **POLA/B** if compare mode 0). This is to ensure defined compare output according to the selected mode on starting the respective compare channel.*

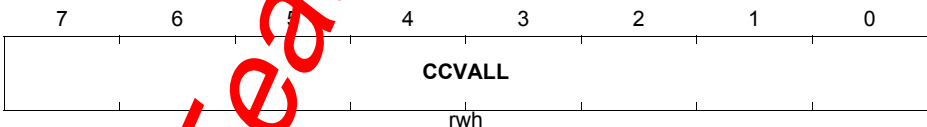
*Note: The interrupt enable for compare channels 4 and 5 is controlled by the respective bit of SCU SFR **MODIEN.CM4EN** and **CM5EN**.*

T2CCU_CCxL (x = 0-2)

T2CCU Capture/Compare Register x Low (C1_H+x*02_H)

Reset Value: 00_H

RMAP: 0, PAGE: 2

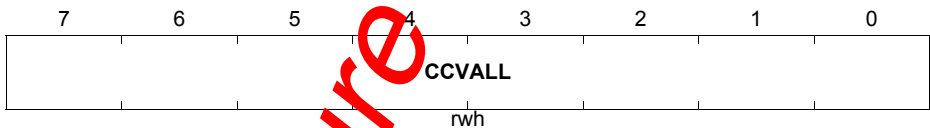


Field	Bits	Typ	Description
CCVALL	[7:0]	rwh	<p>Capture/Compare Low Byte Value for Channel x</p> <p>Capture On read, CCVALL returns the low byte of the last captured value.</p> <p>Compare CCVALL holds the low byte written. CCx acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when corresponding bit T2CCU_COSHADOW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value.</p> <p><i>Note: With T2CCU_CC4 and T2CCU_CC5, only the compare function is available.</i></p>

T2CCU_CCxL (x = 3-5)

T2CCU Capture/Compare Register - Low ($C1_H + x * 02_H - 3 * 02_H$) Reset Value: 00_H

RMAP: 0, PAGE: 3



Feature Discontinued

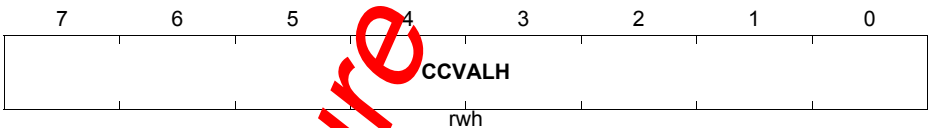
Field	Bits	Typ	Description
CCVALL	[7:0]	rwh	<p>Capture/Compare Low Byte Value for Channel x Capture On read, CCVALL returns the low byte of the last captured value.</p> <p>Compare CCVALL holds the low byte written. CCx acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when corresponding bit T2CCU_COSHLDW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value.</p> <p><i>Note: With T2CCU_CC4 and T2CCU_CC5, only the compare function is available.</i></p>

T2CCU_CCxH (x = 0-2)

T2CCU Capture/Compare Register High (C2_H+x*02_H)

Reset Value: 00_H

RMAP: 0, PAGE: 2



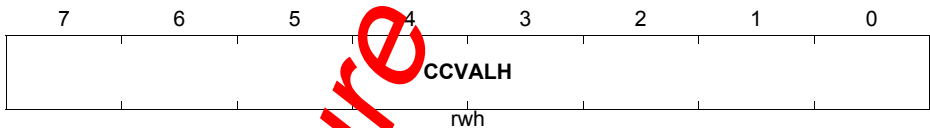
Feature Discontinued

Field	Bits	Typ	Description
CCVALH	[7:0]	rwh	<p>Capture/Compare High Byte Value for Channel x</p> <p><u>Capture</u> On read, CCVALH returns the high byte of the last captured value.</p> <p><u>Compare</u> CCVALH holds the high byte written. CCx acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when corresponding bit T2CCU_COSHDW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value.</p> <p><i>Note: With T2CCU_CC4 and T2CCU_CC5, only the compare function is available.</i></p>

T2CCU_CCxH (x = 3-5)

T2CCU Capture/Compare Register High (C2_H+x*02_H-3*02_H) Reset Value: 00_H

RMAP: 0, PAGE: 3



Feature Discontinued

Field	Bits	Typ	Description
CCVALH	[7:0]	rwh	<p>Capture/Compare High Byte Value for Channel x Capture On read, CCVALH returns the high byte of the last captured value.</p> <p>Compare CCVALH holds the high byte written. CCx acts as a shadow register to the channel x internal compare register. The value written is only transferred to the internal compare register when corresponding bit T2CCU_COSHDW.COOUTx is being set (with ENSHDW = 1), or on overflow of the channel timer base. The read value always return the last written value.</p> <p><i>Note: With T2CCU_CC4 and T2CCU_CC5, only the compare function is available.</i></p>

13.12.9 Implementation Details of T2CCU

This section describes the implementation details of the T2CCU module.

13.12.9.1 Interfaces of the T2CCU

Figure 13-60 gives an overview of the interconnections of the T2CCU. Timer 2 and T2CCU have direct interfaces to support the capture/compare operations.

The T2CCU shares the Timer 2 Bus Peripheral Interface (BPI). The BPI enables the T2CCU kernel to be attached to the 8-bit PD8 Bus. The BPI consists of a clock control logic which gates the clock input to both kernels, and an address decoder for SFRs in the Timer 2 and T2CCU kernels.

The interrupt lines of the T2CCU are connected similarly with that of Timer 2, to the CPU interrupt controller via the SCU.

The General Purpose I/O (GPIO) Ports provide the interface from the T2CCU to the external world.

The T2CCU timers can be suspended when OCDS enters Monitor Mode and has the Debug-Suspend signal activated, provided the respective timer suspend bits, T2SUSP and CCTSUSP (normally in SCU SFR MODSUSP), are set. Refer to SCU and OCDS chapters.

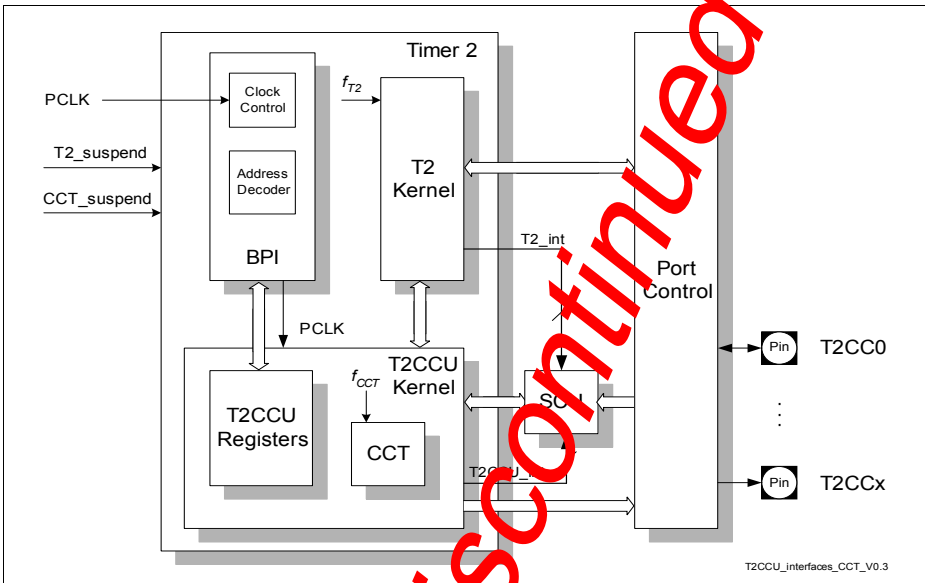


Figure 13-60 Interconnections of T2CCU

13.12.9.2 Design Options

Within the T2CCU kernel, following configurations should be provided:

1. Option for 4 or 6 compare channels depending on product.
2. Option for Concurrent Compare mode.
3. Option for removal of CCT timer and SFRs (and optionally other related components).

Feature Discontinued

13.13 Real-Time Clock

13.13.1 Overview

One of the TC2x_SCR's peripherals is the Real-Time Clock (RTC) that, once started, can work independently of the state of the rest of the microcontroller.

Features

- Periodic Wake-up Mode using either the 100 kHz clock or the 100 MHz/DIV clock
- Wake-up source during standby mode

13.13.2 System Information

This section provides system information relevant to the Real-Time Clock.

13.13.2.1 Interrupt Events and Assignment

Table 13-55 lists the interrupt event sources from the RTC, and the corresponding event interrupt enable bit and flag bit.

Table 13-55 RTC Interrupt Events

Event	Event Interrupt Enable Bit	Event Flag Bit
RTC compare/wake-up	RTC_RTCON.E_RTC	RTC_RTCON.CFRTC

Table 13-56 shows the interrupt node assignment for each RTC interrupt source.

Table 13-56 RTC Events' Interrupt Node Control

Event	Interrupt Node Enable Bit	Interrupt Node Flag Bit	Vector Address
RTC compare/wake-up	EN1.ECCIP3	–	6BH

13.13.2.2 Module Suspend Control

When the On-Chip Debug Support (OCDS) is in Monitor Mode (MMCR2.MMODE = 1) and the Debug-Suspend signal is active (MMCR2.DSUSP = 1), the timer/counter in RTC module in TC2x_SCR can be suspended based on the settings of their corresponding module suspend bits in register MODSUSP.

When suspended, only the timer stops counting as the counter input clock is gated off. The module is still clocked so that module registers are accessible.

13.13.3 Clock Source

RTC clock, f_{RTC} , is selectable between the 100 kHz clock or the 100 MHz/DIV clock via bit RTCLKSEL in RTCON register.

Once power-up, these clocks can operate irrespective of the state of the microcontroller. That is, it keeps running even when the TC2x_SCR has entered idle or the device has entered standby mode.

13.13.4 Real Time Clock Operation

The real-time clock consists of a 41-bit timer (including 8-bit prescaler) which count up. It contains a set of 4 count registers, collectively known as CNT register, that shows the current count value or the current time of the real-time clock. Before starting the real-time clock, CNT register can be written with any value. The value written is used as an initial value for the real-time clock when it is started. Another set of registers, RTCCR register, that consists of 4 registers can be used for interrupt generation. It can also be used to wake-up device from standby mode. The RTCCR register is also used to store the capture value when a capture event is triggered. The real-time clock is started by setting bit RTCC in the RTCON register to 1. This enables the input clock into the real-time clock timer.

13.13.4.1 Periodic Wake-up Mode

Figure 13-61 shows the real-time clock. It consists of a 41-bit general purposes timer. The PCLK is the input clock to the timer.

Before the real-time clock starts to run, CNT register of the real-time clock can be written with any values. The value written is used as an initial value for the real-time clock timer, when it is started by setting bit RTCC in the RTCON register to 1. This bit also enables the input clock into the real-time clock timer. CNT register is protected by the bit protection scheme as described in the SCU chapter. The initial count value can only be updated when the protection scheme is being disabled in the stop mode (RTCC = 0). The real-time clock's lower 9 bits, which serve as a prescaler into the 32-bit counters, CNT, are set to an initial value of 00000000_b when the RTC starts to run. The prescaler can be bypassed by setting bit RTPBYP in the RTCON register to 1.

Feature Discontinued

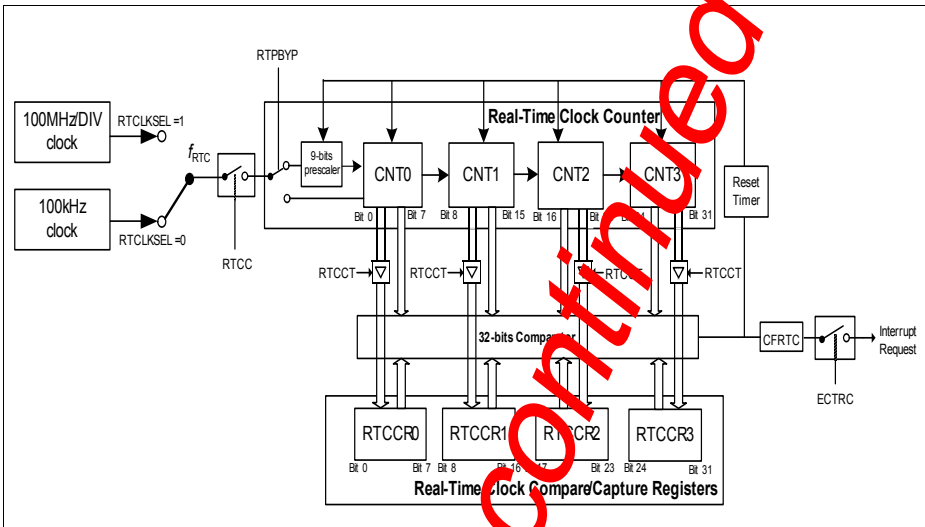


Figure 13-61 Real-time Clock Periodic Wake-up Mode with 9-bit prescaler

While the real-time clock is in operation, the contents of RTCCR register will be compared to the real-time clock counter (CNT). An interrupt will be generated when the contents are equal if ECRTC is set to 1; bit CFRTC in register RTON will be set. The CFRTC flag can be monitored for the real-time clock wake-up request, but the flag has to be cleared to 0 by user software. In such situation, the real-time clock is reset and count from zero again.

With RTC still running in standby mode, it is able to wake-up at a fixed time by programming the number of count value that is equivalent to the length of time to wake-up in the RTCCR register. The real-time clock can generate a wake-up request to the main controller if bit STDBYWRP_RTCWKSEL is set to 1.

A capture event could be triggered by setting RTCCT bit in RTON register to 1. The previous content in the RTCCR register will be overwritten with the captured CNT values after 2 CPU clock cycles. An update of the actual compared value is necessary once a captured event is triggered. In TC2x_SCR, it is recommended to trigger a capture event to read the value of the RTC counter (CNT). There is a potential of reading a wrong 32 bits real-time value while the RTC is in running mode as only 8 bit of data could be fetch at one time.

The real-time clock stops counting and CNT register holds the last value when the RTCC bit is set to 0. Setting this bit subsequently will start a new counting sequence which begin with the stop count.

Note: A compare match event could happen concurrently with the capture event when both events happen within the same clock cycle.

13.13.5 Power Saving Mode Option

Once started, the real-time clock continues counting until the bit FTCON.RTCC is cleared. The real-time clock is not affected by the idle mode of the TC2x_SCR, and continues counting in standby mode provided the PCLK clock is available. In addition, the real-time clock will not automatically disable the operation when there is a clock failure.

Feature Discontinued

13.13.6 Real-Time Clock Register

13.13.6.1 Register Mapping

9 SFRs are used to control the operation of real-time clock. They can be accessed from the standard (non-mapped) SFR area. The registers are described as follows.

Table 13-57 lists the addresses of these SFRs.

Table 13-57 Register Map

Address	Register
95 _H	RTCON
E1 _H	CNT0
E2 _H	CNT1
E3 _H	CNT2
E4 _H	CNT3
E7 _H	RTCCR0
E9 _H	RTCCR1
EA _H	RTCCR2
EB _H	RTCCR3

Feature Discontinued

13.13.6.2 Register Description

RTC_RTCON

 Real-Time Clock Control Register (95_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
0	CFRTC	0	ECRTC	RTCCT	RTPBYP	RTCLKSEL	RTCC
r	rwh	r	rw	rwh	rw	rw	rw

Field	Bits	Type	Description
RTCC	0	rw	Real-Time Clock Start/Stop Control 0 _B Stop real-time clock Operation 1 _B Start real-time clock Operation
RTCLKSEL	1	rw	Real-Time Clock Input Clock Selection 0 _B 100 kHz clock is selected. 1 _B 100 MHz 3.3V clock is selected. <i>Note: Real-time clock must be in stop operation before the clock setting can be changed.</i>
RTPBYP	2	rw	Real-Time Clock 9-bit Prescaler Bypass 0 _B 9-bit prescaler is not bypassed. 1 _B 9-bit prescaler is bypassed. <i>Note: Real-time clock must be in stop operation before the prescaler setting can be changed.</i>
RTCCT	3	rwh	Real-Time Clock Capture Event Trigger 0 _B No action 1 _B Capture event is triggered immediately On set, this bit is held for two PCLK clock cycle, then cleared by hardware. Reading this bit always return 0.
ECRTC	4	rw	Real-Time Clock Compare Interrupt Enable 0 _B Disable real-time clock compare interrupt. 1 _B Enable real-time clock compare interrupt.
CFRTC	6	rwn	Real-Time Clock Compare Flag This bit is set by hardware when there is a compare match and can only be cleared by software. A wake-up request is generated only if the TC2x_SCR is in standby mode.

Field	Bits	Type	Description
0	5, 7	r	Reserved Returns 0 if read; should be written with 0.

4 count registers, CNT0 - CNT3 are used to represent the most significant 32 bits of the 41-bits real-time clock timer.

RTC_CNT0

Count Clock Register 0

 (E1_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
CNT_VAL							
rwh							

Field	Bits	Type	Description
CNT_VAL	[7:0]	rwh	Real-Time Clock Count Value [7:0] These bits represent counter value [7:0] of the current real-time clock timer.

RTC_CNT1

Count Clock Register 1

 (E2_H)

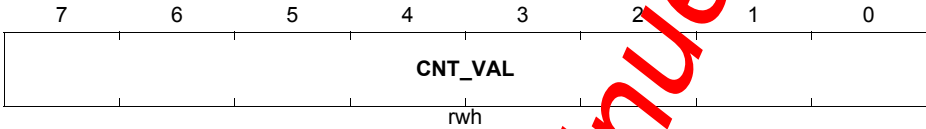
 Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
CNT_VAL							
rwh							

Field	Bits	Type	Description
CNT_VAL	[15:8]	rwh	Real-Time Clock Count Value [15:8] These bits represent counter value [15:8] of the current real-time clock timer.

RTC_CNT2
Count Clock Register 2 (E3_H) **Reset Value: 00_H**
RMAP: 0, PAGE: X



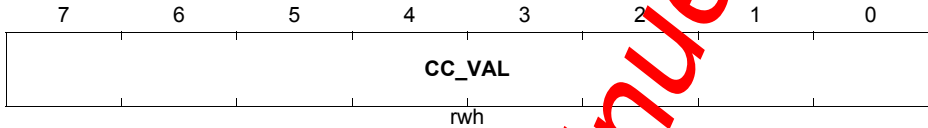
Field	Bits	Type	Description
CNT_VAL	[7:0]	rwh	Real-Time Clock Count Value [23:16] These bits represent counter value [23:16] of the current real-time clock timer.

RTC_CNT3
Count Clock Register 3 (E4_H) **Reset Value: 00_H**
RMAP: 0, PAGE: X

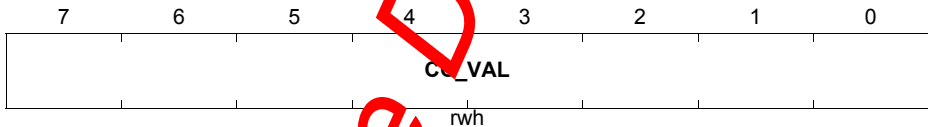


Field	Bits	Type	Description
CNT_VAL	[7:0]	rwh	Real-Time Clock Count Value [31:23] These bits represent counter value [31:23] of the current real-time clock timer.

4 compare and capture registers, RTCCR0 - RTCCR3 are used to represent 32 bits of compare values that will generate a compare event when it matches the counter values as specified in CNT register. When a capture event is triggered by setting RTCCT bit to 1, the content in the FRCR register will be overwritten with the captured CNT values after 2 CPU clock cycles. An update of the actual compared value is necessary once a captured event is triggered.

RTC_RTCCR0
Real-Time Clock Compare/Capture Register 0 (E7_H)
Reset Value: 00_H
RMAP: 0, PAGE: X


Field	Bits	Type	Description
CC_VAL	[7:0]	rwh	Compare/Capture Value [7:0] These bits represent the compare/capture value [7:0] of the 32-bits value that could generate a compare interrupt when it matches with the current counter values.

RTC_RTCCR1
Real-Time Clock Compare/Capture Register 1 (E9_H)
Reset Value: 00_H
RMAP: 0, PAGE: X


Field	Bits	Type	Description
CC_VAL	[7:0]	rwh	Compare/Capture Value [15:8] These bits represent compare/capture value [15:8] of the 32-bits value that could generate a compare interrupt when it matches with the current counter values.

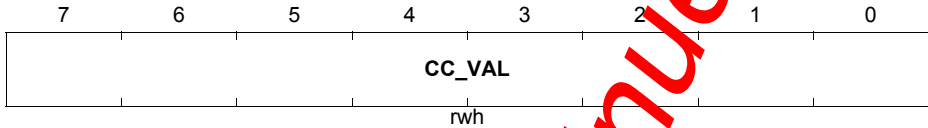
Feature Discontinued

RTC_RTCCR2

 Real-Time Clock Compare/Capture Register 2(EA_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X



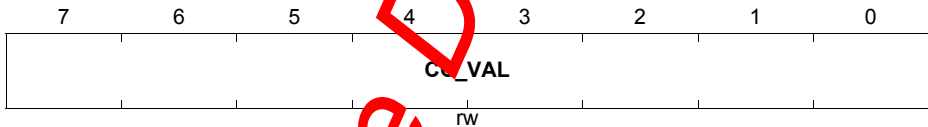
Field	Bits	Type	Description
CC_VAL	[7:0]	rwh	Compare/Capture Value [23:16] These bits represent compare/capture value [23:16] of the 32-bits value that could generate a compare interrupt when it matches with the current counter value.

RTC_RTCCR3

 Real-Time Clock Compare/Capture Register 3(EB_H)

 Reset Value: 00_H

RMAP: 0, PAGE: X



Field	Bits	Type	Description
CC_VAL	[7:0]	rw	Compare/Capture Value [31:23] These bits represent compare/capture value [31:23] of the 32-bits value that could generate a compare interrupt when it matches with the current counter value.

Feature Discontinued

13.14 Universal Asynchronous Receiver/Transmitter

13.15 Overview

The UART provides a full-duplex asynchronous receiver/transmitter, i.e., it can transmit and receive simultaneously. It is also receive-buffered, i.e., it can commence reception of a second byte before a previously received byte has been read from the receive register. However, if the first byte still has not been read by the time reception of the second byte is complete, one of the bytes will be lost.

UART Feature:

- Full-duplex asynchronous modes
 - 8-bit or 9-bit data frames, LSB first
 - fixed or variable baud rate
- Receive buffered
- Multiprocessor communication
- Interrupt generation on the completion of a data transmission or reception

13.15.1 System Information

This section provides system information relevant to the UART.

13.15.1.1 Pinning

In mode 0 (the serial port behaves as shift register), data is shifted in through RXD and out through RXDO, while the TXD pin is used to provide a shift clock which can be used by external devices to clock data in and out. In modes 1, 2 and 3, the port behaves as an UART. Data is transmitted on TXD and received on RXD.

The UART I/O pins are generally assigned as alternate functions to general purpose I/O ports. Bit URRIS in register MODPISEL0 is used to select one of the RXD input function. The UART pin assignment is shown in [Table 13-58](#).

Table 13-58 UART Pin Functions

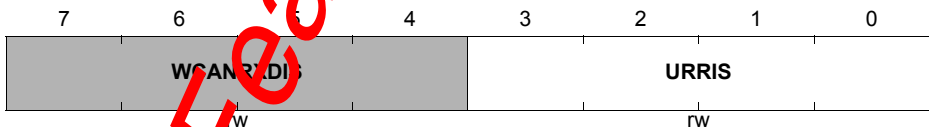
Pin	Function	Description	Selected By
P02.1	RXDA	UART Receive Data Input A	MODPISEL0.URRIS = 0000 _B SCR_P02_IOC1.PC1 = 0XX10 _B
P02.3	RXDB	UART Receive Data Input B	MODPISEL0.URRIS = 0001 _B SCR_P02_IOC3.PC3 = 0XX10 _B
P02.4	RXDC	UART Receive Data Input C	MODPISEL0.URRIS = 0010 _B SCR_P02_IOC4.PC4 = 0XX10 _B

Table 13-58 UART Pin Functions (cont'd)

Pin	Function	Description	Selected By
P00.1	RXDD	UART Receive Data Input D	MODPISEL0.URRIS = 0011 _B SCR_P00_IOCR1.PC1 = 0XX10 _B
P00.3	RXDE	UART Receive Data Input E	MODPISEL0.URRIS = 0100 _B SCR_P00_IOCR3.PC3 = 0XX10 _B
P14.1 ¹⁾	RXDF	UART Receive Data Input F	MODPISEL0.URRIS = 0101 _B P14_IOCR0.PC1 = 0XX10 _B
P15.1 ¹⁾	RXDG	UART Receive Data Input G	MODPISEL0.URRIS = 0110 _B P15_IOCR0.PC1 = 0XX10 _B
P02.0	TXD	UART Transmit Data Output / Clock output	SCR_P02_IOCR0.PC0 = 1X001 _B
P02.2			SCR_P02_IOCR2.PC2 = 1X001 _B
P02.5			SCR_P02_IOCR5.PC5 = 1X001 _B
P00.0			SCR_P00_IOCR0.PC0 = 1X001 _B
P00.2			SCR_P00_IOCR2.PC2 = 1X001 _B
P02.1			SCR_P02_IOCR1.PC1 = 1X001 _B
P02.3	RXDO	UART Transmit Data Output	SCR_P02_IOCR3.PC3 = 1X001 _B
P02.4			SCR_P02_IOCR4.PC4 = 1X001 _B
P00.1			SCR_P00_IOCR1.PC1 = 1X001 _B
P00.3			SCR_P00_IOCR3.PC3 = 1X001 _B

1) The pad control of this pin is in the 32-bit Ports module. The input function of the pad must be enabled before this pin can be used as RXD input.

The bit field PAGE of SCU_PAGE register must be programmed before accessing the MODPISEL0 register.

MODPISEL0
Peripheral Input Select Register 0
(F2_H)
Reset Value: 00_H
RMAP: 0, PAGE: 3


Field	Bits	Type	Description
URRIS	[3:0]	rw	UART Receiver Input Select
			0000 _B UART Receiver Input RX0A is selected.
			0001 _B UART Receiver Input RX0B is selected.
			0010 _B UART Receiver Input RX0C is selected.
			0011 _B UART Receiver Input RX0D is selected.
			0100 _B UART Receiver Input RX0E is selected.
			0101 _B UART Receiver Input RX0F is selected.
			0110 _B UART Receiver Input RX0G is selected.
			0111 _B Reserved
			1000 _B Reserved
			1001 _B Reserved
			1010 _B Reserved
			1011 _B Reserved
			1100 _B Reserved
			1101 _B Reserved
			1110 _B Reserved
1111 _B Reserved			

13.15.1.2 Clocking Configuration

The UART runs on the PCLK at a frequency of up to 20 MHz.

13.15.1.3 Interrupt Events and Assignment

Table 13-59 lists the interrupt event sources from the UART, and the corresponding event interrupt enable bit and flag bit.

Table 13-59 UART Interrupt Events

Event	Event Interrupt Enable Bit	Event Flag Bit
Data Received	URRIS	SCON.RI
Data Transmitted	URRIS	SCON.TI

Table 13-60 shows the interrupt node assignment for each UART interrupt source.

Table 13-60 UART Events' Interrupt Node Control

Event	Interrupt Node Enable Bit	Interrupt Node Flag Bit	Vector Address
Data Received	IEN0.ES	–	23H
Data Transmitted			

13.15.2 UART Modes

The UART can be used in four different modes. In mode 0, it operates as an 8-bit shift register. In mode 1, it operates as an 8-bit serial port. In modes 2 and 3, it operates as a 9-bit serial port. The only difference between mode 2 and mode 3 is the baud rate, which is fixed in mode 2 but variable in mode 3. The variable baud rate is set by the underflow rate on the dedicated baud-rate generator.

The different modes are selected by setting bits SM0 and SM1 to their corresponding values, as shown in [Table 13-61](#).

Table 13-61 UART Modes

SM0	SM1	Operating Mode	Baud Rate
0	0	Mode 0: 8-bit shift register	$f_{PCLK}/2$
0	1	Mode 1: 8-bit shift UART	Variable
1	0	Mode 2: 9-bit shift UART	$f_{PCLK}/64$ or $f_{PCLK}/32$
1	1	Mode 3: 9-bit shift UART	Variable

13.15.2.1 Mode 0, 8-Bit Shift Register, Fixed Baud Rate

In mode 0, the serial port behaves as an 8-bit shift register. Data is shifted in through RXD, and out through RXDO, while the TXD line is used to provide a shift clock which can be used by external devices to clock data in and out.

The transmission cycle is activated by a write to SBUF. One machine cycle later, the data has been written to the transmit shift register with a 1 at the 9th bit position. For the next seven machine cycles, the contents of the transmit shift register are shifted right one position and a zero shifted in from the left so that when the MSB of the data byte is at the output position, it has a 1 and a sequence of zeros to its left. The control block then executes one last shift before setting the TI bit.

Reception is started by the condition REN = 1 and RI = 0. At the start of the reception cycle, 11111110_b is written to the receive shift register. In each machine cycle that follows, the contents of the shift register are shifted left one position and the value sampled on the RXD line in the same machine cycle is shifted in from the right. When

the 0 of the initial byte reaches the leftmost position, the control block executes one last shift, loads SBUF and sets the RI bit.

The baud rate for the transfer is fixed at $f_{PCLK}/2$ where f_{PCLK} is the input clock frequency, i.e. one bit per machine cycle.

13.15.2.2 Mode 1, 8-Bit UART, Variable Baud Rate

In mode 1, the UART behaves as an 8-bit serial port. A start bit (0), 8 data bits, and a stop bit (1) are transmitted on TXD or received on RXD at a variable baud rate.

The transmission cycle is activated by a write to SBUF. The data is transferred to the transmit register and a 1 is loaded to the 9th bit position (as in mode 0). At phase 1 of the machine cycle after the next rollover in the divide-by-16 counter, the start bit is copied to TXD, and data is activated one bit time later. One bit time after the data is activated, the data starts getting shifted right with zeros shifted in from the left. When the MSB gets to the output position, the control block executes one last shift and sets the TI bit.

Reception is started by a high to low transition on RXD (sampled at 16 times the baud rate). The divide-by-16 counter is then reset and $1111\ 1111_B$ is written to the receive register. If a valid start bit (0) is then detected (based on two out of three samples), it is shifted into the register followed by 8 data bits. If the transition is not followed by a valid start bit, the controller goes back to looking for a high to low transition on RXD. When the start bit reaches the leftmost position, the control block executes one last shift, then loads SBUF with the 8 data bits, loads R8 (SCON.2) with the stop bit, and sets the RI bit, provided RI = 0, and either SM2 = 0 (see [Section 13.15.3](#)) or the received stop bit = 1. If none of these conditions is met, the received byte is lost.

The associated timings for transmit/receive in mode 1 are illustrated in [Figure 13-62](#).

Feature Discontinued

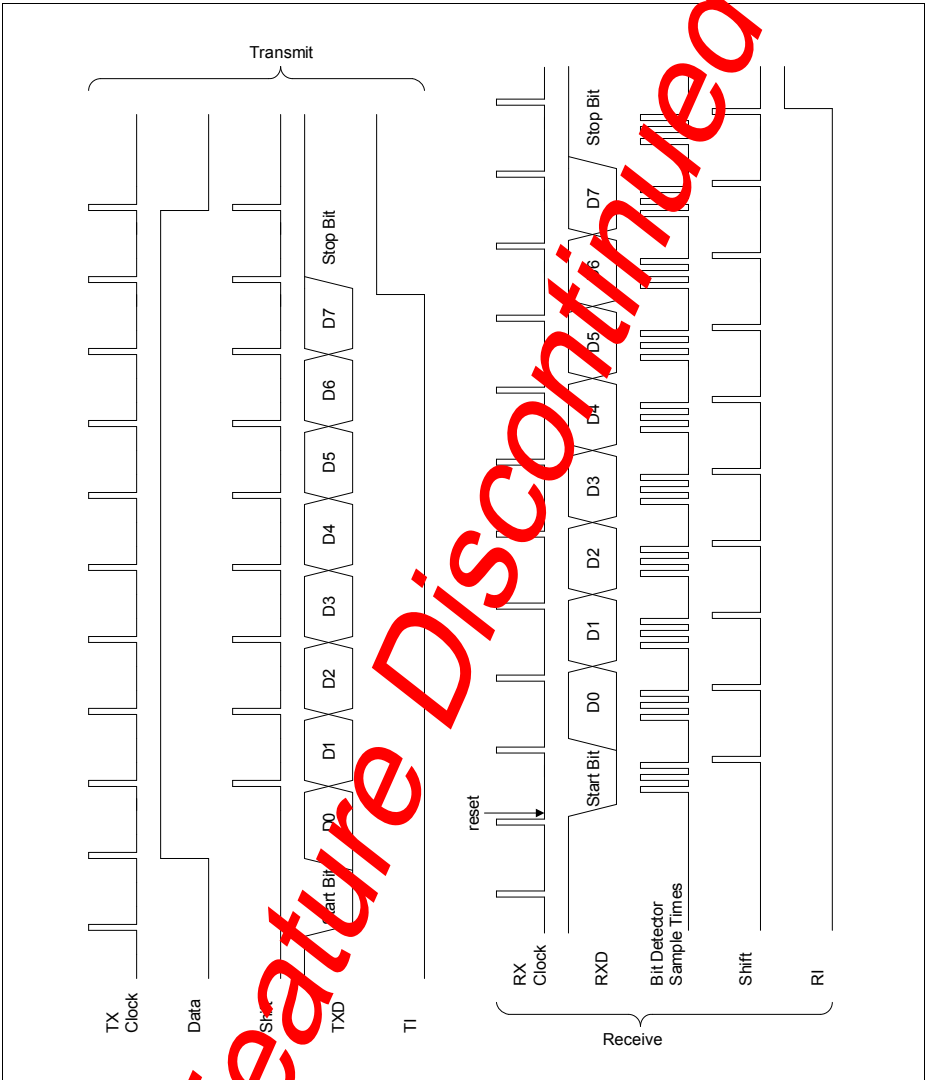


Figure 13-62 Serial Interface, Mode 1, Timing Diagram

13.15.2.3 Mode 2, 9-Bit UART, Fixed Baud Rate

In mode 2, the UART behaves as a 9-bit serial port. A start bit (0), 8 data bits plus a programmable 9th bit and a stop bit (1) are transmitted on TXD or received on RXD. The 9th bit for transmission is taken from TB8 (SCON.3) while for reception, the 9th bit received is placed in RB8 (SCON.2).

The transmission cycle is activated by a write to SBUF. The data is transferred to the transmit register and TB8 is copied into the 9th bit position. At phase 1 of the machine cycle following the next rollover in the divide-by-16 counter, the start bit is copied to TXD and data is activated one bit time later. One bit time after the data is activated, the data starts shifting right. For the first shift, a stop bit (1) is shifted in from the left and for subsequent shifts, zeros are shifted in. When the TB8 bit gets to the output position, the control block executes one last shift and sets the TI bit.

Reception is started by a high to low transition on RXD (sampled at 16 times the baud rate). The divide-by-16 counter is then reset and 11111111_B is written to the receive register. If a valid start bit (0) is then detected (based on two out of three samples), it is shifted into the register followed by 8 data bits. If the transition is not followed by a valid start bit, the controller goes back to looking for a high to low transition on RXD. When the start bit reaches the leftmost position, the control block executes one last shift, then loads SBUF with the 8 data bits, loads RB8 (SCON.2) with the 9th data bit, and sets the RI bit, provided RI = 0, and either SM2 = 0 (see [Section 13.15.3](#)) or the 9th bit = 1. If none of these conditions is met, the received byte is lost.

The baud rate for the transfer is either $f_{PCLK}/64$ or $f_{PCLK}/32$, depending on the setting of the top bit (SMOD) of the PCON (Power Control) register, which acts as a Double Baud Rate selector.

13.15.2.4 Mode 3, 9-Bit UART, Variable Baud Rate

Mode 3 is the same as mode 2 in all respects except that the baud rate is variable.

In all modes, transmission is initiated by any instruction that uses SBUF as a destination register. Reception is initiated in the modes by the incoming start bit if REN = 1.

The serial interface also provides interrupt requests when transmission or reception of the frames has been completed. The corresponding interrupt request flags are TI or RI, respectively. If the serial interrupt is not used (i.e., serial interrupt not enabled), TI and RI can also be used for polling the serial interface.

The associated timing for transmit/receive in modes 2 and 3 are illustrated in [Figure 13-63](#).

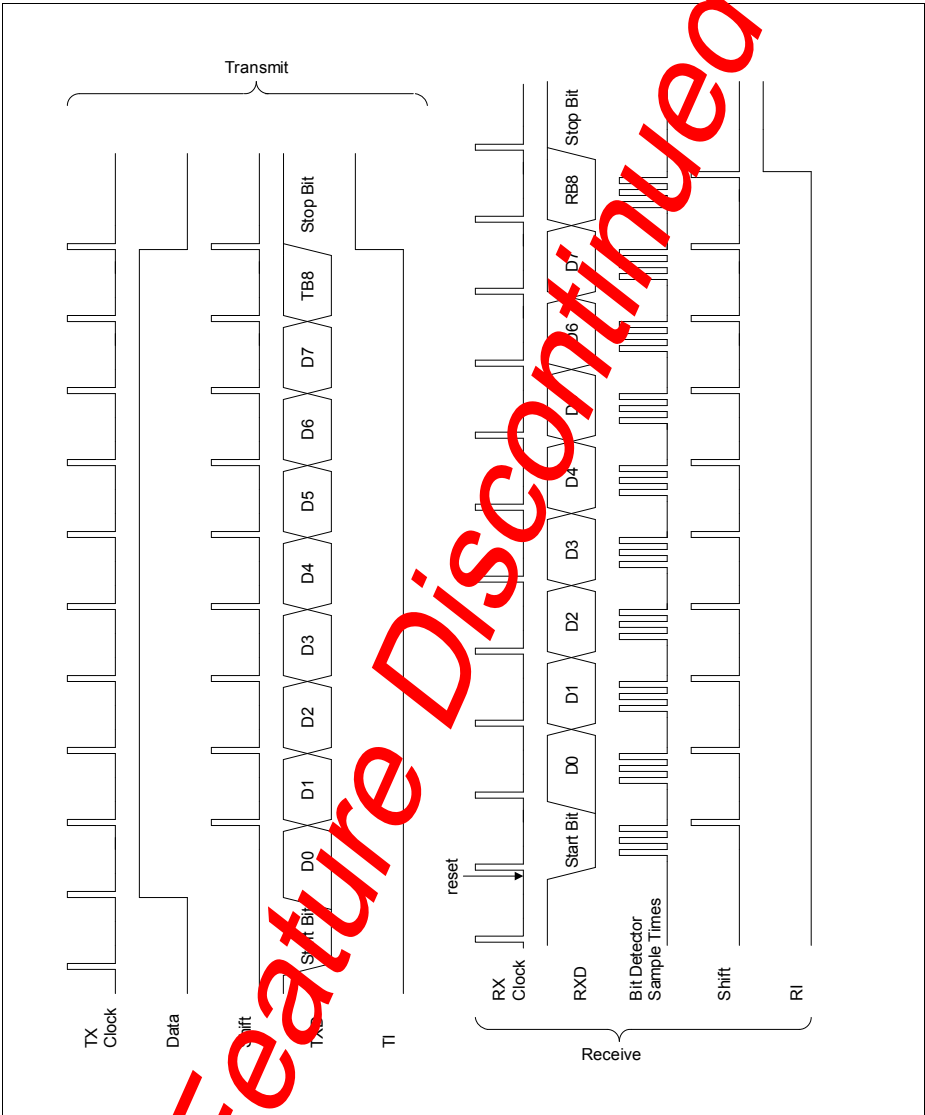


Figure 13-63 Serial Interface, Modes 2 and 3, Timing Diagram

13.15.3 Multiprocessor Communication

Modes 2 and 3 have a special provision for multiprocessor communication using a system of address bytes with bit 9 = 1 and data bytes with bit 9 = 0. In these modes, 9 data bits are received. The 9th data bit goes into RB8. The communication always ends with one stop bit. The port can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB8 = 1.

This feature is enabled by setting bit SM2 in SCON. One of the ways to use this feature in multiprocessor systems is described in the following paragraph.

When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address byte that identifies the target slave. An address byte differs from a data byte in that the 9th bit is 1 in an address byte and 0 in a data byte. With SM2 = 1, no slave will be interrupted by a data byte. An address byte, however, will interrupt all slaves, so that each slave can examine the received byte and see if it is being addressed. The addressed slave will clear its SM2 bit and prepare to receive the data bytes that will be coming. The slaves that were not being addressed retain their SM2s as set and ignore the incoming data bytes.

Bit SM2 has no effect in mode 0. SM2 can be used in mode 1 to check the validity of the stop bit. In a mode 1 reception, if SM2 = 1, the receive interrupt will not be activated unless a valid stop bit is received.

Feature Discontinued

13.15.4 Baud Rate Generation

There are several ways to generate the baud rate clock for the serial ports, depending on the mode in which they are operating.

The baud rates in modes 0 and 2 are fixed, so they use the fixed clock, (see [Section 13.15.4.1](#)). In modes 1 and 3, the variable baud rate is generated using the UART baud-rate generator (see [Section 13.15.4.2](#)).

“Baud rate clock” and “baud rate” must be distinguished from each other. The serial interface requires a clock rate that is 16 times the baud rate for internal synchronization. Therefore, the UART baud-rate generator must provide a “baud rate clock” to the serial interface where it is divided by 16 to obtain the actual “baud rate”. The abbreviation f_{PCLK} refers to the input clock frequency.

13.15.4.1 Fixed Clock

The baud rates in modes 0 and 2 are fixed. However, while the baud rate in mode 0 can only be $f_{PCLK}/2$, the baud rate in mode 2 can be selected as either $f_{PCLK}/64$ or $f_{PCLK}/32$ depending on bit SMOD in the PCON register.

Bit SMOD acts as a double baud rate selector, as shown in [Equation \(13.2\)](#). If SMOD = 0 (value after reset), the baud rate is 1/64 of the input clock frequency f_{PCLK} . If SMOD = 1, the baud rate is 1/32 of f_{PCLK} .

(13.2)

$$\text{Mode 2 baud rate} = \frac{2^{\text{SMOD}}}{64} \times f_{PCLK}$$

13.15.4.2 UART Baud-rate Generator

The UART baud-rate generator is used to generate the variable baud rate for the UART in modes 1 and 3. It has a programmable 11-bit reload value, 3-bit prescaler and 5-bit fractional divider.

The baud-rate generator is clocked derived via a prescaler (f_{DIV}) from the input clock f_{PCLK} . The baud rate timer counts downwards and can be started or stopped through the baud rate control run bit PCON.R. Each underflow of the timer provides one clock pulse to the serial channel. The timer is reloaded with the 11-bit BR_VALUE stored in its reload register BG each time it underflows. The duration between underflows depends on the ‘n’ value in the fractional divider, which can be selected by the bits BGL.FD_SEL. ‘n’ times out of 32, the timer counts one cycle more than specified by BR_VALUE. The prescaler is selected by the bits BCON.BRPRE.

Register BG is the dual-function Baud-rate Generator/Reload register. Reading BG returns the contents of the timer, while writing to BG (low byte) always updates the reload register.

The BG should be written only when BCON.R is 0. An auto-reload of the timer with the contents of the reload register is performed one instruction cycle after the next time BCON.R is set. Any write to BG, while BCON.R is set, will be ignored.

The baud rate of the baud-rate generator depends on the following bits and register values:

- Input clock f_{PCLK}
- Value of bit field BCON.BRPRE.
- Value of bit field BG.FD_SEL
- Value of the 11-bit reload value BG.BR_VALUE

Figure 13-64 shows a simplified block diagram of the baud rate generator.

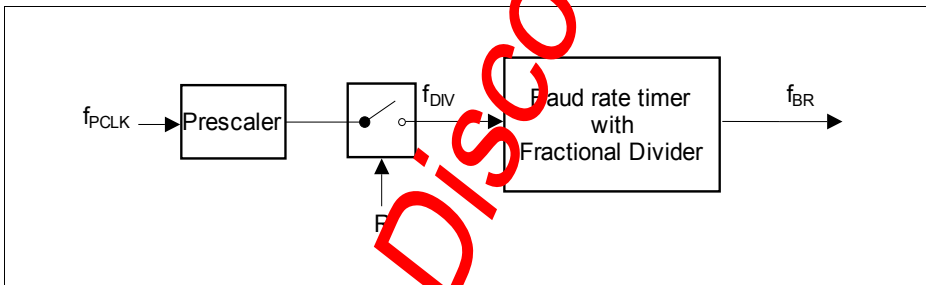


Figure 13-64 Simplified Baud Rate Generator Block Diagram

The following formula calculate the final baud rate.

(13.3)

$$\text{Baud rate} = \frac{f_{PCLK}}{16 \times \text{PRE} \times \left(\text{BR_VALUE} + \frac{n}{32} \right)}$$

The value of PRE (prescaler) is chosen by the bit field BCON.BRPRE. BR_VALUE represents the contents of the reload value, taken as unsigned 11-bit integer from the bit field BG.BR_VALUE. $\frac{n}{32}$ is defined by the fractional divider selection in bit field BG.FDSEL.

The maximum baud rate that can be generated is limited to $f_{PCLK}/32$. Hence, for a module clock of 20 MHz, the maximum achievable baud rate is 0.625 MBaud.

Table 13-62 lists various commonly used baud rates together with their corresponding parameter settings and the deviation errors compared to the intended baud rate.

Table 13-62 Typical Baud Rates of UART ($f_{PCLK} = 20 \text{ MHz}$)

Baud rate ($f_{PCLK} =$ 20 MHz)	PRE	Reload Value (BR_VALUE)	Numerator of Fractional Value (FD_NUM)	BG Register	Deviation Error
115.2 kBaud	1 (BRPRE = 000)	10 (0A _H)	27 (1B _H)	015B _H	+0.06%
20 kBaud	1 (BRPRE = 000)	62 (3E _H)	16 (10 _H)	07D0 _H	+0.00%
19.2 kBaud	1 (BRPRE = 000)	65 (41 _H)	3 (03 _H)	0823 _H	+0.02%
9600 Baud	2 (BRPRE = 001)	65 (41 _H)	3 (03 _H)	0823 _H	+0.02%
4800 Baud	4 (BRPRE = 010)	65 (41 _H)	3 (03 _H)	0823 _H	+0.02%
2400 Baud	8 (BRPRE = 011)	65 (41 _H)	3 (03 _H)	0823 _H	+0.02%

1) The value of the 16-bit BG register is obtained by concatenating the 11-bit BRVALUE and 5-bit FD_NUM into a 16-bit value.

Feature Discontinued

13.15.5 LIN Support in UART

The UART module can be used to support the Local Interconnect Network (LIN) protocol for both master and slave operations. The LIN baud rate detection feature, which consists of the hardware logic for Break and Synch Field detection, provides the capability to detect the baud rate within LIN protocol using Timer 2. This allows the UART module to be synchronized to the LIN baud rate for data transmission and reception.

13.15.5.1 LIN Protocol

LIN is a holistic communication concept for local interconnected networks in vehicles. The communication is based on the SCI (UART) data format, a single-master/multiple-slave concept, a clock synchronization for nodes without stabilized time base. An attractive feature of LIN is self-synchronization of the slave nodes without a crystal or ceramic resonator, which significantly reduces the cost of hardware platform. Hence, the baud rate must be calculated and returned with every message frame.

The structure of a LIN frame is shown in [Figure 13-65](#). The frame consists of the:

- header, which comprises a Break (13-bit time low), Synch Byte (55_H), and ID field
- response time
- data bytes (according to UART protocol)
- checksum

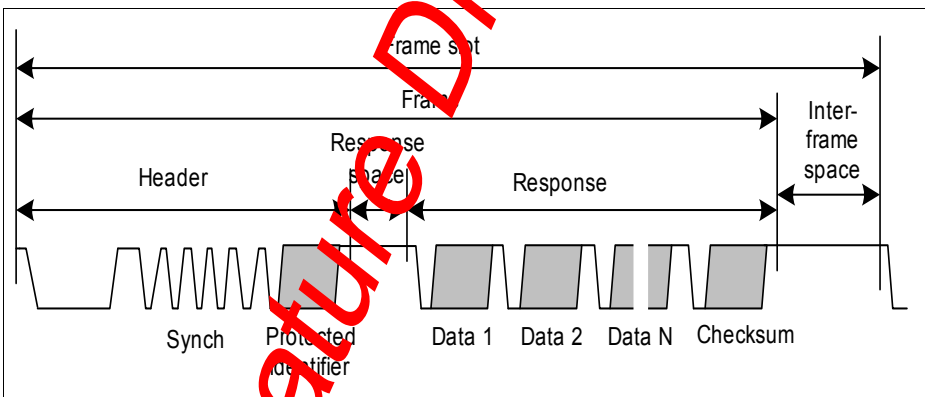


Figure 13-65 The Structure of LIN Frame

Each byte field is transmitted as a serial byte, as shown in [Figure 13-66](#). The LSB of the data is sent first and the MSB is sent last. The start bit is encoded as a bit with value zero (dominant) and the stop bit is encoded as a bit with value one (recessive).

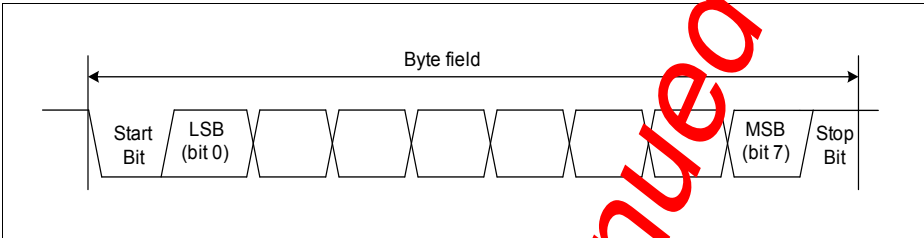


Figure 13-66 The Structure of Byte Field

The break is used to signal the beginning of a new frame. It is the only field that does not comply with [Figure 13-66](#). A break is always generated by the master task (in the master mode) and it must be at least 13 bits of dominant value, including the start bit, followed by a break delimiter, as shown in [Figure 13-67](#). The break delimiter will be at least one nominal bit time long.

A slave node will use a break detection threshold of 11 nominal bit times.



Figure 13-67 The Break Field

Synch Byte is a specific pattern for determination of time base. The byte field is with the data value 55_{Hex}, as shown in [Figure 13-68](#).

A slave task is always able to detect the Break/Synch sequence, even if it expects a byte field (assuming the byte fields are separated from each other). If this happens, detection of the Break/Synch sequence will abort the transfer in progress and processing of the new frame will commence.

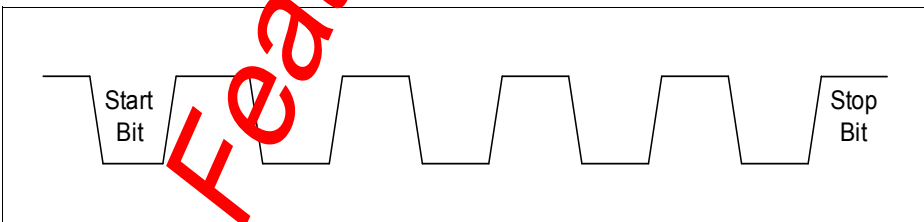


Figure 13-68 The Synch Byte Field

The slave task will receive and transmit data when an appropriate ID is sent by the master:

1. Slave waits for Synch Break
2. Slave synchronizes on Synch Byte
3. Slave snoops for ID
4. According to ID, slave determines whether to receive or transmit data, or do nothing
5. When transmitting, the slave sends 2, 4 or 8 data bytes, followed by check byte

13.15.5.2 LIN Header Transmission

LIN header transmission is only applicable in master mode. In the LIN communication, a master task decides when and which frame is to be transferred on the bus. It also identifies a slave task to provide the data transported by each frame. The information needed for the handshaking between the master and slave tasks is provided by the master task through the header portion of the frame.

The header consists of a break and synch pattern followed by an identifier. Among these three fields, only the break pattern cannot be transmitted as a normal 8-bit UART data. The break must contain a dominant value of 13 bits or more to ensure proper synchronization of slave nodes.

In the LIN communication, a slave task is required to be synchronized at the beginning of the protected identifier field of frame. For this purpose, every frame starts with a sequence consisting of a break field followed by a synch byte field. This sequence is unique and provides enough information for any slave task to detect the beginning of a new frame and be synchronized at the start of the identifier field.

13.15.5.3 Automatic Synchronization to the Host

Upon entering LIN communication, a connection is established and the transfer speed (baud rate) of the serial communication partner (host) is automatically synchronized in the following steps that are to be included in user software:

STEP 1: Initialize interface for reception and timer for baud rate measurement

STEP 2: Wait for an incoming LIN frame from host

STEP 3: Synchronize the baud rate to the host

STEP 4: Enter for Master Request Frame or for Slave Response Frame

The next section, [Section 13.15.5.4](#), provides some hints on setting up the microcontroller for baud rate detection of LIN.

*Note: Re-synchronization and setup of baud rate are always done for **every** Master Request Header or Slave Response Header LIN frame.*

13.15.5.4 Initialization of Break/Synch Field Detection Logic

The LIN baud rate detection feature provides the capability to detect the baud rate within the LIN protocol using Timer 2. Initialization consists of:

- Serial port of the microcontroller set to Mode 1 (8-bit UART, variable baud rate) for communication.
- Provide the baud rate range via bit field BCON.BGSEL.
- Toggle BCON.BRDIS bit (set the bit to 1 before clearing it back to 0) to initialize the Break/Synch detection logic.
- Clear all status flags LINST.BRK, LINST.EOFSYN and LINST.ERRSYN to 0.
- Timer 2 is set to capture mode with falling edge trigger at pin T2EX. Bit T2MOD.EDGESEL is set to 0 by default and bit T2CON.CP/RL2 is set to 1.
- Timer 2 external events are enabled. T2CON.EXEN2 is set to 1. (EXF2 flag is set when a negative transition occurs at pin T2EX)
- f_{T2} can be configured by bit field T2MOD.T2PREF.

Note: It is also recommended to toggle the BCON.BRDIS bit after the reception of each complete LIN frame to avoid a wrong Break field detection in noisy environments (i.e. spikes on the LIN bus).

13.15.5.5 Baud Rate Range Selection

The Break/Synch Field detection logic supports a maximum number of bits in the Break field as defined by [Equation \(13.4\)](#).

$$\text{Maximum number of bits} = \text{Baud Rate} \times \frac{4095}{\text{Sample Frequency}} \quad (13.4)$$

The sample frequency is given by [Equation \(13.5\)](#).

$$\text{Sample Frequency} = \frac{\text{PCLK}}{8 \times 2^{\text{BGSEL}}} \quad (13.5)$$

If the maximum number of bits in the Break field is exceeded, the internal counter will overflow, which results in a baudrate detection error. Therefore, an appropriate BGSEL value has to be selected for the required baudrate detection range.

The baud rate range defined by different BGSEL settings is shown in [Table 13-63](#).

Table 13-63 BGSEL Bit Field Definition for Different Input Frequencies

f_{PCLK}	BGSEL	Baud Rate Select for Detection $f_{pclk}/(2184 \cdot 2^{BGSEL})$ to $f_{pclk}/(72 \cdot 2^{BGSEL})$
20 MHz	00 _B	9.2 kHz to 277.8 kHz
	01 _B	4.6 kHz to 138.9 kHz
	10 _B	2.3 kHz to 69.4 kHz
	11 _B	1.1 kHz to 34.7 kHz

Each BGSEL setting supports a range of baud rate for detection. If the baud rate used is outside the defined range, the baud rate may not be detected correctly.

When $f_{pclk} = 20$ MHz, the baud rate range between 1.1 kHz to 277.8 kHz can be detected. The following examples serve as a guide to select BGSEL value:

- If the baud rate falls in the range of 1.1 kHz to 2.3 kHz, selected BGSEL value is “11_B”.
- If the baud rate falls in the range of 2.3 kHz to 4.6 kHz, selected BGSEL value is “10_B”.
- If the baud rate falls in the range of 4.6 kHz to 9.2 kHz, selected BGSEL value is “01_B”.
- If the baud rate falls in the range of 9.2 kHz to 277.8 kHz, selected BGSEL value is “00_B”. If the baud rate is 20 kHz, the possible values of BGSEL that can be selected are “00_B”, “01_B”, “10_B”, and “11_B”. However, it is advisable to select “00_B” for better detection accuracy.

Feature Discontinued

13.15.5.6 LIN Baud Rate Detection

The baud rate detection for LIN is shown in **Figure 13-69**, the Header LIN frame consists of the:

- SYN Break (13 bit times low)
- SYN byte (55_{H})
- Protected ID field

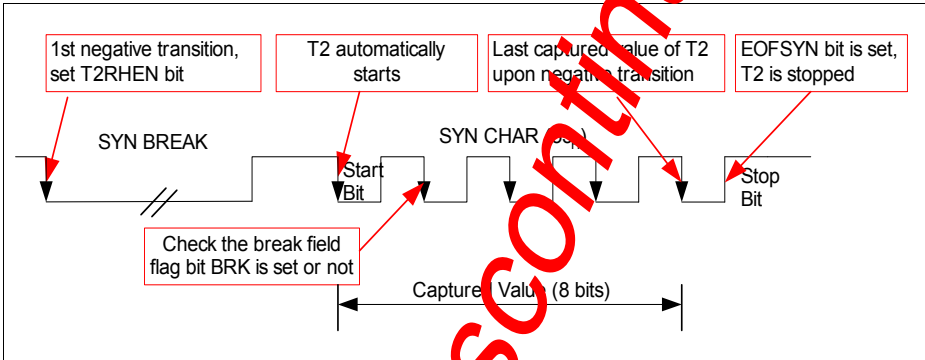


Figure 13-69 LIN Auto Baud Rate Detection

With the first falling edge:

- The Timer 2 External Start Enable bit (T2MOD.T2RHEN) is set. The falling edge at pin T2EX is selected by default for Timer 2 External Start (bit T2MOD.T2REGS is 0).

With the second falling edge:

- Start Timer 2 by the hardware.

With the third falling edge:

- Timer 2 captures the timing of 2 bits of SYN byte.
- Check the Break Field Flag bit LINST.BRK.

If the Break Field Flag LINST.BRK is set, software may continue to capture 4/6/8 bits of SYN byte. Finally, the End of SYN Byte Flag (LINST.EOFSYN) is set, Timer 2 is stopped. T2 Reload/Capture register (RC2H/L) is the time taken for 2/4/6/8 bits according to the implementation. Then the LIN routine calculates the actual baud rate, sets the PRE and BG values if the UART module uses the baud-rate generator for baud rate generation.

After the third falling edge, the software may discard the current operation and continue to detect the next header LIN frame if the following conditions were detected:

- The Break Field Flag LINST.BRK is not set, or
- The SYN Byte Error Flag LINST.ERRSYN is set, or
- The Break Field Flag LINST.BRK is set, but the End of SYN Byte Flag LINST.EOFSYN and the SYN Byte Error Flag LINST.ERRSYN are not set.

13.15.6 Register Description

Besides the SCON and SBUF registers, which can be accessed from both the standard (non-mapped) and mapped SFR area, the rest of the UART's SFRs are located in SCU page 2 of the standard area.

Table 13-64 lists the addresses of these SFRs.

Table 13-64 Register Map

Address	Register
98 _H	SCON
99 _H	SBUF
F2 _H	BCON
F3 _H	BGL
F4 _H	BGM
F5 _H	LNSI

Feature Discontinued

13.15.6.1 UART Registers

UART contains the two Special Function Registers (SFRs), SCON and SBUF. SCON is the control register and SBUF is the data register. On reset, both SCON and SBUF return 00_H. The serial port control and status register is the SFR SCON. This register contains not only the mode selection bits, but also the 9th data bit for transmit and receive (TB8 and RB8) and the serial port interrupt bits (TI and RI).

SBUF is the receive and transmit buffer of the serial interface. Writing to SBUF loads the transmit register and initiates transmission. This register is used for both transmit and receive data. Transmit data is written to this location and receive data is read from this location, but the two paths are independent.

Reading out SBUF accesses a physically separate receive register.

SBUF

Serial Data Buffer

(99_H)

Reset Value: 00_H

RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
VAL							
rwh							

Field	Bits	Type	Description
VAL	[7:0]	rwh	Serial Interface Buffer Register

SCON

Serial Channel Control Register

(98_H)

Reset Value: 00_H

RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
SM0	SM1	SM2	REN	TB8	RB8	TI	RI
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
RI	0	rwh	Receive Interrupt Flag This is set by hardware at the end of the 8th bit on mode 0, or at the half point of the stop bit in modes 1, 2, and 3. Must be cleared by software.

Field	Bits	Type	Description
TI	1	rwh	Transmit Interrupt Flag This is set by hardware at the end of the 8th bit in mode 0, or at the beginning of the stop bit in modes 1, 2, and 3. Must be cleared by software.
RB8	2	rwh	Serial Port Receiver Bit 8 In modes 2 and 3, this is the 8th data bit received. In mode 1, this is the stop bit received. In mode 0, this bit is not used.
TB8	3	rw	Serial Port Transmitter Bit 9 In modes 2 and 3, this is the 9th data bit sent.
REN	4	rw	Enable Receiver of Serial Port 0 _B Serial reception is disabled. 1 _B Serial reception is enabled.
SM2	5	rw	Enable Serial Port Multiprocessor Communication in Modes 2 and 3 In mode 2 or 3, if SM2 is set to 1, RI will not be activated if the received 9th data bit (RB8) is 0. In mode 1, if SM2 is set to 1, RI will not be activated if a valid stop bit (RB8) was not received. In mode 0, SM2 should be 0.
SM1	6	rw	Serial Port Operating Mode Selection See definition for SM0.
SM0	7	rw	Serial Port Operating Mode Selection [SM0, SM1] 0 _B Mode 0: 8-bit shift register, fixed baud rate ($f_{PCLK}/2$). 01 _B Mode 1: 8-bit UART, variable baud rate. 10 _B Mode 2: 9-bit UART, fixed baud rate ($f_{PCLK}/64$ or $f_{PCLK}/32$). 11 _B Mode 3: 9-bit UART, variable baud rate.

Feature Discontinued

13.15.6.2 Baud-rate Generator Control and Status Registers

PCON

Power Control Register

(87_H)

Reset Value: 00_H

RMAP: X, PAGE: X

7	6	5	4	3	2	1	0
SMOD		0		GF1	GF0	0	IDLE
rw		r		rw	rw	r	rw

Field	Bits	Type	Description
SMOD	7	rw	Double Baud Rate Enable 0 _B Do not double the baud rate of serial interface in modes 1, 2 and 3. 1 _B Double the baud rate of serial interface in mode 2.
0	1,[6:4]	r	Reserved Returns 0 on read; should be written with 0.

BCON

Baud Rate Control Register

(F2_H)

Reset Value: 00_H

RMAP: 0, PAGE: 2

7	6	5	4	3	2	1	0
BGSEL		0	BRDIS		BRPRE		R
rw		r	rw		rw		rw

Field	Bits	Type	Description
R	0	rw	Baud Rate Generator Run Control Bit 0 _B Baud-rate generator disabled. 1 _B Baud-rate generator enabled. <i>Note: BR_VALUE should only be written if R = 0.</i>

Field	Bits	Type	Description
BRPRE	[3:1]	rw	Prescaler Bit Selects the input clock for f_{DIV} which is derived from the peripheral clock. $000_B f_{DIV} = f_{PCLK}$ $001_B f_{DIV} = f_{PCLK}/2$ $010_B f_{DIV} = f_{PCLK}/4$ $011_B f_{DIV} = f_{PCLK}/8$ $100_B f_{DIV} = f_{PCLK}/16$ $101_B f_{DIV} = f_{PCLK}/32$ Others: reserved
BRDIS	4	rw	Baud Rate Detection Disable 0_B Break/Synch detection is enabled. 1_B Break/Synch detection is disabled.
BGSEL	[7:6]	rw	Baud Rate Selection for Detection For different values of BGSEL, the baud rate range for detection is defined by the following formula: $f_{pclk}/(218 \cdot 2^{BGSEL}) < \text{baud rate range} < f_{pclk}/(72 \cdot 2^{BGSEL})$ where $BGSEL = 00_B, 01_B, 10_B, 11_B$. See Table 13-63 for bit field BGSEL definition for different input frequencies.

LINST

LIN Status Register

 (F5_H)

 Reset Value: 00_H

RMAP: 0, PAGE: 2

7	6	5	4	3	2	1	0
0	SYNEN	ERRSYN	EOFSYN	BRK		0	
r	rw	rwh	rwh	rwh		r	

Field	Bits	Type	Description
BRK	3	rwh	Break Field Flag This bit is set by hardware and can only be cleared by software. 0_B Break Field is not detected. 1_B Break Field is detected.

Field	Bits	Type	Description
EOfSYN	4	rwh	End of SYN Byte Interrupt Flag This bit is set by hardware and can only be cleared by software. 0 _B End of SYN Byte is not detected. 1 _B End of SYN Byte is detected.
ERRSYN	5	rwh	SYN Byte Error Interrupt Flag This bit is set by hardware and can only be cleared by software. 0 _B Error is not detected in SYN Byte. 1 _B Error is detected in SYN Byte.
SYNEN	6	rw	End of SYN Byte and SYN Byte Error Interrupts Enable 0 _B End of SYN Byte and SYN Byte Error Interrupts are not enabled. 1 _B End of SYN Byte and SYN Byte Error Interrupts are enabled.
0	7, [2:0]	r	Reserved Returns 0 if read; should be written with 0.

13.15.6.3 Baud-rate Generator Time/Reload Registers

The low and high bytes of the baud rate timer/reload register BG contains the 11-bit reload value for the baud rate timer and the 5-bit fractional divider selection.

Reading the low byte of register BG returns the content of the lower three bits of the baud rate timer and the FD_SEL setting, while reading the high byte returns the content of the upper 8 bits of the baud rate timer.

Writing to register BG loads the baud rate timer with the reload and fractional divider values from the BG register, the first instruction cycle after BCON.R is set.

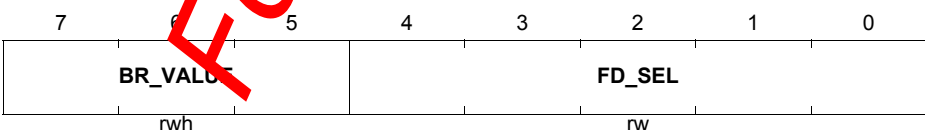
BG should only be written if R = 0.

BGL

Baud Rate Timer/Reload Register, Low Byte
(F3_H)

Reset Value: 00_H

RMAP: 0, PAGE: 2



Field	Bits	Type	Description
FD_SEL	[4:0]	rw	Fractional Divider Selection Selects the fractional divider to be $n/32$, where n is the value of FD_SEL and is in the range of 0 to 31. For example, writing 000 _H to FD_SEL selects the fractional divider to be $1/32$. <i>Note: Fractional divider has no effect if BR_VALUE = 000_H.</i>
BR_VALUE	[7:5]	rwh	Baud Rate Timer/Reload Value The lower three bits of the 11-bit Baud Rate Timer/Reload value. See also description in BGH register.

BGH
Baud Rate Timer/Reload Register, High Byte

 (F_H)

Reset Value: 00_H
RMAP: 0, PAGE: 2

7	6	5	3	2	1	0
BR_VALUE						
rwh						

Field	Bits	Type	Description
BR_VALUE	[7:0]	rwh	Baud Rate Timer/Reload Value The upper 8 bits of the 11-bit Baud Rate Timer/Reload value. When the 11-bit BR_VALUE is 000 _H , baud-rate timer is bypassed.

Feature Discontinued

13.16 High-Speed Synchronous Serial Interface

13.16.1 Overview

The High-Speed Synchronous Serial Interface (SSC) supports both full-duplex and half-duplex serial synchronous communication. The serial clock signal can be generated by the SSC itself (Master Mode) through its own 16-bit baud rate generator, or can be received from an external master (Slave Mode). Data width, shift direction, clock polarity, and phase are programmable. This allows communication with SPI-compatible devices using other synchronous serial interfaces. Transmission and reception of data is double-buffered.

Features

- Master and Slave Mode operation
 - Full-duplex or half-duplex operation
- Transmit and receive buffered
- Flexible data format
 - Programmable number of data bits: 7 to 8 bits
 - Programmable shift direction: Least Significant Bit (LSB) or Most Significant Bit (MSB) shift first
 - Programmable clock polarity: true low or high state for the shift clock
 - Programmable clock/data phase: data shift with leading or trailing edge of the shift clock
- Variable baud rate
- Compatible with Serial Peripheral Interface (SPI)
- Interrupt generation
 - On a transmitter empty condition
 - On a receiver full condition
 - On an error condition (receive, phase, baud rate, transmit error)

Figure 13-70 shows the block diagram of the SSC.

Feature Discontinued

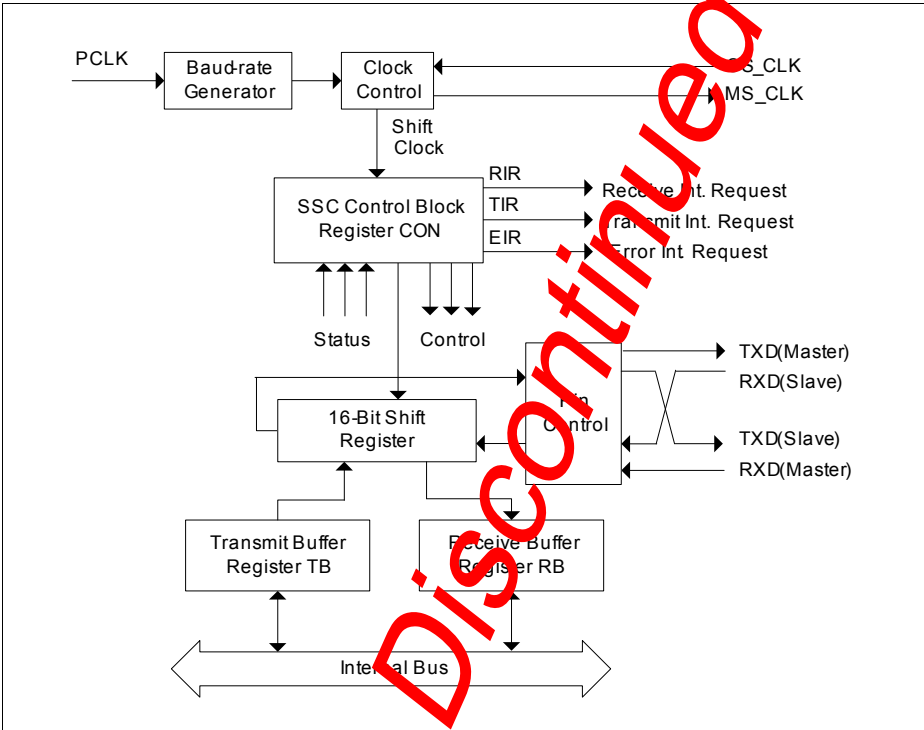


Figure 13-70 SSC Block Diagram

13.16.2 General Operation

The SSC supports full-duplex and half-duplex synchronous communication up to 10 MBaud (@ 20 MHz module clock). The serial clock signal can be generated by the SSC itself (Master Mode) or can be received from an external master (Slave Mode). Data width, shift direction, clock polarity, and phase are programmable. This allows communication with SPI-compatible devices. Transmission and reception of data is double-buffered. A 16-bit baud-rate generator provides the SSC with a separate serial clock signal.

The SSC can be configured in a very flexible way, so it can be used with other synchronous serial interfaces, can serve for master/slave or multimaster interconnections or can operate compatible with the popular SPI interface. Thus, the SSC can be used to communicate with shift registers (I/O expansion), peripherals (e.g. EEPROMs, etc.) or other controllers (networking). The SSC supports half-duplex and full-duplex communication. Data is transmitted or received on lines TXD and RXD,

normally connected with pins MTSR (Master Transmit/Slave Receive) and MRST (Master Receive/Slave Transmit). The clock signal is output via line MS_CLK (Master Serial Shift Clock) or input via line SS_CLK (Slave Serial Shift Clock). Both lines are normally connected to pin SCLK.

13.16.2.1 Operating Mode Selection

The operating mode of the serial channel SSC is controlled by its control register CON. This register serves two purposes:

- During programming (SSC disabled by CON.EN = 0), it provides access to a set of control bits
- During operation (SSC enabled by CON.EN = 1), it provides access to a set of status flags.

The shift register of the SSC is connected to both the transmit lines and the receive lines via the pin control logic. Transmission and reception of serial data are synchronized and take place at the same time, i.e. the same number of transmitted bits is also received. Transmit data is written into the Transmit Buffer (TB) and is moved to the shift register as soon as this is empty. An SSC master (CON.MS = 1) immediately begins transmitting, while an SSC slave (CON.MS = 0) will wait for an active shift clock. When the transfer starts, the busy flag CON.BSY is set and the Transmit Interrupt Request line TIR will be activated to indicate that register TB may be reloaded again. When the programmed number of bits (2 ... 8) has been transferred, the contents of the shift register are moved to the Receive Buffer RB and the Receive Interrupt Request line RIR will be activated. If no further transfers take place (TB is empty), CON.BSY will be cleared at the same time. Software should not modify CON.BSY, as this flag is hardware controlled. The level of the transmit line after the completion of the data transfer is dependent on the value of the first bit shifted in on the receive line.

Note: The SSC starts transmission and sets CON.BSY minimum two clock cycles after transmit data is written into TB. Therefore, it is not recommended to poll CON.BSY to indicate the start and end of a single transmission. Instead, interrupt service routine should be used if interrupts are enabled, or the interrupt flags IRCON1.TIR and IRCON1.RIR should be polled if interrupts are disabled.

Note: Only one SSC (etc) can be master at a given time.

The transfer of serial data (bits) can be programmed in many respects:

- The data width can be specified from 2 bits to 8 bits
- A transfer may start with either the LSB or the MSB
- The shift clock may be idle low or idle high
- The data bits may be shifted with the leading edge or the trailing edge of the shift clock signal
- The baud rate may be set from 152.59 Baud up to 10 MBaud (@ 20 MHz module clock)

- The shift clock can be generated (MS_CLK) or can be received (SS_CLK)

These features allow the adaptation of the SSC to a wide range of applications requiring serial data transfer.

The Data Width Selection supports the transfer of frames of any data length, from 2-bit “characters” up to 8-bit “characters”. Starting with the LSB (CON.HB = 0) allows communication with SSC devices in Synchronous Mode or with 8051 like serial interfaces for example. Starting with the MSB (CON.HB = 1) allows operation compatible with the SPI interface.

Regardless of the data width selected and whether the MSB or the LSB is transmitted first, the transfer data is always right-aligned in registers TR and RB, with the LSB of the transfer data in bit 0 of these registers. The data bits are rearranged for transfer by the internal shift register logic. The unselected bits of TR are ignored; the unselected bits of RB will not be valid and should be ignored by the receiver service routine.

The Clock Control allows the adaptation of transmit and receive behavior of the SSC to a variety of serial interfaces. A specific shift clock edge (rising or falling) is used to shift out transmit data, while the other shift clock edge is used to latch in receive data. Bit CON.PH selects the leading edge or the trailing edge for each function. Bit CON.PO selects the level of the shift clock line in the idle state. Thus, for an idle-high clock, the leading edge is a falling one, a 1-to-0 transition (see [Figure 13-71](#)).

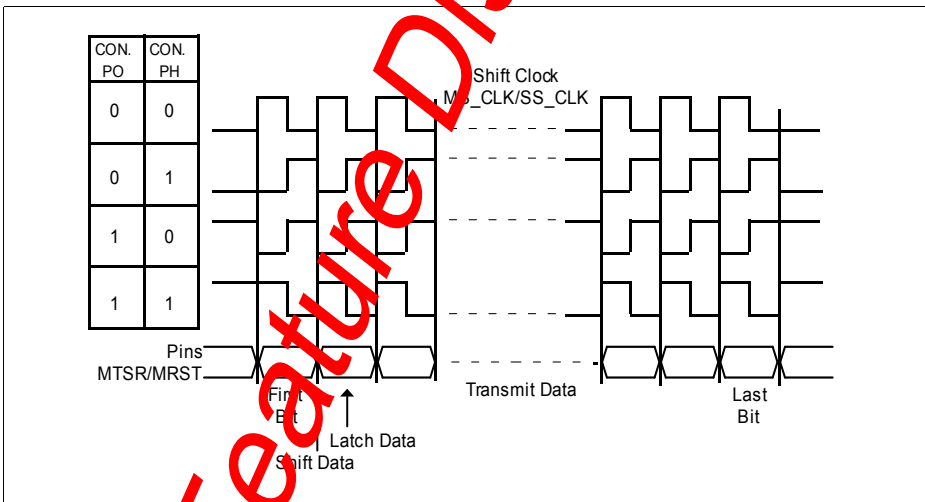


Figure 13-71 Serial Clock Phase and Polarity Options

13.16.2.2 Full-Duplex Operation

The various devices are connected through three lines. The definition of these lines is always determined by the master: the line connected to the master's data output line TXD is the transmit line; the receive line is connected to its data input line RXD; the shift clock line is either MS_CLK or SS_CLK. Only the device selected for master operation generates and outputs the shift clock on line MS_CLK. Since all slaves receive this clock, their pin SCLK must be switched to input mode. The output of the master's shift register is connected to the external transmit line, which in turn is connected to the slaves' shift register input. The output of the slaves' shift register is connected to the external receive line in order to enable the master to receive the data shifted out of the slave. The external connections are hard-wired, the function and direction of these pins is determined by the master or slave operation of the individual device.

Note: The shift direction shown in the figure applies for MSB-first operation as well as for LSB-first operation.

When initializing the devices in this configuration, one device must be selected for master operation while all other devices must be programmed for slave operation. Initialization includes the operating mode of the device's SSC and also the function of the respective port lines.

Feature Discontinued

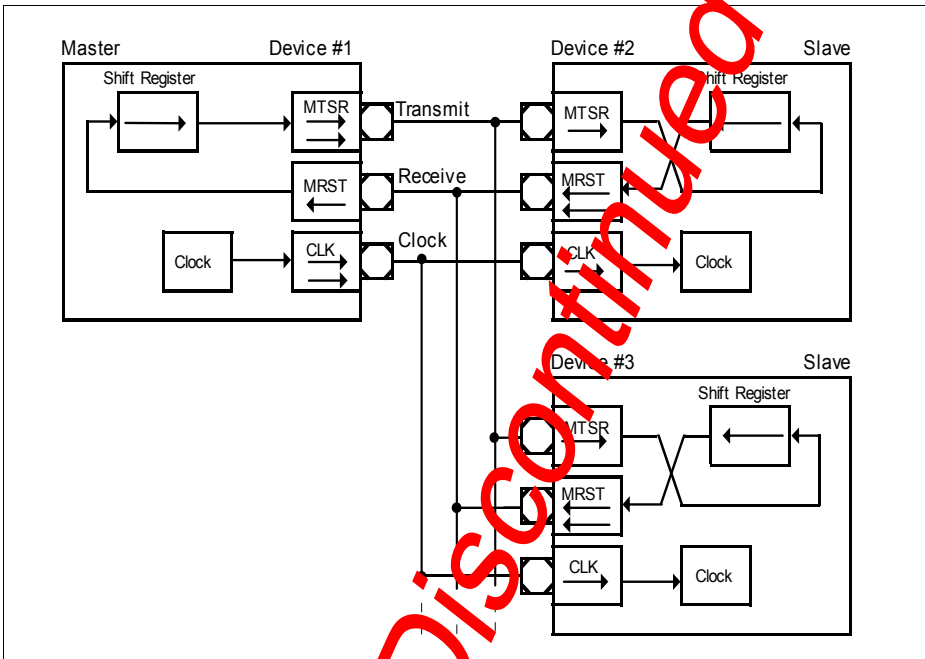


Figure 13-72 SSC Full-Duplex Configuration

The data output pins MRST of all slave devices are connected together onto the one receive line in the configuration shown in [Figure 13-72](#). During a transfer, each slave shifts out data from its shift register. There are two ways to avoid collisions on the receive line due to different slave data:

- Only one slave drives the line, i.e. enables the driver of its MRST pin. All the other slaves must have their MRST pins programmed as input so only one slave can put its data onto the master receive line. Only receiving data from the master is possible. The master selects the slave device from which it expects data either by separate select lines or by sending a special command to this slave. The selected slave then switches its MRST line to output until it gets a de-selection signal or command. This option is applicable only if pin has output driver disabling capability.
- The slaves use open-drain output on MRST. This forms a wired-AND connection. The receive line needs an external pull-up in this case. Corruption of the data on the receive line sent by the selected slave is avoided when all slaves not selected for transmission to the master only send ones (1s). Because this high level is not actively driven onto the line, but only held through the pull-up device, the selected slave can pull this line actively to a low-level when transmitting a zero bit. The master selects

the slave device from which it expects data either by separate select lines or by sending a special command to this slave.

After performing the necessary initialization of the SSC, the serial interfaces can be enabled. For a master device, the alternate clock line will now go to its programmed polarity. The alternate data line will go to either 0 or 1 until the first transfer starts. After a transfer, the alternate data line will always remain at the logic level of the last transmitted data bit.

When the serial interfaces are enabled, the master device can initiate the first data transfer by writing the transmit data into register TB. This value is copied into the shift register (assumed to be empty at this time), and the selected first bit of the transmit data will be placed onto the TXD line on the next clock from the baud-rate generator (transmission starts only if CON.EN = 1). Depending on the selected clock phase, a clock pulse will also be generated on the MS_CLK line. At the same time, with the opposite clock edge, the master latches and shifts in the data detected at its input line RXD. This “exchanges” the transmit data with the receive data. Because the clock line is connected to all slaves, their shift registers will be shifted synchronously with the master’s shift register — shifting out the data contained in the registers, and shifting in the data detected at the input line. After the preprogrammed number of clock pulses (via the data width selection), the data transmitted by the master is contained in all the slaves’ shift registers, while the master’s shift register holds the data of the selected slave. In the master and all slaves, the contents of the shift register are copied into the receive buffer RB and the receive interrupt line RIF is activated.

A slave device will immediately output the selected first bit (MSB or LSB of the transfer data) at line RXD when the contents of the transmit buffer are copied into the slave’s shift register. Bit CON.BSY is not set until the first clock edge at SS_CLK appears. The slave device will not wait for the next clock from the baud-rate generator, as the master does. The reason for this is that, depending on the selected clock phase, the first clock edge generated by the master may already be used to clock in the first data bit. Thus, the slave’s first data bit must already be valid at this time.

*Note: On the SSC, a transmission **and** a reception takes place at the same time, regardless of whether valid data has been transmitted or received.*

Note: The initialization of the CLK pin on the master requires some attention in order to avoid undesired clock transitions, which may disturb the other devices. Before the clock pin is switched to output mode, the clock output level will be selected in the control register CON and the alternate output be prepared via the related ALTSEL register, or the output latch must be loaded with the clock idle level.

13.16.2.3 Half-Duplex Operation

In a Half-Duplex Mode, only one data line is necessary for both receiving **and** transmitting of data. There are two port configuration options for Half-Duplex Mode. The first option uses all the three pins but with the data exchange line connected to both the MTSR and MRST pins of each device; the shift clock line is connected to the SCLK pin. The second option uses two pins and requires the MTSR and MRST lines to be selected for the same GPIO pin, thus establishing also only a single data exchange line. The shift clock line is connected to the SCLK pin as before.

The master device controls the data transfer by generating the shift clock, while the slave devices receive it. Due to the fact that all transmit and receive pins are connected to the one data exchange line, serial data may be moved between arbitrary stations.

Similar to Full-Duplex Mode, there are two ways to avoid collisions on the data exchange line:

- Only the transmitting device may enable its transmit pin driver. This option is applicable only if pin has output driver disabling capability.
- The non-transmitting devices use open drain output and send only ones.

Because the data inputs and outputs are connected together, a transmitting device will clock in its own data at the input pin (MRST for a master device, MTSR for a slave). By this method, any corruptions on the common data exchange line are detected if the received data is not equal to the transmitted data.

Feature Discontinued

The Half-Duplex Mode port configuration using three pins is shown in [Figure 13-73](#).

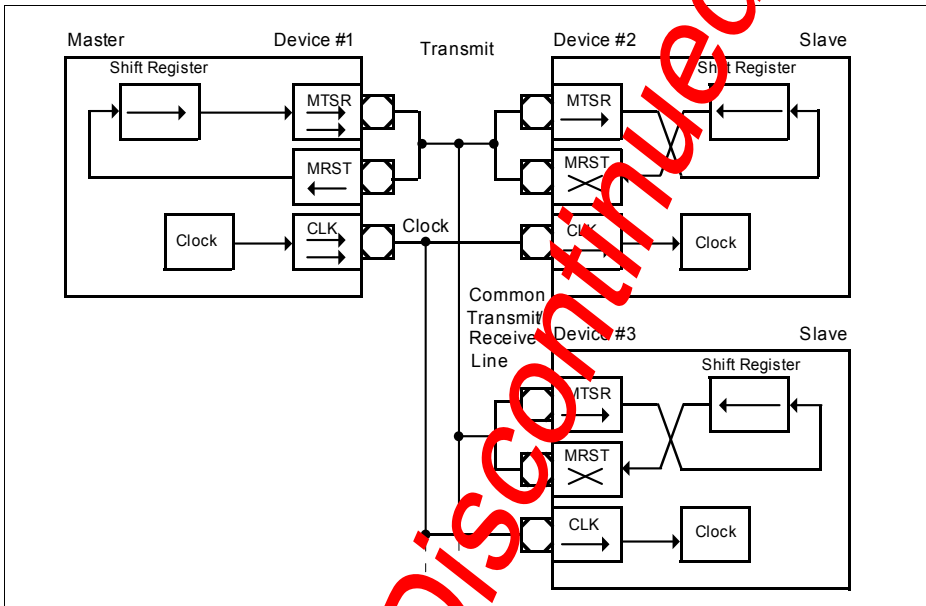


Figure 13-73 SSC Half-Duplex Configuration

13.16.2.4 Continuous Transfers

When the transmit interrupt request flag is set, it indicates that the transmit buffer TB is empty and ready to be loaded with the next transmit data. If TB has been reloaded by the time the current transmission is finished, the data is immediately transferred to the shift register and the next transmission will start without any additional delay. On the data line, there is no gap between the two successive frames. For example, two byte transfers would look the same as one word transfer. This feature can be used to interface with devices that can operate with or require more than 8 data bits per transfer. It is just a matter of software, how long a total data frame length can be. This option can also be used to interface to byte-wide and word-wide devices on the same serial bus, for instance.

Note: Of course, this can happen only in multiples of the selected basic data width, because it would require disabling/enabling of the SSC to reprogram the basic data width on-the-fly.

13.16.2.5 Baud Rate Generation

The serial channel SSC has its own dedicated 16-bit baud-rate generator with 16-bit reload capability, allowing baud rate generation independent of the timers. **Figure 13-74** shows the baud-rate generator.

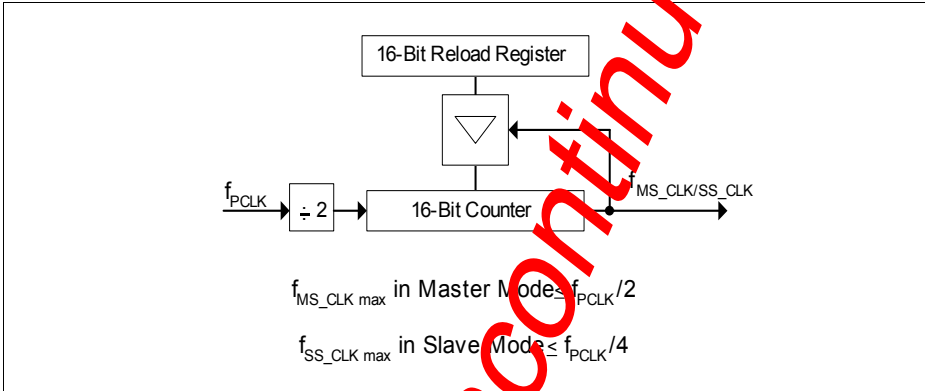


Figure 13-74 SSC Baud-rate Generator

The baud-rate generator is clocked with the module clock f_{hw_clk} . The timer counts downwards. Register BR is the dual-function Baud-rate Generator/Reload register. Reading BR, while the SSC is enabled, returns the contents of the timer. Reading BR, while the SSC is disabled, returns the programmed reload value. In this mode, the desired reload value can be written to BR.

Note: Never write to BR while the SSC is enabled.

The formulas below calculate either the resulting baud rate for a given reload value, or the required reload value for a given baud rate:

$$\text{Baud rate} = \frac{f_{hw_clk}}{2 \cdot (\langle BR \rangle + 1)} \qquad BR = \frac{f_{hw_clk}}{2 \cdot \text{Baud rate}} - 1 \qquad (13.6)$$

$\langle BR \rangle$ represents the contents of the reload register, taken as an unsigned 16-bit integer, while baud rate is equal to f_{MS_CLK/SS_CLK} as shown in **Figure 13-74**.

The maximum baud rate that can be achieved when using a module clock of 20 MHz is 10 MBaud in Master Mode (with $\langle BR \rangle = 0000_H$) or 5 MBaud in Slave Mode (with $\langle BR \rangle = 0001_H$).

Table 13-65 lists some possible baud rates together with the required reload values and the resulting bit time, assuming a module clock of 20 MHz.

Table 13-65 Typical Baud Rates of the SSC ($f_{hw_clk} = 20\text{ MHz}$)

Reload Value	Baud Rate ($= f_{MS_CLK/SS_CLK}$)	Deviation
0000 _H	10 MBaud (only in Master Mode)	0.0%
0001 _H	5 MBaud	0.0%
0004 _H	2 MBaud	0.0%
000A _H	1 MBaud	0.0%
0013 _H	500 kBaud	0.0%
0063 _H	100 kBaud	0.0%
FFFF _H	152.59 Baud	0.0%

13.16.2.6 Error Detection Mechanisms

The SSC is able to detect four different error conditions. Receive Error and Phase Error are detected in all modes; Transmit Error and Baud Rate Error apply only to Slave Mode. When an error is detected, the respective error flag is/can be set and an error interrupt request will be generated by activating the FIR line (see [Figure 13-75](#)) if enabled. The error interrupt handler may then check the error flags to determine the cause of the error interrupt. The error flags are not reset automatically but rather must be cleared by software after servicing. This allows servicing of some error conditions via interrupt, while the others may be polled by software.

Note: The error interrupt handler must clear the associated (enabled) error flag(s) to prevent repeated interrupt requests.

Feature Discontinued

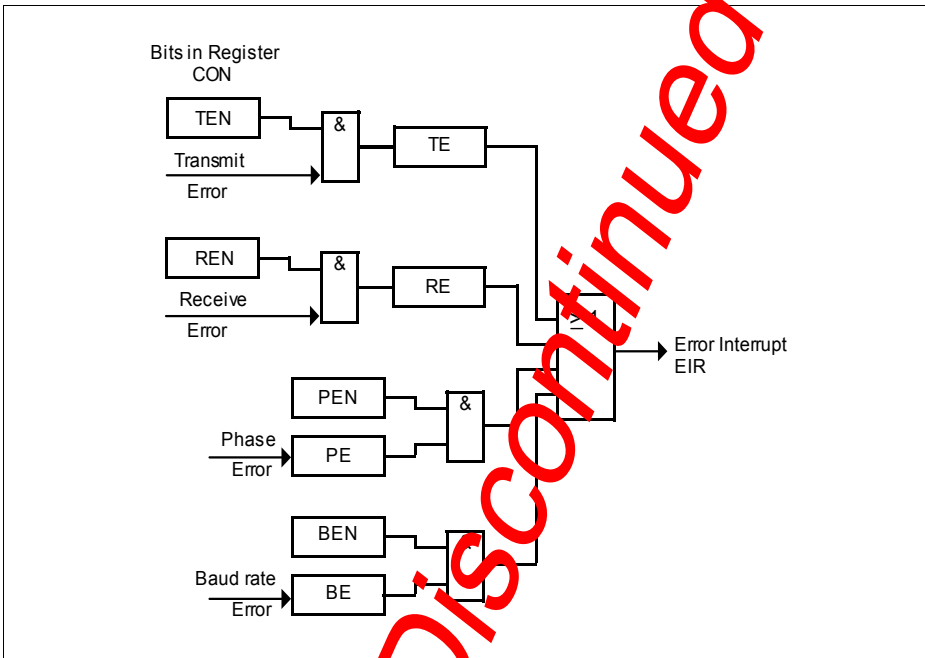


Figure 13-75 SSC Error Interrupt Control

A **Receive Error** (Master or Slave Mode) is detected when a new data frame is completely received but the previous data was not read out of the receive buffer register RB. This condition sets the error flag CON.RE and the error interrupt request line EIR, when enabled via CON.REN. The old data in the receive buffer RB will be overwritten with the new value and is irretrievably lost.

A **Phase Error** (Master or Slave Mode) is detected when the incoming data at pin MRST (Master Mode) or MTSR (Slave Mode), sampled with the same frequency as the module clock, changes between one cycle before and two cycles after the latching edge of the shift clock signal SCLK. This condition sets the error flag CON.PE and, when enabled via CON.PEN, the error interrupt request line EIR.

Note: When receiving and transmitting data in parallel, phase errors occur if the baud rate is configured to $\frac{f_{hw_clk}}{2}$.

A **Baud Rate Error** (Slave Mode) is detected when the incoming clock signal deviates from the programmed baud rate by more than 100%, i.e. it is either more than double or less than half the expected baud rate. This condition sets the error flag CON.BE and, when enabled via CON.BEN, the error interrupt request line EIR. Using this error detection capability requires that the slave's baud-rate generator is programmed to the

same baud rate as the master device. This feature detects false, additional, or missing pulses on the clock line (within a certain frame).

Note: If this error condition occurs and bit CON.AREN = 1, an automatic reset of the SSC will be performed in case of this error. This is done to reinitialize the SSC if too few or too many clock pulses have been detected.

Note: This error can occur after any transfer if the communication is stopped. This is the case due to the fact that the SSC module supports back-to-back transfers for multiple transfers. In order to handle this, the baud rate detector expects after a finished transfer immediately a next clock cycle for a new transfer.

A **Transmit Error** (Slave Mode) is detected when a transfer was initiated by the master (SS_CLK gets active) but the transmit buffer TB of the slave was not updated since the last transfer. This condition sets the error flag CON.TE and the error interrupt request line EIR, when enabled via CON.TEN. If a transfer starts while the transmit buffer is not updated, the slave will shift out the 'old' content of the shift register, which normally is the data received during the last transfer. This may lead to corruption of the data on the transmit/receive line in half-duplex mode (open drain configuration) if this slave is not selected for transmission. This mode requires that slaves not selected for transmission only shift out ones; that is, their transmit buffers must be loaded with 'FFFF_H' prior to any transfer.

Note: In order to avoid possible conflicts or misinterpretations, it is recommended to always load the slave's transmit buffer prior to any transfer.

The cause of an error interrupt request (receive, phase, baud rate, transmit error) can be identified by the error status flags in control register CON.

Note: In contrast to the error interrupt request line EIR, the error status flags CON.TE, CON.RE, CON.PE, and CON.BE, are not reset automatically upon entry into the error interrupt service routine, but must be cleared by software.

Feature Discontinued

13.16.3 Interrupts

The three SSC interrupts can be separately enabled or disabled by setting or clearing their corresponding enable bits in SFR SCU_MODIEN.

For a detailed description of the various interrupts see [Section 13.16.2](#). An overview is given in [Table 13-66](#).

Table 13-66 SSC Interrupt Sources

Interrupt	Signal	Description
Transmission starts	TIR	Indicates that the transmit buffer can be reloaded with new data.
Transmission ends	RIR	The configured number of bits have been transmitted and shifted to the receive buffer.
Receive Error	EIR	This interrupt occurs if a new data frame is completely received and the last data in the receive buffer was not read.
Phase Error	EIR	This interrupt is generated if the incoming data changes between one cycle before and two cycles after the latching edge of the shift clock signal SCLK.
Baud Rate Error (Slave Mode only)	EIR	This interrupt is generated when the incoming clock signal deviates from the programmed baud rate by more than 100%.
Transmit Error (Slave Mode only)	EIR	This interrupt is generated when TB was not updated since the last transfer if a transfer is initiated by a master.

Feature Discontinued

13.16.4 Register Description

The SSC Special Function Registers are accessed from the standard (non-mapped) SFR area. The addresses of the SFRs are listed in [Table 13-67](#).

Table 13-67 Register Map

Address	Register
AA _H	CONL
AB _H	CONH
AC _H	TBL
AD _H	RBL
AE _H	BRL
AF _H	BRH

Feature Discontinued

13.16.4.1 Configuration Register

The operating mode of the serial channel SSC is controlled by the control register CON. This register contains control bits for mode and error check selection, and status flags for error identification. Depending on bit EN, either control functions or status flags and master/slave control are enabled.

CON.EN = 0: Programming Mode

SSC_CONL

Control Register Low [Programming Mode]
(AA_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
LB	PO	PH	HB			BM	
rw	rw	rw	rw			rw	

Field	Bit	Type	Description
BM	[3:0]	rw	Data Width Selection 0000 _B reserved. Do not use this combination. 0001 _B - 0111 _B Transfer Data Width is 2 ... 8 bits (<BM>+1). <i>Note: The most significant bit of BM field is always fixed at 0. Thus the transfer and receive data width is restricted at maximum 8 bits. Transfer Data Width is 2...8 bits (<BM>+1).</i>
HB	4	rw	Leading Control 0 _B Transmit/Receive LSB First. 1 _B Transmit/Receive MSB First.
PH	5	rw	Clock Phase Control 0 _B Shift transmit data on the leading clock edge, latch on trailing edge. 1 _B Latch receive data on leading clock edge, shift on trailing edge.
PO	6	rw	Clock Polarity Control 0 _B Idle clock line is low, leading clock edge is low-to-high transition. 1 _B Idle clock line is high, leading clock edge is high-to-low transition.

Field	Bit	Type	Description
LB	7	rw	Loop Back Control 0 _B Normal output. 1 _B Receive input is connected with transmit output (half-duplex mode).

SSC_CONH
Control Register High [Programming Mode]
 (AB_H)

Reset Value: 00_H
RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
EN	MS	0	AREN	BEN	PEN	REN	TEN
rw	rw	r	rw	rw	rw	rw	rw

Field	Bit	Type	Description
TEN	0	rw	Transmit Error Enable 0 _B Ignore transmit errors. 1 _B Check transmit errors.
REN	1	rw	Receive Error Enable 0 _B Ignore receive errors. 1 _B Check receive errors.
PEN	2	rw	Phase Error Enable 0 _B Ignore phase errors. 1 _B Check phase errors.
BEN	3	rw	Baud Rate Error Enable 0 _B Ignore baud rate errors. 1 _B Check baud rate errors.
AREN	4	rw	Automatic Reset Enable 0 _B No additional action upon a baud rate error. 1 _B The SSC is automatically reset upon a baud rate error.
MS	6	rw	Master Select 0 _B Slave Mode. Operate on shift clock received via SCLK. 1 _B Master Mode. Generate shift clock and output it via SCLK.

Field	Bit	Type	Description
EN	7	rw	Enable Bit = 0 Transmission and reception disabled. Access to control bits.
0	5	r	Reserved Returns 0 if read; should be written with 0.

Feature Discontinued

CON.EN = 1: Operating Mode

SSC_CONL

Control Register Low [Operating Mode]

(AA_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
0				BC			
r				rh			

Field	Bit	Type	Description
BC	[3:0]	rh	Bit Count Field Shift counter is updated with every shift bit. <i>Note: This bit field is not to be written to.</i>
0	[7:4]	r	Reserved Returns 0 if read; should be written with 0.

SSC_CONH

Control Register High [Operating Mode]

(AB_H)

Reset Value: 00_H

RMAP: 0, PAGE: X

7	6	5	4	3	2	1	0
EN	MS	0	BSY	BE	PE	RE	TE
rw	rw	r	rh	rwh	rwh	rwh	rwh

Field	Bit	Type	Description
TE	0	rwh	Transmit Error Flag 0 _B No error. 1 _B Transfer starts with the slave's transmit buffer not being updated.
RE	1	rwh	Receive Error Flag 0 _B No error. 1 _B Reception completed before the receive buffer was read.

Field	Bit	Type	Description
PE	2	rwh	Phase Error Flag 0 _B No error. 1 _B Received data changes around sampling clock edge.
BE	3	rwh	Baud Rate Error Flag 0 _B No error. 1 _B More than factor 2 or 0.5 between slave's actual and expected baud rate.
BSY	4	rh	Busy Flag Set while a transfer is in progress. <i>Note: This bit is not to be written to.</i>
MS	6	rw	Master Select Bit 0 _B Slave Mode. Operate on shift clock received via SCLK. 1 _B Master Mode. Generate shift clock and output it via SCLK.
EN	7	rw	Enable Bit Transmission and reception enabled. Access to status flags and M/S control.
0	5	r	Reserved Returns 0 if read; should be written with 0.

Note: The target of an access to CON (control bits or flags) is determined by the state of CON.EN prior to the access: that is, writing C057_H to CON in programming mode (CON.EN = 0) will initialize the SSC (CON.EN was 0) and then turn it on (CON.EN = 1). When writing to CON, ensure that reserved locations receive zeros.

13.16.4.2 Baud Rate Timer Reload Register

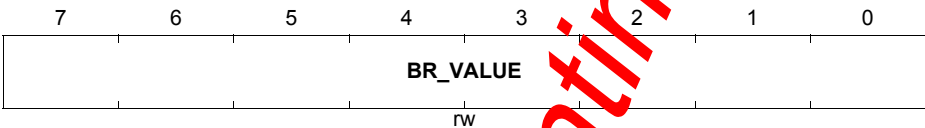
The SSC baud rate timer reload register BR contains the 16-bit reload value for the baud rate timer.

SSC_BRL

Baud Rate Timer Reload Register Low (AE_H)

Reset Value: 00_H

RMAP: 0, PAGE: X



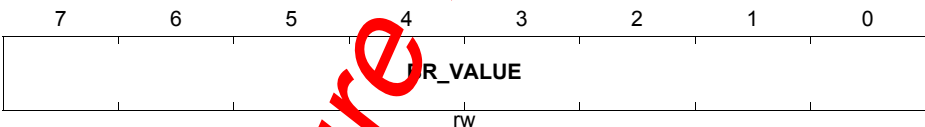
Field	Bit	Type	Description
BR_VALUE	[7:0]	rw	Baud Rate Timer/Reload Register Value Reading BR returns the 16-bit contents of the baud rate timer. Writing BR loads the baud rate timer reload register with BR_VALUE.

SSC_BRH

Baud Rate Timer Reload Register High (AF_H)

Reset Value: 00_H

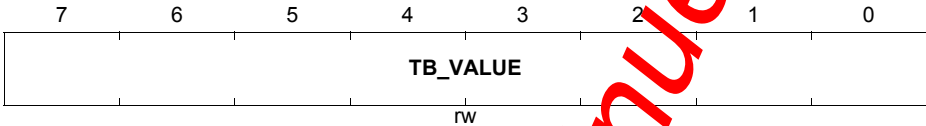
RMAP: 0, PAGE: X



Field	Bit	Type	Description
BR_VALUE	[7:0]	rw	Baud Rate Timer/Reload Register Value Reading BR returns the 16-bit contents of the baud rate timer. Writing BR loads the baud rate timer reload register with BR_VALUE.

13.16.4.3 Transmitter Buffer Register

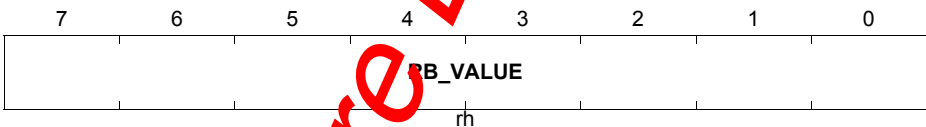
The SSC transmitter buffer register TB contains the transmit data value.

SSC_TBL
Transmitter Buffer Register Low
(AC_H)
Reset Value: 00_H
RMAP: 0, PAGE: X


Field	Bit	Type	Description
TB_VALUE	[7:0]	rw	Transmit Data Register Value TB_VALUE is the data value to be transmitted. Unselected bits of TB are ignored during transmission.

13.16.4.4 Receiver Buffer Register

The SSC receiver buffer register RB contains the receive data value.

SSC_RBL
Receiver Buffer Register Low
(AD_H)
Reset Value: 00_H
RMAP: 0, PAGE: X


Field	Bit	Type	Description
RB_VALUE	[7:0]	rh	Receive Data Register Value RB contains the received data value RB_VALUE. Unselected bits of RB will be not valid and should be ignored.

13.17 Wake-up CAN (WCAN) Filter

This chapter describes the WCAN controller of the SCR module. It contains the following sections:

- Overview on the Wake-up Capabilities (see [Page 13-364](#))
- Functional description of the WCAN Kernel (see [Page 13-366](#))
- WCAN Kernel register description (see [Page 13-367](#))
- Functional description on the CAN based Kernel (see [Page 13-390](#))
- SCR implementation specific details and registers of the WCAN (port connections and control, interrupt control, address decoding, clock control (see [Page 13-405](#)))

Note: The WCAN kernel register names described in this chapter will be referenced in the Standby Domain Internal Target Specification by the module name prefix "WCAN_".

Feature Discontinued

13.17.1 Wake-up CAN Filter Implementation

13.17.1.1 Overview

This section describes the 11898-6 datalink layer controller based on the well known MultiCAN. The Wakable CAN module contains of 1 CAN node, representing the receive part of the communication interface. Furthermore, the the Wake-up CAN filter functionality and interfaces in TC26x B-Step module.

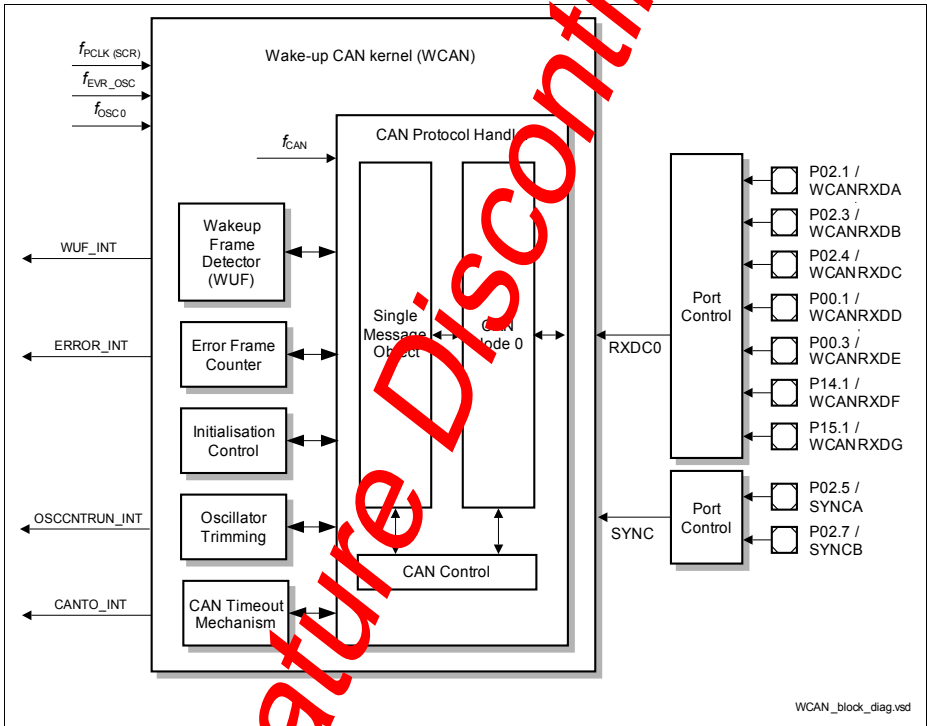


Figure 13-76 Wake-up CAN Filter Interfaces

13.17.1.2 WCAN Filter Features

There are three types of CAN frames:

- Wake-up CAN functionality in conformance to ISO 11898-
- 1 Wake-up CAN node mapped to CAN 0
- Guaranteed data transfer rate up to 500kBaud, baudrates higher are dependent on customer settings. Default settings for 125k, 250k and 500k are according to the given propagation delays of German OEMs.
- 1 Wake-up message object mapped to the wake-up node
 - Set up to handle frames with 11-bit and/or 29-bit identifiers
 - Programmable acceptance mask register for filtering
 - Programmable wake-up message ID, DLC and Data
- High Precision Oscillator trimming possible either via external pin or external crystal
- Wake-up CAN message Time-out feature
- Error counter to debounce CAN protocol and wake-up function errors
- Interrupt generation on
 - Successful Wake-up message frame reception
 - Completion of an Oscillator trim period.
 - Error recognition
 - Wake-up CAN timeout

Feature Discontinued

13.17.1.3 Introduction

Feature Overview

Reception and evaluation of CAN frames is handled in accordance to CAN specification ISO 11898-6 data link layer. The received messages will be compared against the programmed wake-up pattern and in case of pattern match a wake-up will be issued. In case of an erroneous understanding or errors on the bus, the receive error counter can also reach a wake-up threshold. The node will be purely listening to the bus and will not communicate to the outside.

The CAN node is having a single message object. In this message object, the identifier including a bitwise mask, the corresponding DLC, as well as a data mask is included. Only (outside the erroneous case), if all criteria match, a wake-up will be initiated.

The bit timings for the CAN nodes are derived from the peripheral clock (f_{CAN}) and are programmable up to a data rate of 1 MBaud, nonetheless guaranteed wake-up is only given for special configurations according to the propagation delay defined by the German OEMs. The ports are overlapping to the normal CAN nodes.

Features

- Compliant to ISO 11898-6.
- Receive operation according to ISO 11898-6.
- Dedicated control registers for the CAN node.
- Data transfer rates are guaranteed until 500kBaud, in case of higher baudrates more wake-up frames might be needed.
- Transmitter tolerances up to 0.4% are calculated and guaranteed. Higher transmitter tolerances are not pretested.
 - The message object can be configured to accept only standard or only extended frames or to accept both standard and extended frames.
 - Four interrupt sources do exist, oscillator trimming, wake-up frame found, $t_{SILENCE}$ expired and error overview.

Feature Discontinued

13.17.1.4 WCAN Module Control Registers

Wake-up CAN Configuration Register

CFG

Wake-up CAN Configuration Register (B0_H)

Reset Value: 00_H

RMAP: 0, PAGE: 0

7	6	5	4	3	2	1	0
OSCCAL	TRIMEN		CANTOMASK	CCE	ELVWKEN	CFGVAL	INITREQ
rw	rw		rw	rw	rw	rwh	rw

Field	Bits	Type	Description
INITREQ	0	rw	<p>WCAN initialization</p> <p>This bit initializes the CAN protocol handler and signals if the wake-up frame detection is running with the INITACK bit.</p> <p>0_B Disable WCAN and cannot receive CAN frames.</p> <p>1_B Enable WCAN and can receive CAN frames.</p>
CFG_VAL	1	rwh	<p>WCAN Configuration valid</p> <p>This bit reflects the status of a configuration valid check.</p> <p>0_B Configuration is not valid</p> <p>1_B WCAN configuration valid, written to enable wake-up functionality</p>

Feature Discontinued

Field	Bits	Type	Description
SELWK_EN	2	rw	Selective Wake-up Enable 0 _B Selective wake disabled 1 _B Selective wake enabled The Selective Wake-up Enable serves several functions when enabled: <ol style="list-style-type: none"> 1. Allows Wake-up source WUF to generate WUF_INT (See Figure 13-77). 2. Allows error counter generator ERROR_INT (See Figure 13-83) indicated by INTESTAT0.ECOPL. 3. Triggers ERROR_INT and wake-up if there are any changes to the message object registers, indicated by INTESTAT0.MOCR 4. If a SYSERR occurs while SELWK_EN is enabled, then the Selective Wake-up function shall be disabled, indicated by INTESTAT0.ERSEL.
CCE	3	rw	Configuration Change Enable 0 _B The Bit Timing Register may only be read. All attempts to modify them are ignored. 1 _B The Bit Timing Register may be read and written.
CANTOMASK	4	rw	CAN Time Out Masking This bit controls the interrupt masking for CANTO_INT interrupt. <ol style="list-style-type: none"> 0 CAN Time out is masked - no interrupt is triggered 1 CAN Time out is signaled as an interrupt
TRIMEN	[6:5]	rw	(Un)locking mechanism of oscillator recalibration 00 _B locked 01 _B locked 10 _B locked 11 _B unlocked TRIMEN unlocks the oscillation calibration mode. Only the bit combination '11' is the valid unlock.

Feature Discontinued

Field	Bits	Type	Description
OSCCAL	7	rw	Oscillator Calibration Mode This bit controls the Oscillator trimming (see Section 13.17.1.6) 0 _B Oscillator Calibration is disabled. 1 _B Oscillator Calibration is enabled. <i>Note: Oscillator Calibration is disabled for the standby domain.</i>
0	3	r	Reserved Read as 0 after reset; value last written is read back; should be written with 0.

Wake-up CAN Interrupt Mask and Reference Select Register

INTMRSLT

Wake-up CAN Interrupt Mask and Reference Select Register(B1_H) Reset Value:

00_H

RMAP: 0, PAGE: 0

7	6	5	4	3	2	1	0
0			OSCMASK	WUFMASK	ERRMASK	GATESEL	
r			rw	rw	rw	rw	

Field	Bits	Type	Description
GATESEL	0	rw	Reference Clock selection for gating This bit selects the different reference (i.e external gating SYNC pin or FOSC0) for trimming of the internal 100MHz RC-Oscillator. 0 _B External gating signal via SYNC pin 1 _B XTAL Osc 0 Clock source selected

Field	Bits	Type	Description
ERRMASK	1	rw	SYSERR Masking This bit controls the interrupt masking for SYSERR interrupt. 0 _B SYSERR is masked - no interrupt is triggered 1 _B SYSERR is signaled as an interrupt
WUFMASK	2	rw	Wake-up Frame Interrupt Masking This bit controls the interrupt masking for WUF_INT interrupt. 0 _B WUF_INT is masked - no interrupt is triggered 1 _B WUF_INT is signaled as an interrupt
OSCMASK	3	rw	Oscillator Run Interrupt Masking This bit controls the interrupt masking for OSCNTRRUN_INT interrupt. 0 _B OSCNTRUN_INT is masked - no interrupt is triggered 1 _B OSCNTRUN_INT is signaled as an interrupt
0	[7:2]	r	Reserved Read as 0 after reset; value last written is read back; should be written with 0.

Wake-up CAN Reference Clock Register 0

REFCLK0

Wake-up CAN Reference Clock Register 0(B2_H)

Reset Value: 00_H

RMAP: 0, PAGE: 0



Feature Discontinued

Field	Bits	Type	Description
XTALPERIOD0	[7:0]	rw	XTAL Oscillator period This bit field specifies the lower LSB in the 16-bit period0/1 of time (in terms of period) for FOSC0 that will be used to compare with FEVR_OSC. i.e PERIOD0/1 allows a value from 0 to 65535, say when a value of 3 is chosen, 3 periods of FOSC0 time is used to compare with FEVR_OSC and vice versa.

Wake-up CAN Reference Clock Register 1

REFCLK1

Wake-up CAN Reference Clock Register 1(B3...)

Reset Value: 00_H

RMAP: 0, PAGE: 0



Field	Bits	Type	Description
XTALPERIOD1	[7:0]	rw	XTAL Oscillator period This bit field specifies the upper MSB in the 16-bit period0/1 of time (in terms of period) for FOSC0 that will be used to compare with FEVR_OSC. i.e PERIOD0/1 allows a value from 0 to 65535, say when a value of 3 is chosen, 3 periods of FOSC0 time is used to compare with FEVR_OSC and vice versa.

Feature Discontinued

Wake-up CAN Interrupt and Event Status Register 0
INTESTAT0
Wake-up CAN Interrupt and Event Status Register 0(B0_H) **Reset Value: 01_H**
RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
0	ERSEL	MORC	ECOFL	CANTO	SYSERR	OSCCNTR UN	INITACK
r	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
INITACK	0	rh	CAN initialisation acknowledge This bit indicates if the WCAN is enabled and can receive CAN frames. 0 _B WCAN is enabled and can receive CAN frames when CFG_INITREQ is set to "1", enabled. 1 _B WCAN is disabled and cannot receive CAN frames.
OSCCNTRUN	1	rh	Oscillator counter run status 0 _B Oscillator counter has stopped 1 _B Oscillator counter is running
SYSERR	2	rh	CAN Wakeup System Error The SYSERR indicates a configuration error, frame detect error, internal failure conditions and an error counter overflow condition. 0 _B Selective Wake Mode is possible 1 _B System Error detected, WCAN enabling not possible
CANTO	3	rh	CAN Time Out Detection This bit captures, that there was an inactive CAN bus on RXDCO pin within tSILENCE of 750ms. See Section 13.17.1.7 . A software write to INTESTATCLR0.CANTOCLR is required to reset this bit. 0 _B Normal operation 1 _B CAN Time Out detected

Field	Bits	Type	Description
ECOFL	4	rh	Error Counter Overflow 0 _B Error Counter have not overflowed (<31) 1 _B Error Counter overflowed (>31)
MORC	5	rh	Message Object Register Changed 0 _B Message object registers not modified after SELWK_EN is set. 1 _B Message object registers was modified after SELWK_EN is set.
ERSEL	6	rh	Error Select 0 _B SYSERR is not set while SELWK_EN is enabled. 1 _B SYSERR is set while SELWK_EN is enabled.
0	7	r	Reserved Read as 0 after reset; value last written is read back; should be written with 0.

Feature Discontinued

Wake-up CAN Interrupt and Event Status Register 1
INTESTAT1
Wake-up CAN Interrupt and Event Status Register 1(B1_H) **Reset Value: 00_H**
RMAP: 0, PAGE: 1

	7	6	5	4	3	2	1	0
	LEC			0	CANSIL	SWKSET	SYNC	WUF
	rh			r	rh	r	rh	rh

Field	Bits	Type	Description
WUF	0	rh	Wake-up Frame Detection (acc. ISO 11898-6) This bit shows the status if a Wake-up Frame have been detected. 0 _B No Wake-up Frame detected 1 _B Wake-up Frame detected
SYNC	1	rh	Synchronization (at least one CAN frame without fail must have been received) This bit is set on valid frame is detected and is updated after every received frame. 0 _B WCAN function not working or not synchronous to CAN bus 1 _B Valid CAN frame received, WCAN function is synchronous to CAN bus
SWKSET	2	rh	Selective Wake Activity 0 _B Selective Wake is not active 1 _B Selective Wake is activated
CANSIL	3	rh	CAN Bus Silent Time during Selective Wake operation This bit is set if there is no activity on RXDC0 pin on the CAN bus after tSILENCE of 750ms expired. If any activity is detected on the bus, this bit is reset automatically. 0 _B tSILENCE not exceeded. 1 _B Set if tSILENCE is exceeded. tSILENCE is fixed at 750ms

Feature Discontinued

Field	Bits	Type	Description
LEC	[7:5]	rh	Last Error Code This bit field indicates the type of the last (most recent) CAN error. The encoding of this bit field is described in Table 13-68 .
0	4	r	Reserved Read as 0 after reset; value last written is read back; should be written with 0.

Encoding of the LEC Bit Field

Table 13-68 Encoding of the LEC Bit Field

LEC Value	Signification
000 _B	No Error: No error was detected for the last (most recent) message on the CAN bus.
001 _B	Stuff Error: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.
010 _B	Form Error: A fixed format part of a received frame has the wrong format.
011 _B	Ack Error: The transmitted message was not acknowledged by another node.
100 _B	Bit1 Error: During a message transmission, the CAN node tried to send a recessive level (1) outside the arbitration field and the acknowledge slot, but the monitored bus value was dominant.
101 _B	Bit0 Error: Two different conditions are signaled by this code: <ol style="list-style-type: none"> 1. During transmission of a message (or acknowledge bit, active-error flag, received flag), the CAN node tried to send a dominant level (0), but the monitored bus value was recessive. 2. During bus-off recovery, this code is set each time a sequence of 11 recessive bits has been monitored. The CPU may use this code as indication that the bus is not continuously disturbed.
110 _B	CRC Error: The CRC checksum of the received message was incorrect.

Wake-up CAN Frame Error Counter Register

FRMERRCNT

Wake-up CAN Error Counter Register (B2_H)

Reset Value: 00_H

RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
ECNTCLR	0					ECNT	
w	r					r	

Field	Bits	Type	Description
ECNT	[5:0]	rh	<p>SWK CAN Frame Error Counter The CAN Frame Error Counter is incremented when an error message is received and decremented on a reception of a correct message, if ECNT>0. See Section 13.17.18</p> <p>000000_B No Frame Error 000001_B 1 Frame Error have been counted ...B ... 011111_B 3 Frame Errors have been counted 100000_B Error counter overflow - Wake-up filter will be disabled</p>
ECNTCLR	7	w	<p>Error Counter Clear Writing a '1' to this bit clears the error counter FRMERRCNT.ECNT.</p>
0	6	r	<p>Reserved Read as 0 after reset; value last written is read back; should be written with 0.</p>

Feature Discontinued

Wake-up CAN Interrupt and Event Status Clear Register 0
INTESCLR0
Wake-up CAN Interrupt and Event Status Clear Register 0 (B7_H) Reset Value: 00_H
 RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
0	ERSELCLR	MORCCLR	ECOFCLR	CANTOCLR	SYSERRCLR	OSCCNTRUNCLR	0
r	w	w	w	w	w	w	r

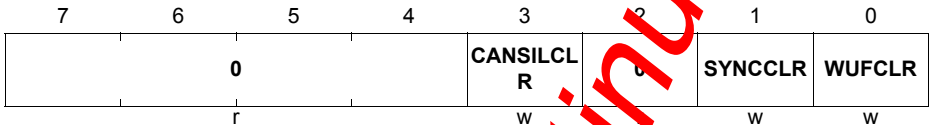
Field	Bits	Type	Description
OSCCNTRUNCLR	1	w	Oscillator counter run status Clear Writing a '1' to this bit clears the corresponding status register bit INTESTAT0.OSCCNTRUN .
SYSERRCLR	2	w	CAN WUP System Error Clear Writing a '1' to this bit clears the corresponding status register bit INTESTAT0.SYSERR .
CANTOCLR	3	w	CAN Time Out Detection Clear Writing a '1' to this bit clears the corresponding status register bit INTESTAT0.CANTO .
ECOFCLR	4	w	Error Counter Overflow Clear Writing a '1' to this bit clears the corresponding status register bit INTESTAT0.ECOFL .
MORCCLR	5	w	Message Object Register Changed Clear Writing a '1' to this bit clears the corresponding status register bit INTESTAT0.MORC .
ERSELCLR	6	w	Error Select Clear Writing a '1' to this bit clears the corresponding status register bit INTESTAT0.ERSEL .
0	0,7		Reserved Read as 0 after reset; value last written is read back; should be written with 0.

Feature Discontinued

Wake-up CAN Interrupt and Event Status Clear Register 1

INTESCLR1

Wake-up CAN Interrupt and Event Status Clear Register 1 (B_H) Reset Value: 00_H
 RMAP: 0, PAGE: 1



Field	Bits	Type	Description
WUFCLR	0	w	STAT.WUF bit clear/acknowledge Clear Writing a '1' to this bit clears the corresponding status register bit INTESTAT1.WUF .
SYNCCLR	1	w	Synchronisation (at least one can frame without fail must have been received) Clear Writing a '1' to this bit clears the corresponding status register bit INTESTAT1.SYNC .
CANSILCLR	3	w	CAN Silent Time during SWK operation Clear Writing a '1' to this bit clears the corresponding status register bit INTESTAT1.CANSIL .
0	2, [7:4]	r	Reserved Read as 0 after reset; value last written is read back; should be written with 0.

Feature Discontinued

13.17.1.5 Wake-up Frame Configuration and Detection

The received frame is compared to the configured wake-up frame and in case a match is detected a wake-up request is issued. The wake-up request sets the flag and issues an interrupt.

*Note: In addition to the message object configuration registers (i.e. **DLC_CTRL**, **IDn_CTRL**(n=0-3), **MASK_IDn_CTRL**(n=0-3), **DATAN_CTRL** (n = 0-7)) the node bit timing configuration is configured in the **BTLn_CTRL**(n=1-2) register.*

Configuration of wake-up frame filter

The message object configuration of the wake-up frame are done in the following registers

- Register **IDn_CTRL** (n=0-3) contains the CAN identifier filter bits.
- Register **MASK_IDn_CTRL** (n=0-3) contains the ID mask bits.
- Register **DLC_CTRL** are the data length code filter bits.
- Register **DATAN_CTRL** (n = 0-7) are the data filter bits.

Detection of wake-up frame filter

The compare mechanisms are different according to the information field inside the CAN frame.

- The identifier programmed inside **IDn_CTRL** (n=0-3) and **MASK_IDn_CTRL** (n=0-3) have to match the data in the received frame and the logic seen in **Figure 13-77**, a wake-up condition is detected.
- The DLC programmed in **DLC_CTRL** together with the data filter bits in **DATAN_CTRL** (n = 0-7) will be compared to the DLC and data inside the received frame, in case a single '1' match in **DATAN_CTRL** (n = 0-7), a wake-up condition within the data is detected. DLC values greater than 8 will be treated as 8. See **Figure 13-77**.

If and only if, identifier and masking, the **DLC_CTRL** and the **DATAN_CTRL** (n = 0-7) will lead to a match, a valid wake-up Frame (WUF) is detected and a wake-up interrupt will be issued if **CFG.SELWUF_EN** is enabled.

Feature Discontinued

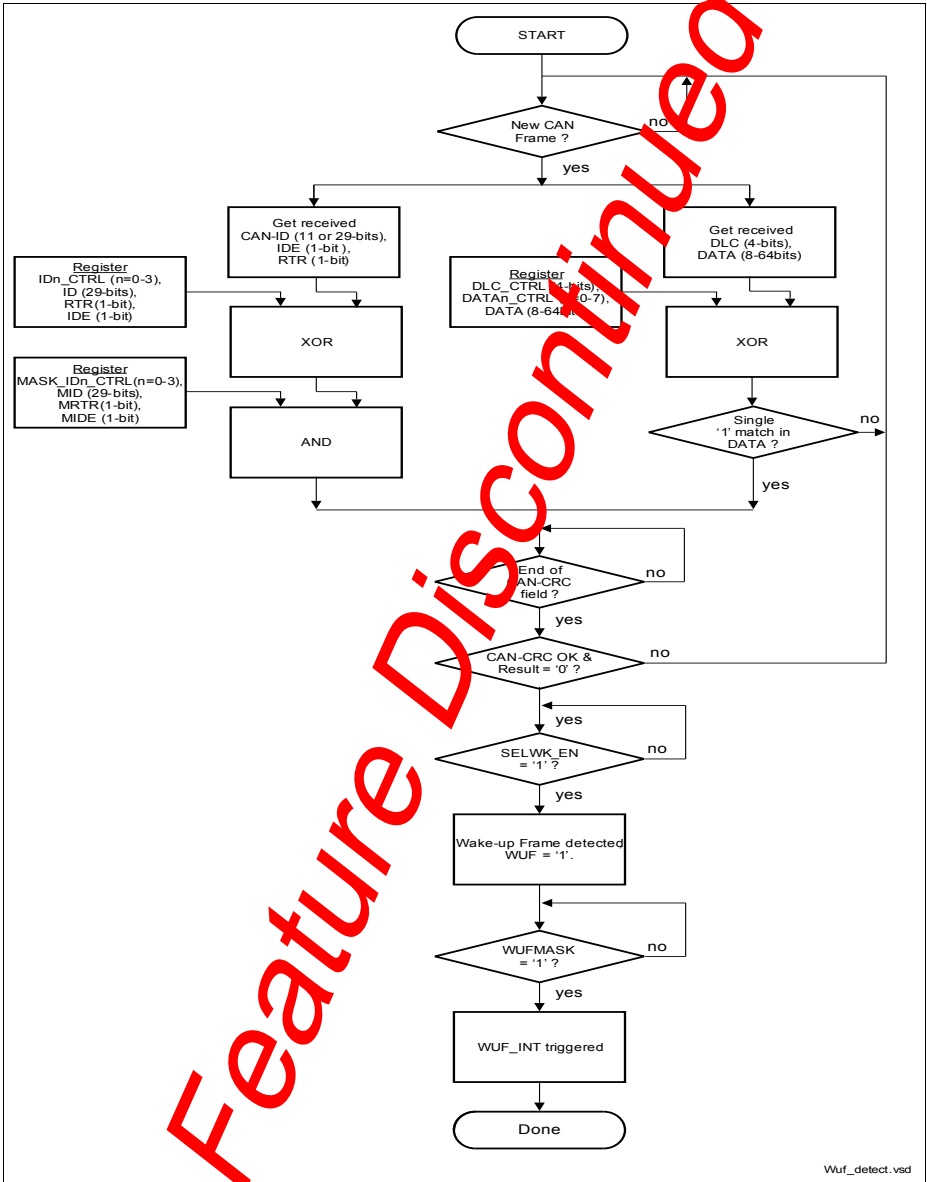


Figure 13-77 Wake-up Frame Detection

13.17.1.6 Oscillator Trimming

An internal oscillator trimming mechanism is available which is able to trim the internal 100MHz RC-OSC, F_{EVR_OSC} present at the EVR. Trimming of the internal high precision oscillator is implemented to cover process and supply / frequency variation over temperature.

The internal oscillator trimming process is achieved by calibrating the internal 100MHz RC-OSC (F_{EVR_OSC}) with a more accurate reference clock F_{OSCO} (XTAL) or F_{SYNC} via the SYNC pin selected by programming **INTMRSLT.GATESEL**.

Next the length of time for F_{OSCO} (XTAL) and F_{EVR_OSC} (100MHz) comparison has to be selected by programming the 16-bit **REFCLK0/1** register. The XTALPERIOD0/1 allows a value from 0 to 65535, say when a value of 8 is chosen, 8 periods of F_{OSCO} (XTAL) time is used to compare with F_{EVR_OSC} and vice versa.

The difference between F_{OSCO} (XTAL) and F_{EVR_OSC} (100MHz) comparison for the length of time defined by **REFCLK0/1** register is accumulated over the 16bit counter (**CLKCMP0/1.OSCNT**). See [Figure 13-78](#)

To start the process of F_{EVR_OSC} with reference clock (F_{OSCO}/F_{SYNC}) comparison, **CFG.OSCCAL** and **CFG.TRIMEN** have to be first programmed accordingly. When comparison of F_{EVR_OSC} with the reference clock signal is completed, an interrupt (**OSCCNTRUN_INT**) is generated and **INTLSTAT0.OSCCNTRUN** flag stopped.

The difference accumulated over **CLKCMP0/1.OSCNT** is communicated by the SCR (safe CPU) to the EVR module and the difference value is used in the trimming register, thereby completing the trimming process.

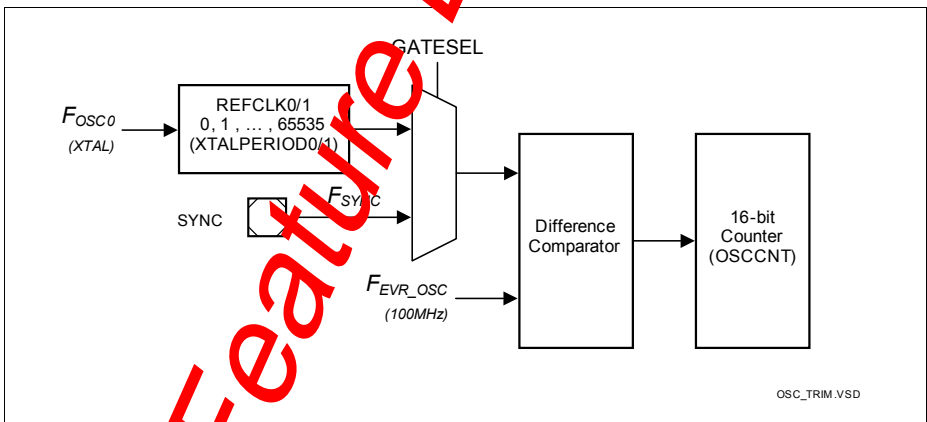


Figure 13-78 Oscillator trimming

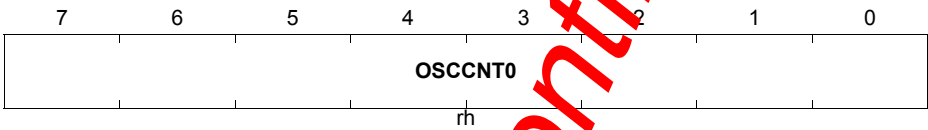
Clock Compare Result Register

Clock Compare Result Register 0

The Clock Compare Result Register contains the difference in terms of counts of the reference clock (F_{OSCO} or F_{SYNC}) and the internal 100MHz RC Oscillator (F_{EVR_OSC}).

CLKCMP0

Clock Compare Result Register 0 (B4_H) **Reset Value: 00_H**
RMAP: 0, PAGE: 0

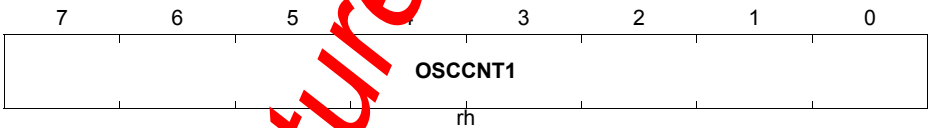


Field	Bits	Type	Description
OSCCNT0	[7:0]	rh	Oscillator Counter This bit field specifies the lower LSB in the 16-bit Oscillator counter.

Clock Compare Result Register 1

CLKCMP1

Clock Compare Result Register 1 (B5_H) **Reset Value: 00_H**
RMAP: 0, PAGE: 0



Field	Bits	Type	Description
OSCCNT1	[7:0]	rh	Oscillator Counter This bit field specifies the upper MSB in the 16-bit Oscillator counter.

Feature Discontinued

13.17.1.7 CAN Message Timeout

A timeout mechanism to monitor CAN bus activity is available on the WCAN. The timeout mechanism monitors the RXDC0 pin. When the RXDC0 pin is inactive for a $t_{SILENCE}$ time of 750ms, both CAN bus silent bit (**INTESTAT1.CANSIL**) and CAN timeout bit (**INTESTAT0.CANTO**) will be set with a CAN time out interrupt (**CANTO_INT**).

The CAN bus silent bit is reset when there is activity on the CAN bus where else the CAN timeout bit can be reset by setting CAN time out clear bit (**INTESTCLR0.CANTOCLR**).

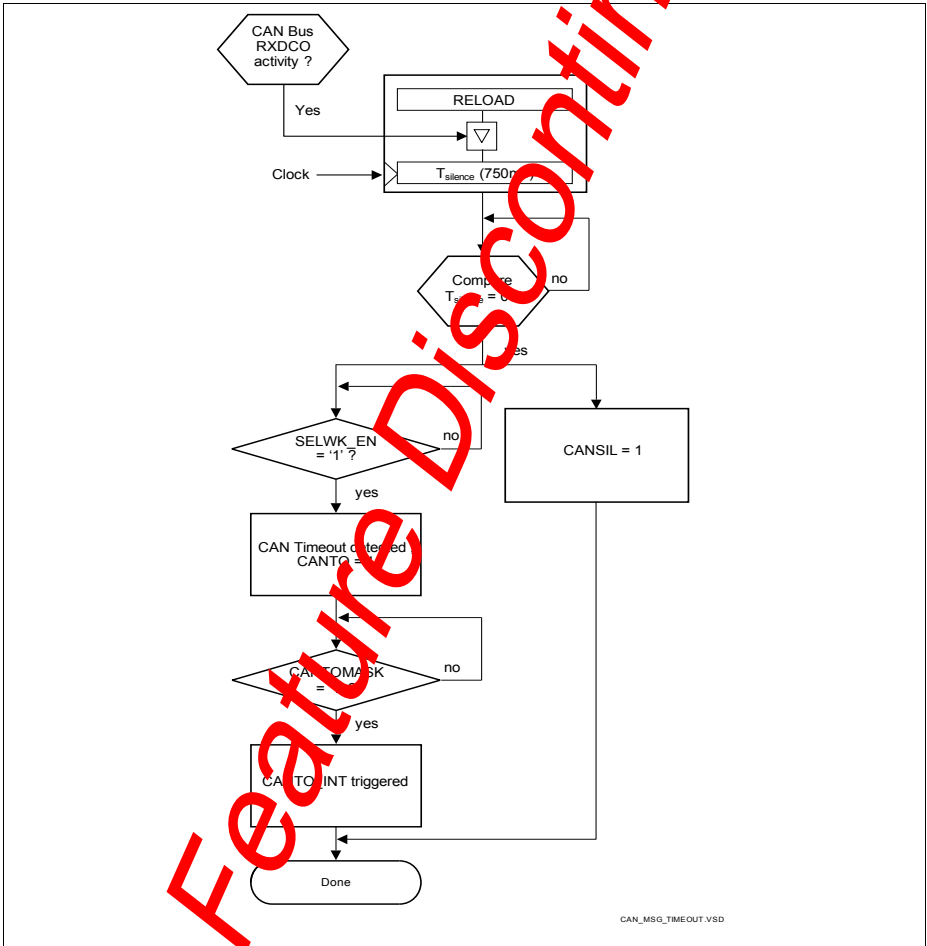


Figure 13-79 CAN Timeout

13.17.1.8 Error Counter

The error counting mechanism in ISO 11898-6 is different to ISO 11898-1. As the device is only receiving messages and never transmitting it could never get into a bus-off situation. Therefore an ISO11898-6 implementation is incrementing its error counter on a Form, Stuff and a CRC error will lead to an increment of the error counter. In case of a correct message, the error will be decremented, if it is greater than 0. In case the error counter is greater than 31 an overflow occurs and a wake-up will be initiated. This shall enable the software to correct the situation. In case of overflow the SYSERR flag is set. See [Figure 13-80](#)

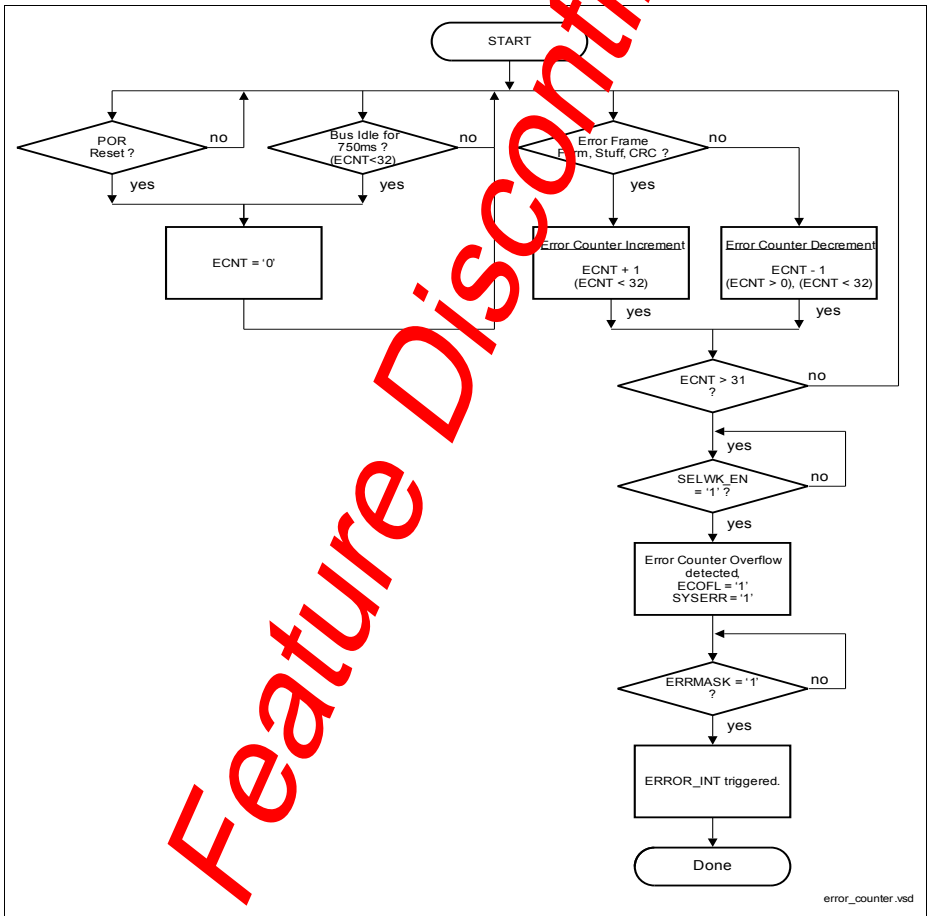


Figure 13-80 Error counter

13.17.2 CAN Functional Description

13.17.2.1 CAN Node Control

The CAN node is equipped with an individual set of SFR registers to control and to monitor the CAN node.

Bit Timing

According to ISO 11898 standard, a CAN bit time is subdivided into different segments (). Each segment consists of multiples of a time quantum t_q . The magnitude of t_q is adjusted by the bit field BRP, both controlling the baud rate prescaler (see bit timing register BTLn_CTRL(n=1-2)). The baud rate prescaler is driven by the MultiCAN module clock f_{CAN} . For ISO 11898-6 some settings are guaranteed to be working according to the given propagation delay times (see [Section](#)). Working settings for 125k, 250k and 500k will be provided. The information below is given to enable settings for different networks.

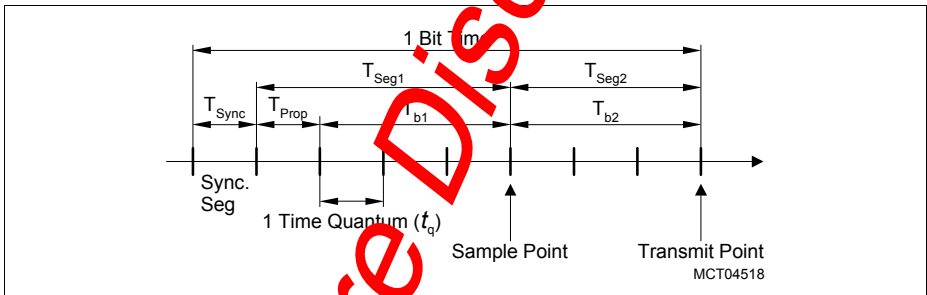


Figure 13-81 CAN Bus Bit Timing Standard

The Synchronization Segment (T_{Sync}) allows a phase synchronization between transmitter and receiver time bases. The Synchronization Segment length is always $1 t_q$. The Propagation Time Segment (T_{Prop}) takes into account the physical propagation delay in the transmitter output driver, on the CAN bus line and in the transceiver circuit. For a working collision detect mechanism, T_{Prop} has to be two times the sum of all propagation delay quantities rounded up to a multiple of t_q . The Phase Buffer Segments 1 and 2 (T_{b1} , T_{b2}) before and after the signal sample point are used to compensate a mismatch between transmitter and receiver clock phase detected in the synchronization segment.

The maximum number of time quanta allowed for resynchronization is defined by bit field SJW in the CAN Node Bit Timing register BTLn_CTRL(n=1-2). The Propagation Time Segment and the Phase Buffer Segment 1 are combined to parameter TSeg1, which is

defined by the value TSEG1 in the respective CAN Node Bit Timing register BTLn_CTRL(n=1-2). A minimum of 3 time quanta is requested by the ISO standard. Parameter TSeg2, which is defined by the value of TSEG2 in the CAN Node Bit Timing Register BTLn_CTRL(n=1-2), covers the Phase Buffer Segment (t_q) minimum of 2 time quanta is requested by the ISO standard. According ISO standard, a CAN bit time, calculated as the sum of T_{Sync}, T_{Seg1} and T_{Seg2}, must not fall below 8 time quanta.

Calculation of the bit time:

$$\begin{aligned}
 t_q &= (\text{BRP} + 1) / f_{\text{CAN}} \\
 T_{\text{Sync}} &= 1 \times t_q \\
 T_{\text{Seg1}} &= (\text{TSEG1} + 1) \times t_q \quad (\text{min. } 3 t_q) \\
 T_{\text{Seg2}} &= (\text{TSEG2} + 1) \times t_q \quad (\text{min. } 2 t_q) \\
 \text{bit time} &= T_{\text{Sync}} + T_{\text{Seg1}} + T_{\text{Seg2}} \quad (\text{min. } 8 t_q)
 \end{aligned}$$

To compensate phase shifts between clocks of different CAN controllers, the CAN controller has to synchronize on any edge from the recessive to the dominant bus level. If the hard synchronization is enabled (at the start of frame), the bit time is restarted at the synchronization segment. Otherwise, the resynchronization jump width T_{SJW} defines the maximum number of time quanta a bit time may be shortened or lengthened by one resynchronization. The value of SJW is programmed in the CAN Node Bit Timing Register.

$$\begin{aligned}
 T_{\text{SJW}} &= (\text{SJW} + 1) \times t_q \\
 T_{\text{Seg1}} &\geq T_{\text{SJW}} + T_{\text{prop}} \\
 T_{\text{Seg2}} &\geq T_{\text{SJW}}
 \end{aligned}$$

The maximum relative tolerance for f_{CAN} depends on the Phase Buffer Segments and the resynchronization jump width:

$$\begin{aligned}
 df_{\text{CAN}} &\leq \min(T_{b1}, T_{b2}) \times (1.3 \times \text{bit time} - T_{b2}) \quad \text{AND} \\
 df_{\text{CAN}} &\leq T_{\text{SJW}} / 20 \times \text{bit time}
 \end{aligned}$$

The configuration change enable, **CFG.CCE** bit has to be enabled before a valid CAN bit timing can be written to the CAN Node Bit Timing Register, BTLn_CTRL(n=1-2). This is then followed by resetting the INIT bit in **CFG.INITREQ**, i.e. before enabling the operation of the CAN node.

Note: The proposed sample point is at 80% of the bit time. For optimum oscillator tolerance, $BTLn(CTRL(n=1-2))$ setting should be done such that the number of time quantas, t_q and SJW is maximum.

Network Propagation Delays

The “worst case” network propagation delay is the network propagation delay after the 5th bit for the dominant to recessive edge on the bus. (See [Figure 13-82](#))

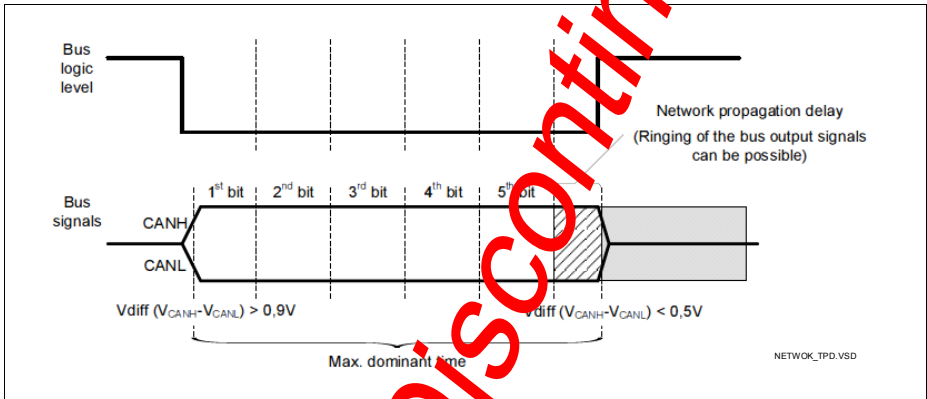


Figure 13-82 Network propagation delay after 5th bit

The WCAN module will sample the 5th bit after 5 consecutive dominant bits as recessive bit to ensure a safe wake-up, even if the dominant to recessive edge is delayed for the value of the network propagation delay shown in [Table 13-69](#). The network propagation delay describes the maximal occurrence propagation delay which can be seen on the bus. This maximum network propagation delay does not include any EMC jitter. (i.e no HF injection)

Table 13-69 Network propagation delay

Baud Rate (Baud)	Network Propagation Delay (ns)	Maximum Dominant time (ns) (sender oscillator tolerance included)
1MBaud	550	5550
500KBaud	1350	11350
250KBaud	2675	22675
125KBaud	5400	45450

For HF injection the WCAN module is able to handle EMC network propagation delay as shown in [Table 13-70](#).

Table 13-70 EMC Network propagation delay

Baud Rate (Baud)	Network Propagation Delay (ns)	Maximum Dominant time (ns) (sender oscillator tolerance included)
1MBaud	450	5450
500KBaud	1250	11250
250KBaud	2575	22575
125KBaud	5350	45350

Also the WCAN module is able to decode frames that contain sequences of 5 consecutive dominant bits followed by 5 consecutive recessive bits as shown in **Figure 13-82**.

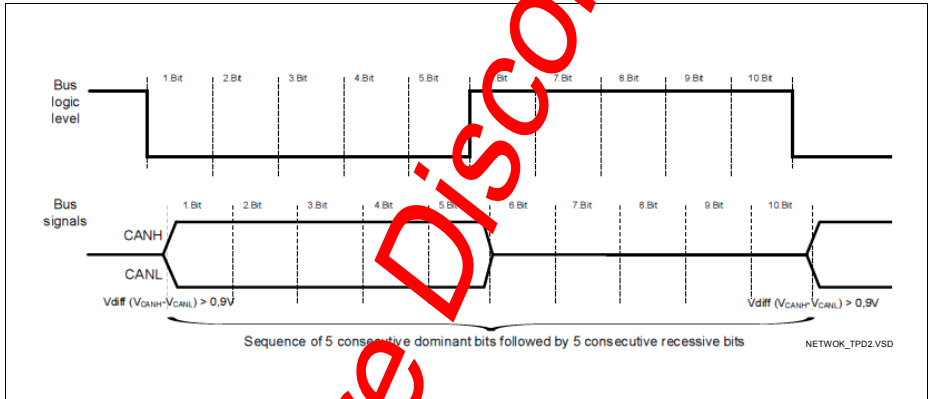


Figure 13-83 Network propagation delay after 10th bit

The transmitting CAN node on the bus has to guarantee a maximum baud rate tolerance of +/- 0.4% for correct detection of the CAN frames (See **Table 13-71**)

Table 13-71 Worst Case length of 10 bits

Baud Rate (Baud)	Worst Case time for length of 10 bits by sender oscillator tolerance (-0.4%)	Worst Case time for length of 10 bits by sender oscillator tolerance (+0.4%)
1MBaud	9970ns	10040ns
500KBaud	19970ns	20080ns

Table 13-71 Worst Case length of 10 bits

Baud Rate (Baud)	Worst Case time for length of 10 bits by sender oscillator tolerance (-0.4%)	Worst Case time for length of 10 bits by sender oscillator tolerance (+0.4%)
250KBaud	39840ns	40160ns
125KBaud	79680ns	80320ns

Feature Discontinued

13.17.2.2 WCAN Kernel Registers

Register Address Map

The register set of the WCAN module consists of the below distinct subsets:

1. The **Configuration, Miscellaneous Registers** exists in one instance and applies for the entire WCAN module.
2. The **Wake-up CAN Registers** consists of the following:
 - a) **CAN Node Registers** (BTLx_CTRL), which applies to a single CAN node and exist once for each CAN node
 - b) **Message Object Registers** (DLC_CTRL, IDx_CTRL, MASK_IDx_CTRL, DATAx_CTRL) is define for a single message object and thus exists once for each message object.

Configuration Registers

The configuration registers of a WCAN node are located according to [Table 13-72](#) which shows the absolute address as seen on the Standby Controller (SCR).

Table 13-72 CAN Nodes Register Address Map (Relative to MultiCAN Base Address)

Register	Addr.	Page	Total Range
CFG	B0 _H	0	Configuration Register
INTMRSLT	B1 _H	0	Interrupt Mask and Reference Select Register
REFCLK0	B2 _H	0	Clock Divider 0 Register
REFCLK1	B3 _H	0	Clock Divider 1 Register
CLKCMP0	B4 _H	0	Clock Compare Result 0 Register
CLKCMP1	B5 _H	0	Clock Compare Result 1 Register
INTESTAT0	B0 _H	1	Interrupt and Event Status 0 Register
INTESTAT1	B1 _H	1	Interrupt and Event Status 1 Register
FRMERRCNT	B3 _H	1	Frame Error Counter Register
INTESCLR0	B3 _H	1	Interrupt and Event Status Clear Register
INTESCLR1	B4 _H	1	Interrupt and Event Status Clear Register

CAN Node Registers

The registers of a CAN node are located according to [Table 13-73](#) which shows the absolute address of the CAN Node Registers as seen on the Standby Controller (SCR).

Table 13-73 CAN Nodes Register Address Map (Relative to MultiCAN Base Address)

Register	Addr.	Page	Total Range
BTL1_CTRL	B5 _H	1	Bit Timing Logic Control Register
BTL2_CTRL	B6 _H	1	Bit Timing Logic Control Register

Message Object Registers

The registers of a message object are located according to [Table 13-74](#) which shows the absolute address of the CAN Message Object Registers as seen on the Standby Controller (SCR).

Table 13-74 CAN Message Object Registers Address Map (Relative to MultiCAN Base Address)

Register	Addr.	Page	Full Name of Register
DLC_CTRL	B7 _H	1	Message Data Length Code Control Register
ID0_CTRL	B0 _H	2	Message Identifier Control Register 0
ID1_CTRL	B1 _H	2	Message Identifier Control Register 1
ID2_CTRL	B2 _H	2	Message Identifier Control Register 2
ID3_CTRL	B3 _H	2	Message Identifier Control Register 3
MASK_ID0_CTRL	B4 _H	2	Message Identifier Acceptance Mask Register 0
MASK_ID1_CTRL	B5 _H	2	Message Identifier Acceptance Mask Register 1
MASK_ID2_CTRL	B6 _H	2	Message Identifier Acceptance Mask Register 2
MASK_ID3_CTRL	B7 _H	2	Message Identifier Acceptance Mask Register 3
DATA0_CTRL	B0 _H	3	Message Data Control Register 0
DATA1_CTRL	B1 _H	3	Message Data Control Register 1
DATA2_CTRL	B2 _H	3	Message Data Control Register 2
DATA3_CTRL	B3 _H	3	Message Data Control Register 3
DATA4_CTRL	B4 _H	3	Message Data Control Register 4
DATA5_CTRL	B5 _H	3	Message Data Control Register 5
DATA6_CTRL	B6 _H	3	Message Data Control Register 6
DATA7_CTRL	B7 _H	3	Message Data Control Register 7

CAN Node Registers

The CAN node specific registers exist once for each CAN node of the MultiCAN module. They contain information that is directly related to the operation of the CAN nodes and which may not be shared among the nodes.

The Node Control Register contains basic settings that define the operation of the CAN node and the interaction of the CAN node with the message objects.

Bit Timing Logic Control Register

The Bit Timing Logic Control Registers contains all parameters to setup the bit timing for the CAN transfer.

Note: The startup-software (SSW) overwrites the BTL1_CTRL register with 14_H and BTL2_CTRL register with 58_H after reset to obtain a 500Kbits/s baud rate. Please use the following values for 1Mbits/s (BTL1_CTRL1, BTL2_CTRL), 250Kbits/s (BTL1_CTRL1, BTL2_CTRL) and 125Kbits/s (BTL1_CTRL1, BTL2_CTRL) baud rate.

The Configuration Change Enable, CFG.CCE has to be enabled before the BTLx_CTRL register can be written.

BTL1_CTRL

Bit Timing Logic 1 Control Register (15_H)

Reset Value: 7D_H

RMAP: 0, PAGE: 1

7	6	5	4	3	2	1	0
0	TSEG2			TSEG1			
r	rw			rw			

Field	Bits	Type	Description
TSEG1	[3:0]	rw	Time Segment Before Sample Point (TSEG1 + 1) time quanta is the user-defined nominal time between the end of the synchronization segment and the sample point. It includes the propagation segment, which takes into account signal propagation delays. The time segment may be lengthened due to re-synchronization. Valid values for TSEG1 are 2 to 15.

Field	Bits	Type	Description
TSEG2	[6:4]	rw	Time Segment After Sample Point (TSEG2 + 1) time quanta is the user defined nominal time between the sample point and the start of the next synchronization segment. It may be shortened due to resynchronization. Valid values for TSEG2 are 4 to 7.
0	7	r	Reserved Read as 0 after reset, value last written is read back; should be written with 0.

Feature Discontinued

Bit Timing Logic Control Register

BTL2_CTRL

Bit Timing Logic 2 Control Register (B6_H)

Reset Value: C7_H

RMAP: 0, PAGE: 1



Field	Bits	Type	Description
BRP	[5:0]	rw	Baud Rate Prescaler The duration of one time quantum is given by (BRP + 1) clock cycles.
SJW	[7:6]	rw	(Re)Synchronization Jump Width (SJW + 1) time quanta are allowed for resynchronization.

Feature Discontinued

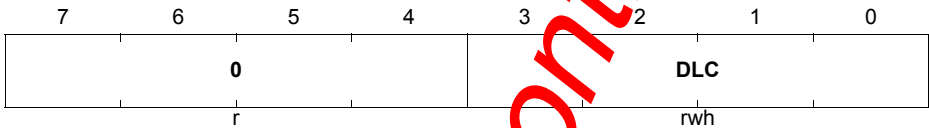
Message Object Registers

Message Data Length Code Control Register

Register DLC_CTRL are the data length code filter bits. There are no additional mask bits. See [Figure 13-77](#) for flow of acceptance filtering which would determine if a wake-up frame has been detected.

DLC_CTRL

Message Data Length Code Control Register(B7_H) **Reset Value: 00_H**
RMAP: 0, PAGE: 1



Field	Bits	Type	Description
DLC	[3:0]	rwh	Data Length Code Valid values for the data length are 0 to 8. DLC = 8 leads to 8 data bytes, but the DLC code is not truncated upon reception or transmission of CAN frames.
0	[7:4]	r	Reserved Read as 0 after reset; value last written is read back; should be written with 0.

Feature Discontinued

Message Identifier Control Register 0

Register IDn_CTRL (n=0-3) contains the CAN identifier filter bits. See [Figure 13-77](#) for flow of acceptance filtering which would determines if a Wake-up Frame have been detected.

ID0_CTRL

Message Identifier Control Register 0 (B0_H)

Reset Value: 00_H

RMAP: 0, PAGE: 2

7	6	5	4	3	2	1	0
0	ID					RTR	IDE
r	rwh					rwh	rwh

Field	Bits	Type	Description
IDE	0	rwh	Identifier Extension Bit of Message Object 0 0 _B Standard frame with 11-bit identifier 1 _B Extended frame with 29-bit identifier
RTR	1	rwh	Remote Transmission Request Bit 0 _B Data Frame 1 _B Remote Frame
ID	[6:2]	rwh	CAN Identifier of Message Object 0 Identifier of a standard message (ID[28:18]) or an extended message (ID[28:0]). For standard identifiers, bits ID[17:0] are "don't care". This bit field specifies ID4 to ID0.
0	7	r	Reserved Read as 0 after reset; value last written is read back; should be written with 0.

Message Identifier Control Register 1

ID1_CTRL

Message Identifier Control Register 1 (B1_H)

Reset Value: 00_H

RMAP: 0, PAGE: 2

7	6	5	4	3	2	1	0
ID							
rwh							

Field	Bits	Type	Description
ID	[7:0]	rwh	CAN Identifier of Message Object 0 Identifier of a standard message (ID[28:18]) or an extended message (ID[28:0]). For standard identifiers, bits ID[17:0] are "don't care". These bit field specifies ID12 to ID5.

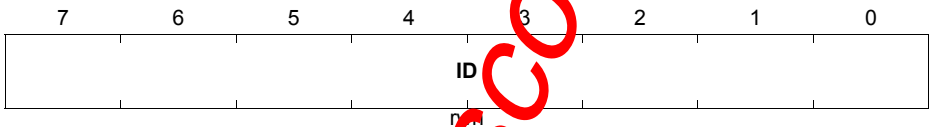
Message Identifier Control Register 2

ID2_CTRL

Message Identifier Control Register 2 (B2_H)

Reset Value: 00_H

RMAP: 0, PAGE: 2



Field	Bits	Type	Description
ID	[7:0]	rwh	CAN Identifier of Message Object 0 Identifier of a standard message (ID[28:18]) or an extended message (ID[28:0]). For standard identifiers, bits ID[17:0] are "don't care". This bit field specifies ID20 to ID13.

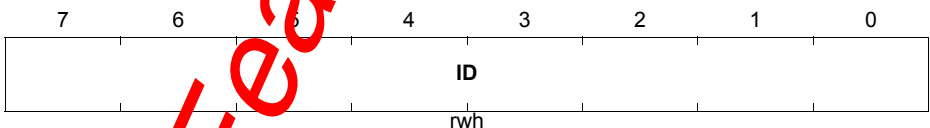
Message Identifier Control Register 3

ID3_CTRL

Message Identifier Control Register 3 (B3_H)

Reset Value: 00_H

RMAP: 0, PAGE: 2



Feature Discontinued

Field	Bits	Type	Description
ID	[7:0]	rwh	CAN Identifier of Message Object 0 Identifier of a standard message (ID[28:18]) or an extended message (ID[28:0]). For standard identifiers, bits ID[17:0] are "don't care". These bit field specifies ID28 to ID21.

Feature Discontinued

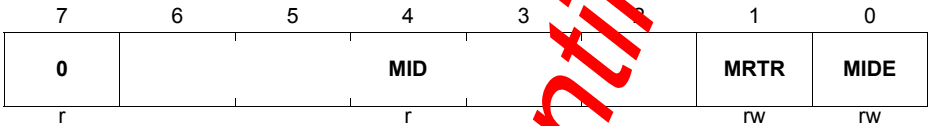
Message Identifier Mask Register 0

Register MASK_IDn_CTRL (n = 0-3) contains the mask bits for the message object. See [Figure 13-77](#) for flow of acceptance filtering which would determine if a Wake-up Frame have been detected.

MASK_ID0_CTRL

Message Identifier Mask Register 0 (B4_H)
RMAP: 0, PAGE: 2

Reset Value: 00_H



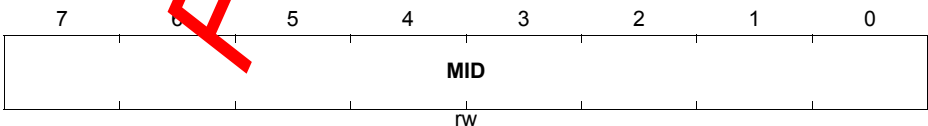
Field	Bits	Type	Description
MIDE	0	rw	Mask Bit for Message IDE Bit Mask to filter standard frame (11-bit) or extended frame (29-bit).
MRTR	1	rw	Mask Bit for Request Transmission Request Bit Mask to filter Data Frame or Remote Frame.
MID	[6:2]	rw	Mask for Message Identifier Mask to filter incoming messages with standard identifiers (MID[28:18]) or extended identifiers (MID[28:0]). For standard identifiers bits MID[17:0] are "don't care". This bit field specifies MID4 to MID0.
0	7	r	Reserved Read as 0 after reset; value last written is read back; should be written with 0.

Message Identifier Mask Register 1

MASK_ID1_CTRL

Message Identifier Mask Register 1 (B5_H)
RMAP: 0, PAGE: 2

Reset Value: 00_H



Field	Bits	Type	Description
MID	[7:0]	rw	Mask for Message Identifier Mask to filter incoming messages with standard identifiers (MID[28:18]) or extended identifiers (MID[28:0]). For standard identifiers bits MID[17:0] are “don’t care”. These bit field specifies MID12 to MID5.

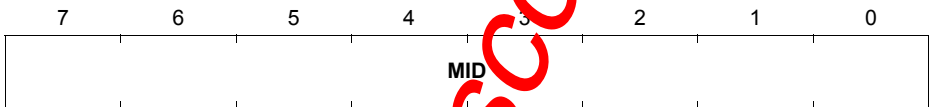
Message Identifier Mask Register 2

MASK_ID2_CTRL

Message Identifier Mask Register 2 (B6_H)

Reset Value: 00_H

RMAP: 0, PAGE: 2



Field	Bits	Type	Description
MID	[7:0]	rw	Mask for Message Identifier Mask to filter incoming messages with standard identifiers (MID[28:18]) or extended identifiers (MID[28:0]). For standard identifiers bits MID[17:0] are “don’t care”. These bit field specifies MID20 to MID13.

Message Identifier Mask Register 3

MASK_ID3_CTRL

Message Identifier Mask Register 3 (B7_H)

Reset Value: 00_H

RMAP: 0, PAGE: 2



Field	Bits	Type	Description
MID	[7:0]	rw	<p>Mask for Message Identifier</p> <p>Mask to filter incoming messages with standard identifiers (MID[28:18]) or extended identifiers (MID[28:0]). For standard identifiers bits MID[17:0] are “don’t care”.</p> <p>These bit field specifies MID28 to MID21.</p>

Feature Discontinued

Data Control Register

Data Control Register, $DATA_n_CTRL(n = 0-7)$ contains the data filter bits. See [Figure 13-77](#) for flow of acceptance filtering which would determine if a Wake-up Frame have been detected. It is a match if a single bit is equally set at the same bit position in the incoming data and $DATA_n_CTRL(n = 0-7)$ register.

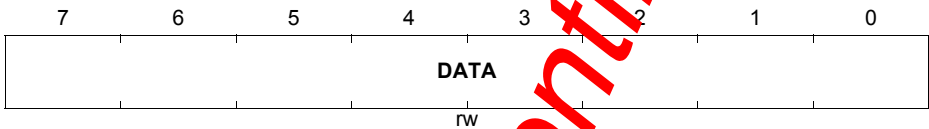
$DATA_n_CTRL (n = 0-7)$

Data Control Register

$(B0_H+n*1_H)$

Reset Value: 00_H

RMAP: 0, PAGE: 3



Field	Bits	Type	Description
DATA	[7:0]	rw	CAN Data Byte These bit fields specifies bit 7 to bit 0 of the 8-bit data that will be compared with data in the received message object.

Feature Discontinued

13.17.2.3 Miscellaneous Register

Paging Register

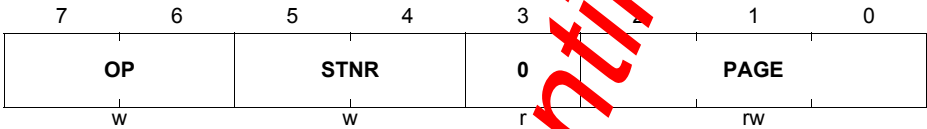
WCAN_PAGE

Page Register for WCAN

(C9_H)

Reset Value: 00_H

RMAP: 0, PAGE: X



Field	Bits	Type	Description
PAGE	[2:0]	rw	<p>Page Bits</p> <p>When written, the value indicates the new page address.</p> <p>When read, the value indicates the currently active page address [y:x+1].</p>
STNR	[5:4]	w	<p>Storage Number</p> <p>This number indicates which storage bit field is the target of the operation defined by bit OP.</p> <p>If OP = 10_B, the contents of PAGE are saved in STx before being overwritten with the new value.</p> <p>If OP = 11_B, the contents of PAGE are overwritten by the contents of STx. The value written to the bit positions of PAGE is ignored.</p> <p>00_B ST0 is selected. 01_B ST1 is selected. 10_B ST2 is selected. 11_B ST3 is selected.</p>

Feature Discontinued

Field	Bits	Type	Description
OP	[7:6]	w	Operation 0x _B Manual page mode. The value of STNR is ignored and PAGE is directly written. 10 _B New page programming with automatic page saving. The value written to the bit positions of PAGE is stored. In parallel, the former contents of PAGE are saved in the storage bit field STx indicated by STNR. 11 _B Automatic restore page action. The value written to the bit positions PAGE is ignored and instead, PAGE is overwritten by the contents of the storage bit field STx indicated by STNR.
0	3	r	Reserved Returns 0 if read, should be written with 0.

Feature Discontinued

13.17.3 WCAN Module Implementation

This section describes CAN module interfaces with the clock control, port connections, interrupt control, and address decoding.

13.17.3.1 Wake-up CAN Module Registers

The registers listed in [Figure 13-84](#)

Configuration Registers	Wake up CAN Registers	Miscellaneous Registers
CFG	BTL1_CTRL	WCAN.PAGE
INTMRSLT	BTL2_CTRL	
CLKDIV0	DLC_CTRL	
CLKDIV1	ID0_CTRL	
INTESTAT0	ID1_CTRL	
INTESTAT1	ID2_CTRL	
FRMERRCNT	ID3_CTRL	
INTESCLR0	MASK_ID0_CTRL	
INTESCLR1	MASK_ID1_CTRL	
CLKCMP0	MASK_ID2_CTRL	
CLKCMP1	MASK_ID3_CTRL	
	DATA0_CTRL	
	DATA1_CTRL	
	DATA2_CTRL	
	DATA3_CTRL	
	DATA4_CTRL	
	DATA5_CTRL	
	DATA6_CTRL	
	DATA7_CTRL	

WakeupCAN_SFR.vsd

Figure 13-84 Wake-up CAN Special Function Registers on Standby Controller (SCR)

Feature Discontinued

The addresses of the kernel SFRs are listed in

Table 13-75 SFR Address List for Pages 0 – 3

Address	Page 0	Page 1	Page 2	Page 3
B0H	CFG	INTESTAT0	ID0_CTRL	DATA0_CTRL
B1H	INTMRSLT	INTESTAT1	ID1_CTRL	DATA1_CTRL
B2H	REFCLK0	FRMERRCNT	ID2_CTRL	DATA2_CTRL
B3H	REFCLK1	INTESCLR0	ID3_CTRL	DATA3_CTRL
B4H	CLKCMP0	INTESCLR1	MASK_ID0_CTRL	DATA4_CTRL
B5H	CLKCMP1	BTL1_CTRL	MASK_ID1_CTRL	DATA5_CTRL
B6H		BTL2_CTRL	MASK_ID2_CTRL	DATA6_CTRL
B7H		DLC_CTRL	MASK_ID3_CTRL	DATA7_CTRL

Table 13-76 SFR Address Listing for Pages 4 – 7

Address	Page 4	Page 5	Page 6	Page 7
B0H				
B1H				
B2H				
B3H				
B4H				
B5H				
B6H				
B7H				

Feature Discontinued

13.17.3.2 Port and I/O Line Control

The interconnections between the Wake-up CAN module and the port I/O lines are controlled in the port control logic for Port 0, 2 and in the EVR for Port 14, 15. Additionally to the port input selection, the following port control operations must be executed:

- Input/output function selection (SCR_IOCR registers (SCR module) and IOCR registers (GPIO module))

Input/Output Function Selection in Ports

The port input/output control registers contain the bit fields that select the digital output and input driver characteristics such as pull-up/down devices, port direction (input/output), open-drain, and alternate output selection. The I/O lines for the WCAN module are controlled by the port input/output control registers shown below. (See MODPISEL0 register at SCU chapter)

Table 13-77 shows how bits and bit fields must be programmed for the required I/O functionality of the CAN I/O lines.

Table 13-77 WCAN I/O Control Selection and Setup

Module	Port Lines	Input Output Control Register Bits	I/O
CAN	CAN0		
	P02.1 / WCANRXDA	SCR_P02_IOCR1.PC1 = 0XXXX _B	Input
	P02.3 / WCANRXDB	SCR_P2_IOCR3.PC3 = 0XXXX _B	Input
	P02.4 / WCANRXDC	SCR_P2_IOCR4.PC4 = 0XXXX _B	Input
	P00.1 / WCANRXDD	SCR_P0_IOCR1.PC1 = 0XXXX _B	Input
	P00.3 / WCANRXDE	SCR_P0_IOCR3.PC3 = 0XXXX _B	Input
	P14.1 / RXDCAN	P14_IOCR12.PC1 = 0XXXX _B	Input
	P15.1 / RXDCAN	P15_IOCR12.PC1 = 0XXXX _B	Input
	P02.5 / SYNCA	SCR_P2_IOCR5.PC5 = 0XXXX _B	Input
P02.7 / SYNCF	SCR_P2_IOCR7.PC7 = 0XXXX _B	Input	

Interrupt Control

The interrupt control logic in the Wake-up CAN module consists of the following interrupts shown in **Table 13-78** and **Figure 13-85**.

Table 13-78 WCAN Interrupt Outputs

Interrupt inputs	Connected to WCAN Interrupt Output
SCR	WUF_INT (Wake-up Frame Interrupt)
	ERROR_INT (Error Counter Overflow Interrupt)
	OSCCNTRUN_INT (Oscillator Counter Run Interrupt)
	CANTO_INT (CAN Timeout Interrupt)

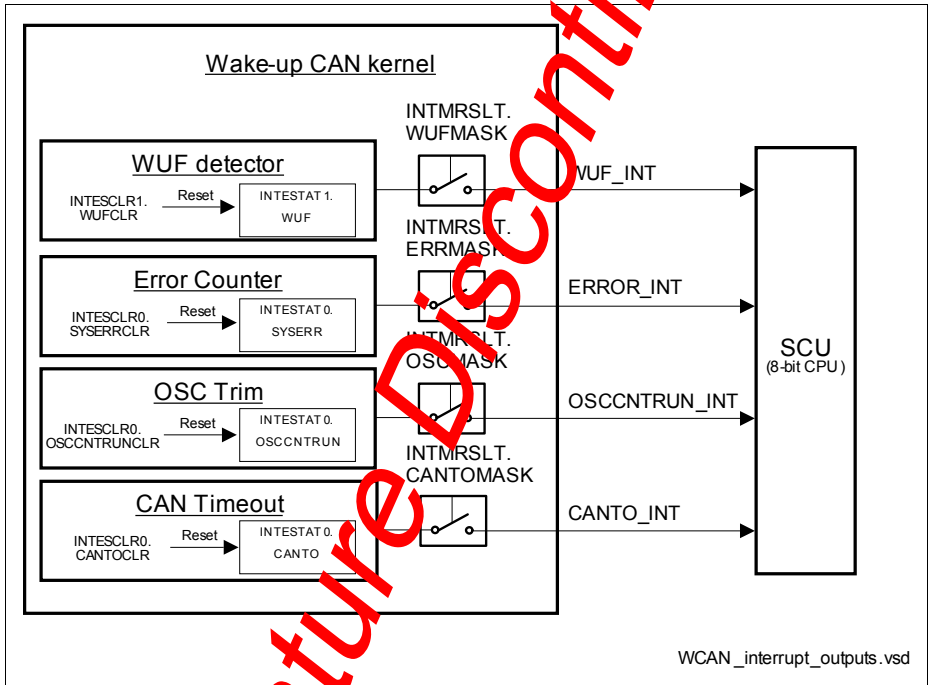


Figure 13-85 Wake-up CAN Interrupt Outputs

13.17.4 Revision history

Revision History

Revision History (from WCAN v0.1 to WCAN v0.2)

- Picture 1-42 Interrupt Compressor picture 64 message objects exchanged by picture 256 message objects
- Correction to match register generation tool
- Mark Oscillator measurement registers as internal (can be external as soon as patent situation is clear)

Revision History (from WCAN v0.2 to WCAN v0.3)

- Unhide all text and Registers set visible internal
 - Section 1.2.2.2 CAN Error Handling
 - Section 1.2.3 CAN Node Interrupts
 - Section 1.2.4 CAN Node Analysis Features
 - Section 1.2.6 Message Postprocessing Feature
 - Section 1.2.7 Message Object Data Handling
 - Section 1.2.8 Message Object Functionality
 - NCR, NSR, NIPR, NBTR, NECNT, NFCR, MOFGPR, MOIPR, MOCTR, ID, PANCTR, MCR, MITR, LIST, MSPND, MSIDK, MSIMASK, NCR, NSR, NIPR, VECNT, NFCR

Revision History (from WCAN v0.3 to WCAN v0.4)

- structured document and added various registers

Revision History (from WCAN v0.4 to WCAN v0.5)

- Corrected external conditional text settings for ITS as it is not correct

Revision History (from WCAN v0.5 to WCAN v0.6)

- Modified **INTESCLR0** and **INTESCLR1** bit settings to “w” for bit clear register
- Corrected **CFG.CFGVAL** to wh
- As per meeting on 6. April CAN Wake-up Filter Function Alignment
 - For node registers, only NBTR, NECNT, NFCR are available in CAN protocol handler, however only NBTR are only made available for CAN timing control. Keeping the same naming structure of (DoorSBC+Selective wake-up), this is the CTRL_DAT0 register. Thus removed NBTR0_DAT0, NBTR0_DAT1 and NPCR0_DAT0 register.
 - For Message Object registers, MOAR0_DATn (n=0-3), MOAMR0_DATn (n=0-3), MOFCR0_DATn (n=0-3), MODATAL0_DATn (n=0-3), MODATAH0_DATn (n=0-3), MOCTRn, MOIPRn, MOFGPRn are not delivered on the CAN protocol handler

and thus these registers are removed. Message Object Registers available for Acceptance Filter Operation to determine Wake-up are added keeping the same naming structure of (Door SBC + selective wake-up) DLC_CTRL, MASK_IDn_CTRL (n=0-3), IDn_CTRL (n=0-3), DATAn_CTRL (n=0-3).

- All other CAN registers such as global module registers ID, PANCTR, MCR, MITR, LISTn, MSPNDk, MSIDk, MSIMASK, NCRx, NSRx, NIPRx, NPCRx are not available on the CAN protocol handler and not implemented on the IP (Door SBC + selective wake-up) are also deleted.
- Removed Section CAN Module Related External Registers as these are not available on IP (Door SBC + selective wake-up) and will not be implemented as there are no further registers to access. WCAN_ADCON0, WCAN_ADL, WCAN_ADH, WCAN_DAT0, WCAN_DAT1, WCAN_DAT2, WCAN_DAT3 are removed.
- Removed Section Message Object Registers and corresponding MOFGPRn, MOIPRn, MOCTRn registers as these are not available on IP (Door SBC + selective wake-up).
- Removed Section CAN Node Registers and corresponding NFCRx, NECNTx, NPCR, NIPRx, NSRx, NCRx as these are not available on IP (Door SBC + selective wake-up).
- Removed Section Global Module Registers and corresponding MSIMASK, MSIDK, MSPNDK, LISTn, MITR, MCR, PANCTR, ID as these are not available on IP (Door SBC + selective wake-up).
- Removed Section on CAN Module Constants, Message Object List Structure, List of Unallocated Elements, List Command Panel, Analyze Mode, CAN Node Analysis Feature, CAN Bit Timing Analysis, Message Postprocessing Interface, Message Object Functionality, Message Object Data Handling, Message Acceptance Filtering, Message Object List Structure as these are not available on IP (Door SBC + selective wake-up).

Revision History (from WCAN v0.6 to WCAN v0.8)

- Replaced [Figure 13-76](#)
- Replaced Fig1-3 WUF detection with [Figure 13-77](#) Wake-up Frame Detection
- Replaced Fig1-4 with [Figure 13-78](#) Oscillator Trimming and modified description of [Section 13.17.1.6](#) Oscillator Trimming.
- Replaced Fig1-5 with [Figure 13-79](#) CAN Timeout and modified description of [Section 13.17.1.7](#) CAN MessageTimeout.
- Replaced Fig 1-6 with [Figure 13-80](#) Error Counter
- Added Section 1.2.2 CAN Protocol Handler Clock
- Rename of registers below and on [Figure 13-84](#)
 - CTRL_DATn (n = 0-3) to CTRLn (n = 0-3)
 - OSCTRIMSTAT_DATn (n = 0-1) to OSCTRIMSTATn (n = 0-1)
 - STAT_DATn (n = 0-2) to STATn (n = 0-2)

- STATCLR_DATn (n = 0-1) to STATCLRn (n = 0-1)
- Added additional register bits below
 - Selective Wake Enable Bit on CTRL0.SELWK_EN.
 - Oscillator Run Interrupt Masking Bit on CTRL1.OSCMAK.
 - Wake-up Frame Interrupt Masking on CTRL1.WUFMASK.
- Added on CTRL2 on XTALOSCCNT0 divider factor is XTALOSCCNT0+1.
- Removed on DIV8=1 information on BTL2_CTRL.BRP and [Section](#)

Revision History (from WCAN v0.8 to WCAN v0.9)

- Added CTRL0.SELWK_EN, selective wake-up enabled bit functions with added text on functions. Modified [Figure 13-77](#), [Figure 13-79](#), [Figure 13-80](#) to include SELWK_EN.
- Added text on CTRL0.SYSERR to indicate configuration error and overcounter overflow condition.
- Added text on STAT0.CANTO to indicate CAN bus inactivity in $t_{SILENCE}$ of 750ms.

Revision History (from WCAN v0.9 to WCAN v1.0)

- Modified BTL1_CTRL reset value to 12_H, $T_{seg1} = 3t_q$, $T_{seg2} = 2t_q$. Thus Bit time = $T_{sync} + T_{seg1} + T_{seg2} = 8t_q$.
- Modified BTL2_CTRL reset value to 18_H, BRP = 24 (decimal). Assume $F_{CAN} = 100\text{MHz}$, default Baud Rate is set to 500 kbits/s after reset.
- Added note on CTRL0.OSCCAL, Oscillator calibration is disabled for the standby domain.
- Removed text SWKSET= CFGVAL of SYSERR under STAT1.SWKSET.
- Added Section 1.2.1.2 Network Propagation Delays
- [Table 13-73](#).

Revision History (from WCAN v1.0 to WCAN v1.1)

- [Table 13-73](#).
- Removed of NIPR2.TEINP as Timer Event Interrupt Node Pointer is not available for CAN.
- Modified BTL1_CTRL reset value to 14_H, $T_{seg1} = 5t_q$, $T_{seg2} = 2t_q$. Thus Bit time = $T_{sync} + T_{seg1} + T_{seg2} = 8t_q$.
- Modified BTL2_CTRL reset value to 58_H. Assume $F_{CAN} = 100\text{MHz}$, default Baud Rate is set to 500Kbits/s after reset.
- Modified STAT0 register reset value to 08_H. At reset STAT0.CANTO value is '1' CAN time out detected.
- Modified STAT1 register reset value to 08_H. At reset STAT1.CANSIL value is '1' $t_{SILENCE}$ is exceeded.

Revision History (from WCAN v1.1 to WCAN v1.2)

- Updated Fig1-2, 1-4, 1-5 addition of “done” complete flowchart
- Corrected text on CTRL0.SELWK_EN. On item 3. Triggers ERROR_INT and wake-up if there are any changes to the message object register.
- Added STAT0.ECOFL, MORC, ERSEL to indicate if SYSERR occurs what is the trigger source. Error Counter Overflow, Message Object Register Changed after SELWK_EN is set, SYSERR is set while SELWK_EN is set. Updated Fig1-5 to include ECOFL='1'
- Added corresponding clear registers STATCLR0.ECOFLCLR, MORCCLR, ERSELCLR.

Revision History (from WCAN v1.2 to WCAN v1.3)

- Updated text on CTRL0.SELWK_EN to indicate source of ERROR_INT.
- Modified on CTRL0.INITREQ register bit 'rwh' to 'rw'

Revision History (from WCAN v1.3 to WCAN v1.4)

- Removed NCR0/1, NSR0/1, NIPR0/1, NIPR0, NECNT0/1/2/3, NFCR0/1/2 register and remove corresponding Table on SFR address listings for Pages 4-7 which housed these registers.
- Added Last Error Code (LEC), 3bits to **INTESTAT1** bit7-5 and moved SYNC bit 6 to bit 1. Also moved SYNCCLR of STAT1 bit 6 to bit 1 correspond to move on **INTESTAT1** register.
- Added hint on [Section Input/Output Function Selection in Ports](#) reference to (See MODPISEL0 register at SCU chapter) for port control selection.
- Rename of registers
 - STAT0/1 to **INTESTAT0/1**, Interrupt and Event Status Register 0/1
 - STAT2 to **FRMERRCNT**, Frame Error Counter Register
 - STATCLR0/1 to **INTESCLR0/1**, Interrupt and Event Status Clear Register 0/1
 - CTRL0 to **CFG**, Configuration Register
 - CTRL1 to **INTMRSLT**, Interrupt Mask and Reference Select Register
 - CTRL2/3 to **REFCLK0/1**, Clock Divider Register 0/1
 - OSCTRIMSTAT0/1 to **CLKCMP0/1**, Clock Compare Result Register 0/1

Revision History (from WCAN v1.4 to WCAN v1.5)

- CEC bit added to **CFG** bit 3. **INTESTAT1**.LEC bit 7-5 changed to 'rh' and bit 111_B option removed.
- Added back NECNT0/1/2/3, NFCR0/1/2 registers and added corresponding Table on “SFR address listings for Pages 4-7” which housed these registers.

Revision History (from WCAN v1.5 to WCAN v1.6)

- Modified on below

- Definition on **CFG.INITREQ**, **INTESTAT0.INITACK**, .
- Added on **FRMERRCNT.ECNTCLR** bit 7, error counter clear bit.
- Added note on **BTL1_CTRL**, **BTL2_CTRL** register that startup software will overwrite the reset values to 14_H, 58_H after reset.
- Corrected typo error on **INTESCLR1.SYNCLR** bit 6 to bit 7.
- Corrected typo error on **CFG** to page 0.
- Modified naming, definition on **REFCLK0/REFCLK1.PERIOD** and on **Section 13.17.1.6 Oscillator Trimming**.

Revision History (from WCAN v1.6 to WCAN v1.7)

- Modified on **NECNT2.EWRNLVL** reset value to 60_H.
- Modified on **INTESCLR1.SWKSETCLR**, bit 2 is removed.
- Corrected on **INTMRSLT** bit 7 to 4 to 'r' reserved bits.
- Modified on **INTESTAT0.CANTO** reset from 08_H to 00_H and on **INTESTAT1.CANSIL** reset from 08_H to 00_H.

Revision History (from WCAN v1.7 to WCAN v1.8)

- Added on **Section** note “ The proposed sample point is at 80% of the bit time. For optimum oscillator tolerance, BTLn(CTRLn[1-2]) setting should be done such that the number of time quantas, tq and SW is maximum.”
- Removed Section 1.2.2 CAN Protocol Handler Clock with Table on Minimum Operating Frequencies.

Revision History (from WCAN v1.8 to WCAN v1.9)

- Added text on **Section**, **Section** “CFG.CCE has to be enabled, before writing to the bit timing registers BTLx_CTR.”

Revision History (from WCAN v1.9 to WCAN v1.10)

- Updated Fig1-3, Fig1-5 to include the 'No' Path.
- Per AI 237961, defined the **INTESTAT0.INITACK** bit, and added section on “Initialisation Request and Initialisation Acknowledge”.

Feature Discontinued

Feature Discontinued

13.18 TC2x_SCR Implementation

This chapter describes the product specific implementation of TC2x_SCR.

13.18.1 TC2x_SCR and QSPI Interface

This section describes the implementation details for TC2x_SCR and QSPI interface. The main TC2x system can access TC2x_SCR XRAM via the QSPI2 interface. **Table 13-79** summarizes the interface signals for SPI communication.

Table 13-79 QSPI interface signals for SPI communication with SCR XRAM

Signal name	Signal Description
SLSO215	SPI chip select output
MRST2H	SPI MRST input
MTRSR2	SPI MTRSR output
SCLK2	SPI clock output

Feature Discontinued

14 General Purpose I/O Ports and Peripheral I/O Lines (Ports)

The TC26x B-Step has digital General Purpose Input/Output (GPIO) port lines which are connected to the on-chip peripheral units.

14.1 Basic Port Operation

Figure 14-1 is a general block diagram of a TC26x B-Step GPIO port slice.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

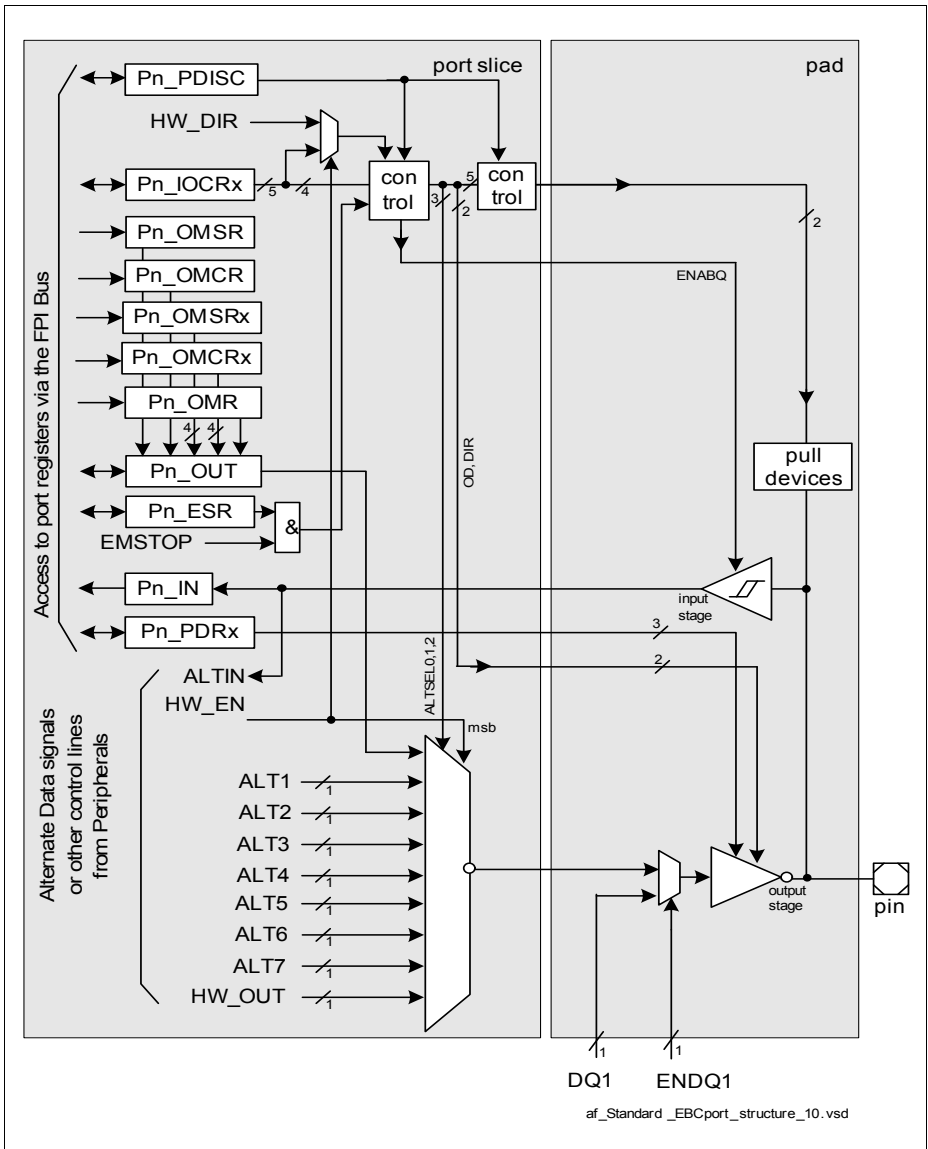


Figure 14-1 General Structure of a Port Pin

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Each port line has a number of control and data bits, enabling very flexible usage of the line. Each port pin can be configured for input or output operation. In input mode, the output driver is switched off (high-impedance). The actual voltage level present at the port pin is translated into a logical 0 or 1 via a Schmitt-Trigger device and can be read via the read-only register Pn_IN. Input signals are connected directly to the various inputs of the peripheral units (AltDataIn). The function of the input line from the pin to the input register Pn_IN and to AltDataIn is independent of whether the port pin operates as input or output. This means that when the port is in output mode, the level of the pin can be read by software via Pn_IN or a peripheral can use the pin level as an input.

In output mode, the output driver is activated and drives the value supplied through the multiplexer to the port pin. Switching between input and output mode is accomplished through the Pn_IOCR register, which enables or disables the output driver. If a peripheral unit uses a GPIO port line as a bi-directional I/O line, register Pn_IOCR has to be written for input or output selection. The Pn_IOCR register further controls the driver type of the output driver, and determines whether an internal weak pull-up, pull-down, or without input pull device is alternatively connected to the pin when used as an input. This offers additional advantages in an application.

The output multiplexer in front of the output driver selects the signal source for the GPIO line when used as output. If the pin is used as general-purpose output, the multiplexer is switched by software (Pn_IOCR register) to the Output Data Register Pn_OUT. Software can set or clear the bit in Pn_OUT through separate Pn_OMSR or Pn_OMCR registers. The set or clear operations for the bits in Pn_OUT can also be done for up to four bits per register in Pn_OMSRx and Pn_OMCRx (x=0,4,8,12). Alternatively, the set, clear or toggle function can be achieved through Pn_OMR, where adjacent pins within the same port can be set, cleared or toggled within one write operation. The manipulation of the control bits in these registers can directly influence the state of the port pin. If the on-chip peripheral units use the pin for output signals, the alternate output lines ALT1 to ALT7 can be switched via the multiplexer to the output driver. The data written into the output register Pn_OUT by software can be used as input data to an on-chip peripheral. This enables, for example, peripheral tests via software without external circuitry.

When selected as general-purpose output line, the logic state of each port pin can be changed individually by programming the pin-related bits in the Output Modification Set Register Pn_OMSR, Output Modification Set Register x Pn_OMSRx (x=0,4,8,12), Output Modification Clear Register Pn_OMCR, Output Modification Clear Register x Pn_OMCRx (x=0,4,8,12) or Output Modification Register, OMR. The bits in Pn_OMSR/Pn_OMSRx and Pn_OMCR/Pn_OMCRx make it possible to set and clear the bits in the Pn_OUT register. While the bits in Pn_OMR allows the bits in Pn_OUT to be set, cleared, toggled or remain unchanged.

When selected as general-purpose output line, the actual logic level at the pin can be examined through reading Pn_IN and compared against the applied output level (either applied through software via the output register Pn_OUT, or via an alternate output function of a peripheral unit). This can be used to detect some electrical failures at the

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

pin caused through external circuitry. In addition, software-supported arbitration schemes can be implemented in this way using the open-drain configuration and an external wired-And circuitry. Collisions on the external communication lines can be detected when a high level (1) is output, but a low level (0) is seen when reading the pin value via the input register Pn_IN.

Most of the GPIO lines of the TC26x B-Step have an emergency stop logic¹⁾. This logic makes it possible to individually disconnect outputs and put them onto a well defined logic state in an emergency case. In an emergency case, the pin is switched to input function with internal pull-up device connected or tri-state. The Emergency Stop Register Pn_ESR determines whether an output is enabled or disabled in an emergency case.

1) This feature is not available for all port lines, for detail please see the dedicated Target Data Sheet / Data Sheet.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.2 Description Scheme for the Port IO Functions

The following two general building block can be used to describe each GPIO pin:

Table 14-1 Port x Input/Output Functions

Port Pin	I/O	Select	Connected Signal(s)	From / to Module
Px.y	Input		Signal(s)	module(s)
	Output	GPIO	Signal	module
		ALT1	Signal	module
		ALT2	Signal	module
		ALT3	Signal	module
		ALT4	Signal	module
		ALT5	Signal	module
		ALT6	Signal	module
	ALT7	Signal	module	
HW_DIR	HW_Out	Signal	module; group En	

or:

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-2 Port x Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
Px.y	I	GPIO	Px_IN.Py	Px_IOCz. PC0	0XXX0 _B	
	O	GPIO		Px_OUT.Py		1X000 _B
						1X001 _B
						1X010 _B
						1X011 _B
						1X100 _B
						1X101 _B
						1X110 _B
				1X111 _B		
HW_DIR	module; group En	Signal		HW_DIR		

- **HW_DIR:**
The type Alternate Direction signal which is needed if HW_En is active:
 - Out -always output
DIRx - the pins in one port having the same DIRx (x=0, 1, 2, ...), are controlled as a group by a dedicated HW_DIR signal.
SDIR- Single DIR- the pin is controlled by its own, dedicated, single HW_DIR signal.
- grouping indicates if the respective pin is controlled by hardware:
 - ENx - the pins in one port having the same ENx (x=0, 1, 2, ...), are controlled as a group by a dedicated HW_EN signal.
 - SEN - Single EN - the pin is controlled by its own, dedicated, single HW_EN signal
- Digital port slices with HW_DIR defined are the ports described in [Figure 14-1](#).

Note: Emergency Stop has higher priority than the HW_EN signal. Emergency Stop is functional when the pins are set in the GPIO mode.

*Note: HW_DIR signal, output case, switches the pad to push-pull output state.
HW_DIR signal, input case, switches the pad to the input state with pull-up/down setting as defined by the IOCR register.*

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.3 Port Register Description

The individual control and data bits of each GPIO port are implemented in a number of registers. The registers are used to configure and use the port as general-purpose I/O or alternate function input/output. Destructive read is not implemented in any of the registers. For some ports, not all registers are implemented. The availability of the registers in the specific ports is described separately.

Note: Parallel requests from on chip bus masters to the ports module are executed sequentially via the on chip bus system. Read-modify-write feature provides an atomic read/write sequence where no other master can access the ports module in between the operations.

Port Register Overview

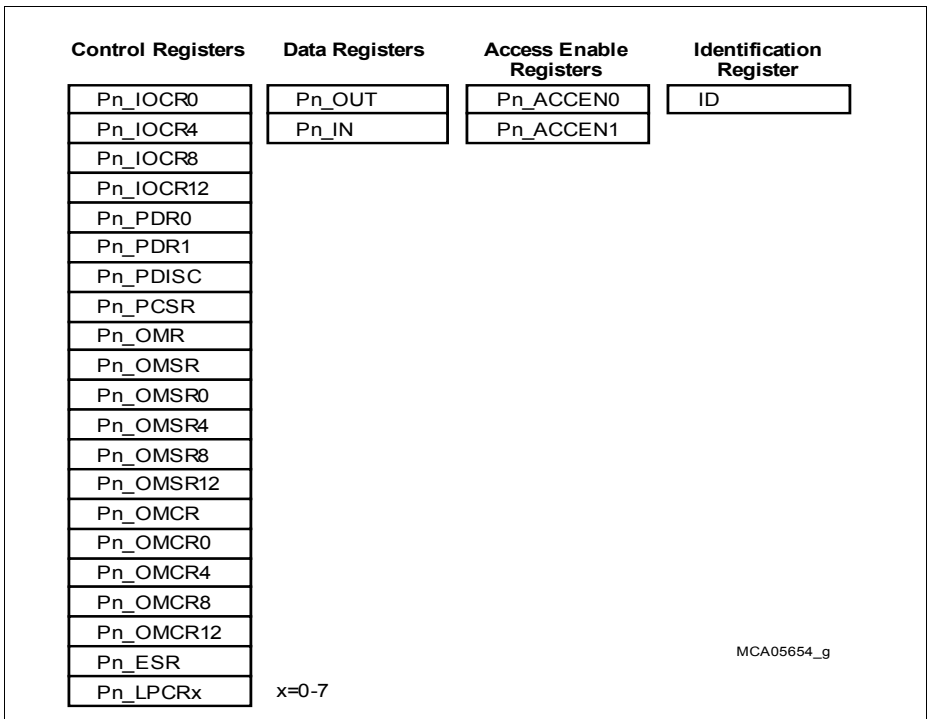


Figure 14-2 Port Registers

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-3 Registers Address Space

Module	Base Address	End Address	Note
P00	F003 A000 _H	F003 A0FF _H	13 pins
P02	F003 A200 _H	F003 A2FF _H	9 pins
P10	F003 B000 _H	F003 B0FF _H	9 pins
P11	F003 B100 _H	F003 B1FF _H	7 pins
P13	F003 B300 _H	F003 B3FF _H	4 pins
P14	F003 B400 _H	F003 B4FF _H	11 pins
P15	F003 B500 _H	F003 B5FF _H	9 pins
P20	F003 C000 _H	F003 C0FF _H	13 pins
P21	F003 C100 _H	F003 C1FF _H	8 pins
P22	F003 C200 _H	F003 C2FF _H	4 pins
P23	F003 C300 _H	F003 C3FF _H	6 pins
P32	F003 D200 _H	F003 D2FF _H	6 pins
P33	F003 D300 _H	F003 D3FF _H	16 pins
P40	F003 E000 _H	F003 E0FF _H	8 pins

Table 14-4 Registers Overview

Register Short Name	Register Long Name	Offset Address	Access Mode		Reset	Desc. see
			Read	Write		
Pn_OUT	Port n Output Register	0000 _H	U, SV	U, SV, P	Application Reset	Page 14-3 6
Pn_OMR	Port n Output Modification Register	0004 _H	U, SV	U, SV, P	Application Reset	Page 14-3 8
ID	Module Identification Register	0008 _H	U, SV	BE	Application Reset	Page 14-1 2
Pn_IOCRO	Port n Input/Output Control Register 0	0010 _H	U, SV	U, SV, P	Application Reset	Page 14-1 3
Pn_IOCRR4	Port n Input/Output Control Register 4	0014 _H	U, SV	U, SV, P	Application Reset	Page 14-1 3

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-4 Registers Overview (cont'd)

Register Short Name	Register Long Name	Offset Address	Access Mode		Reset	Desc. see
			Read	Write		
Pn_IOCR8	Port n Input/Output Control Register 8	0018 _H	U, SV	U, SV, P	Application Reset	Page 14-13
Pn_IOCR12	Port n Input/Output Control Register 12	001C _H	U, SV	U, SV, P	Application Reset	Page 14-13
–	Reserved	0020 _H	BE	BE	–	–
Pn_IN	Port n Input Register	0024 _H	U, SV	BE	Application Reset	Page 14-51
–	Reserved	0028 _H –003C _H	BE	BE	–	–
Pn_PDR0	Port n Pad Driver Mode 0 Register	0040 _H	U, SV	SV, E, P	Application Reset	Page 14-25
Pn_PDR1	Port n Pad Driver Mode 1 Register	0044 _H	U, SV	SV, E, P	Application Reset	Page 14-25
–	Reserved	0048 _H –004C _H	BE	BE	–	–
Pn_ESR	Port n Emergency Stop Register	0050 _H	U, SV	SV, E, P	Application Reset	Page 14-50
–	Reserved	0054 _H –005C _H	BE	BE	–	–
Pn_PDISC	Port n Pin Function Decision Control Register	0060 _H	U, SV	SV, E, P	Application Reset	Page 14-32
Pn_PCSR	Port n Pin Controller Select Register	0064 _H	U, SV	SV, SE	Application Reset	Page 14-33
–	Reserved	0068 _H –006C _H	BE	BE	–	–
Pn_OMSR0	Port n Output Modification Set Register 0	0070 _H	U, SV	U, SV	Application Reset	Page 14-41
Pn_OMSR4	Port n Output Modification Set Register 4	0074 _H	U, SV	U, SV	Application Reset	Page 14-41

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-4 Registers Overview (cont'd)

Register Short Name	Register Long Name	Offset Address	Access Mode		Reset	Desc. see
			Read	Write		
Pn_OMSR8	Port n Output Modification Set Register 8	0078 _H	U, SV	U, SV	Application Reset	Page 14-4 1
Pn_OMSR12	Port n Output Modification Set Register 12	007C _H	U, SV	U, SV	Application Reset	Page 14-4 1
Pn_OMCR0	Port n Output Modification Clear Register 0	0080 _H	U, SV	U, SV	Application Reset	Page 14-4 6
Pn_OMCR4	Port n Output Modification Clear Register 4	0084 _H	U, SV	U, SV	Application Reset	Page 14-4 6
Pn_OMCR8	Port n Output Modification Clear Register 8	0088 _H	U, SV	U, SV	Application Reset	Page 14-4 6
Pn_OMCR12	Port n Output Modification Clear Register 12	008C _H	U, SV	U, SV	Application Reset	Page 14-4 6
Pn_OMSR	Port n Output Modification Set Register	0090 _H	U, SV	U, SV	Application Reset	Page 14-4 0
Pn_OMCR	Port n Output Modification Clear Register	0094 _H	U, SV	U, SV	Application Reset	Page 14-4 5
–	Reserved	0098 _H –009C _H	BE	BE	–	–
Pn_LPCR0	Port n LVDS Pad Control Register 0	00A0 _H	U, SV	SV, E, P	Application Reset	Page 14-2 8
Pn_LPCR1	Port n LVDS Pad Control Register 1	00A4 _H	U, SV	SV, E, P	Application Reset	Page 14-2 8
Pn_LPCR2	Port n LVDS Pad Control Register 2	00A8 _H	U, SV	SV, E, P	Application Reset	Page 14-2 8

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-4 Registers Overview (cont'd)

Register Short Name	Register Long Name	Offset Address	Access Mode		Reset	Desc. see
			Read	Write		
Pn_LPCR3	Port n LVDS Pad Control Register 3	00AC _H	U, SV	SV, E, P	Application Reset	Page 14-2 8
Pn_LPCR4	Port n LVDS Pad Control Register 4	00B0 _H	U, SV	SV, E, P	Application Reset	Page 14-2 8
Pn_LPCR5	Port n LVDS Pad Control Register 5	00B4 _H	U, SV	SV, E, P	Application Reset	Page 14-2 8
Pn_LPCR6	Port n LVDS Pad Control Register 6	00B8 _H	U, SV	SV, E, P	Application Reset	Page 14-2 8
Pn_LPCR7	Port n LVDS Pad Control Register 7	00BC _H	U, SV	SV, E, P	Application Reset	Page 14-2 8
–	Reserved	00C0 _H -00F4 _H	BE	BE	–	–
Pn_ACCEN1	Port n Access Enable Register 1	00F8 _H	U, SV	SV, SE	Application Reset	Page 14-5 4
Pn_ACCEN0	Port n Access Enable Register 0	00FC _H	U, SV	SV, SE	Application Reset	Page 14-5 3

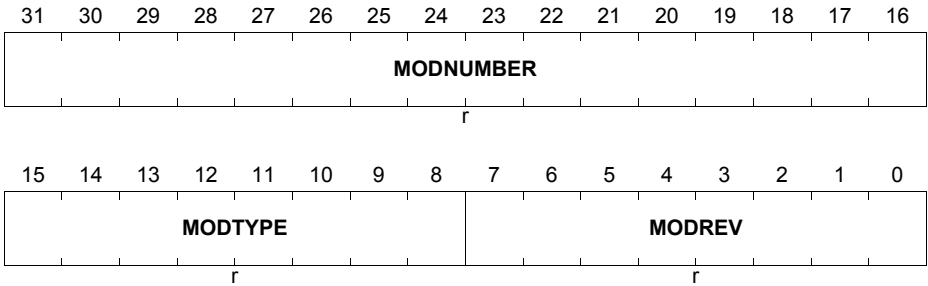
General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.3.1 Module Identification Register

The module Identification Register ID contains read-only information about the module version.

ID

Identification Register (08_H) Reset Value: 00C8 C0XX_H



Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number This bit field indicates the revision number of the TC26x B-Step module (01 _H = first revision).
MODTYPE	[15:8]	r	Module Type This bit field is C0 _H . It defines a 32-bit module
MODNUMBER	[31:16]	r	Module Number This bit field defines the module identification number. The value for the Ports module is 00C8 _H

14.3.2 Port Input/Output Control Registers

The port input/output control registers select the digital output and input driver functionality and characteristics of a GPIO port pin. Port direction (input or output), pull-up, pull-down, or no pull devices for inputs, and push-pull or open-drain functionality for outputs can be selected by the corresponding bit fields PCx (x = 0-15). Each 32-bit wide port input/output control register controls four GPIO port lines:

Register Pn_IOCR0 controls the Pn.[3:0] port lines

Register Pn_IOCR4 controls the Pn.[7:4] port lines

Register Pn_IOCR8 controls the Pn.[11:8] port lines

Register Pn_IOCR12 controls the Pn.[15:12] port lines

The diagrams below show the register layouts of the port input/output control registers with the PCx bit fields. One PCx bit field controls exactly one port line Pn.x.

The reset values of 1010 1010_H and 0000 0000_H for Pn_IOCRx registers represents input pull-up and no input pull device (tri-state mode) being activated, respectively. The switching of the intended mode of the device is controlled by HWCFG6. When a cold reset is activated and HWCFG6=1, the port pins except P33.8 and P40 are set to input pull-up mode, P33.8 and P40 are in tri-state mode as long as PORST is activated. If HWCFG6=0, the pins has the default state of tri-state mode. The pad state can also be configured by software through PMSWCR0.TRISTREQ bit. In the event of a warm reset or wake-up from standby mode, PMSWCR0.TRISTREQ is not affected by reset, hence Pn_IOCRx registers have the reset values configured as per the last state of the TRISTREQ bit.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P00_IOCR0

Port 00 Input/Output Control Register 0

(10_H)

Reset Value: 1010 1010_H

P02_IOCR0

Port 02 Input/Output Control Register 0

(10_H)

Reset Value: 1010 1010_H

Pn_IOCR0 (n=10-11)

Port n Input/Output Control Register 0

(F003 A610_H + n*100_H)

Reset Value: 1010 1010_H

Pn_IOCR0 (n=13-15)

Port n Input/Output Control Register 0

(F003 A610_H + n*100_H)

Reset Value: 1010 1010_H

Pn_IOCR0 (n=21-23)

Port n Input/Output Control Register 0

(F003 AC10_H + n*100_H)

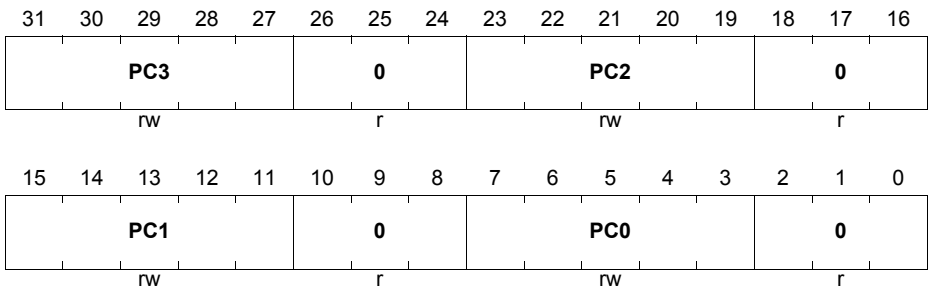
Reset Value: 1010 1010_H

P33_IOCR0

Port 33 Input/Output Control Register 0

(10_H)

Reset Value: 1010 1010_H



Field	Bits	Type	Description
PC0, PC1, PC2, PC3	[7:3], [15:11], [23:19], [31:27]	rw	Port Control for Port n Pin 0 to 3 This bit field determines the Port n line x functionality (x = 0-3) according to the coding table (see Table 14-5).
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P00_IOCR0

Port 00 Input/Output Control Register 0

(10_H)

Reset Value: 0000 0000_H

P02_IOCR0

Port 02 Input/Output Control Register 0

(10_H)

Reset Value: 0000 0000_H

Pn_IOCR0 (n=10-11)

Port n Input/Output Control Register 0

(F003 A610_H + n*100_H)

Reset Value: 0000 0000_H

Pn_IOCR0 (n=13-15)

Port n Input/Output Control Register 0

(F003 A610_H + n*100_H)

Reset Value: 0000 0000_H

Pn_IOCR0 (n=21-23)

Port n Input/Output Control Register 0

(F003 AC10_H + n*100_H)

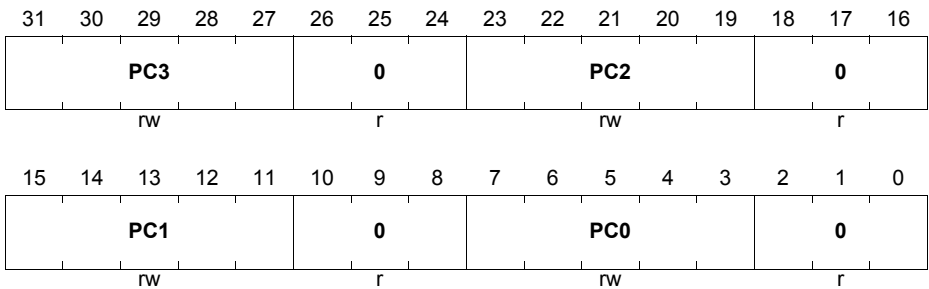
Reset Value: 0000 0000_H

P33_IOCR0

Port 33 Input/Output Control Register 0

(10_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
PC0, PC1, PC2, PC3	[7:3], [15:11], [23:19], [31:27]	rw	Port Control for Port n Pin 0 to 3 This bit field determines the Port n line x functionality (x = 0-3) according to the coding table (see Table 14-5).
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P00_IOCR4

Port 00 Input/Output Control Register 4

(14_H)

Reset Value: 1010 1010_H

P02_IOCR4

Port 02 Input/Output Control Register 4

(14_H)

Reset Value: 1010 1010_H

Pn_IOCR4 (n=10-11)

Port n Input/Output Control Register 4

(F003 A614_H + n*100_H)

Reset Value: 1010 1010_H

Pn_IOCR4 (n=14-15)

Port n Input/Output Control Register 4

(F003 A614_H + n*100_H)

Reset Value: 1010 1010_H

P21_IOCR4

Port 21 Input/Output Control Register 4

(14_H)

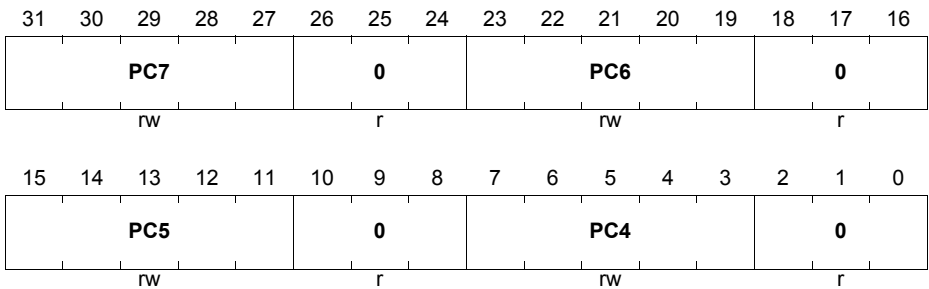
Reset Value: 1010 1010_H

P33_IOCR4

Port 33 Input/Output Control Register 4

(14_H)

Reset Value: 1010 1010_H



Field	Bits	Type	Description
PC4, PC5, PC6, PC7	[7:3], [15:11], [23:19], [31:27]	rw	Port Control for Port n Pin 4 to 7 This bit field determines the Port n line x functionality (x = 4-7) according to the coding table (see Table 14-5).
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P00_IOCR4

Port 00 Input/Output Control Register 4

(14_H)

Reset Value: 0000 0000_H

P02_IOCR4

Port 02 Input/Output Control Register 4

(14_H)

Reset Value: 0000 0000_H

Pn_IOCR4 (n=10-11)

Port n Input/Output Control Register 4

(F003 A614_H + n*100_H)

Reset Value: 0000 0000_H

Pn_IOCR4 (n=14-15)

Port n Input/Output Control Register 4

(F003 A614_H + n*100_H)

Reset Value: 0000 0000_H

P21_IOCR4

Port 21 Input/Output Control Register 4

(14_H)

Reset Value: 0000 0000_H

P33_IOCR4

Port 33 Input/Output Control Register 4

(14_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC7				0				PC6				0			
rw				r				rw				r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC5				0				PC4				0			
rw				r				rw				r			

Field	Bits	Type	Description
PC4, PC5, PC6, PC7	[7:3], [15:11], [23:19], [31:27]	rw	Port Control for Port n Pin 4 to 7 This bit field determines the Port n line x functionality (x = 4-7) according to the coding table (see Table 14-5).
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P00_IOCR8

Port 00 Input/Output Control Register 8

(18_H)

Reset Value: 1010 1010_H

P11_IOCR8

Port 11 Input/Output Control Register 8

(18_H)

Reset Value: 1010 1010_H

P20_IOCR8

Port 20 Input/Output Control Register 8

(18_H)

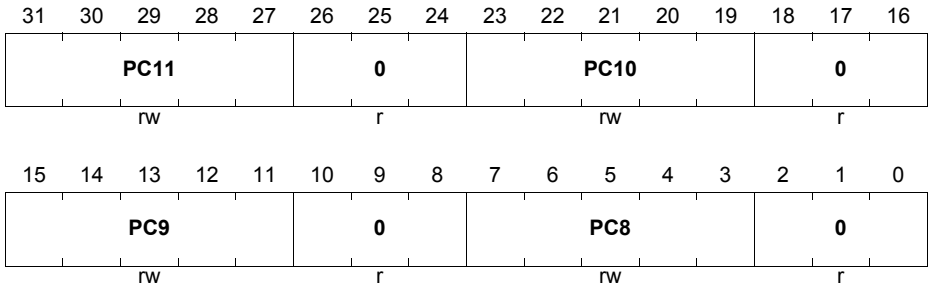
Reset Value: 1010 1010_H

P33_IOCR8

Port 33 Input/Output Control Register 8

(18_H)

Reset Value: 1010 1000_H



Field	Bits	Type	Description
PC8, PC9, PC10, PC11	[7:3], [15:11], [23:19], [31:27]	rw	Port Control for Port n Pin 8 to 11 This bit field determines the Port n line x functionality (x = 8-11) according to the coding table (see Table 14-5).
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P00_IOCR8

Port 00 Input/Output Control Register 8

(18_H)

Reset Value: 0000 0000_H

P11_IOCR8

Port 11 Input/Output Control Register 8

(18_H)

Reset Value: 0000 0000_H

P20_IOCR8

Port 20 Input/Output Control Register 8

(18_H)

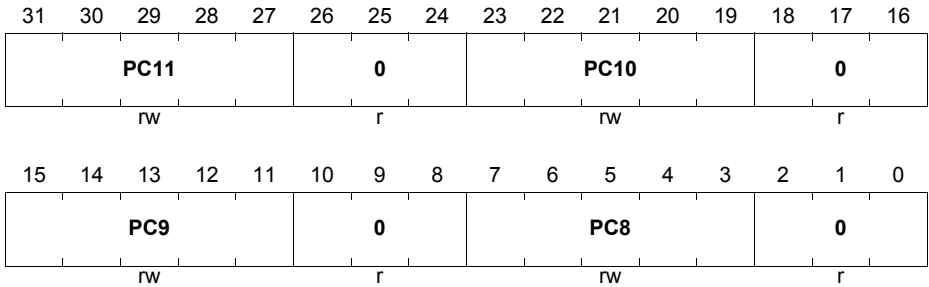
Reset Value: 0000 0000_H

P33_IOCR8

Port 33 Input/Output Control Register 8

(18_H)

Reset Value: 0000 0000_H

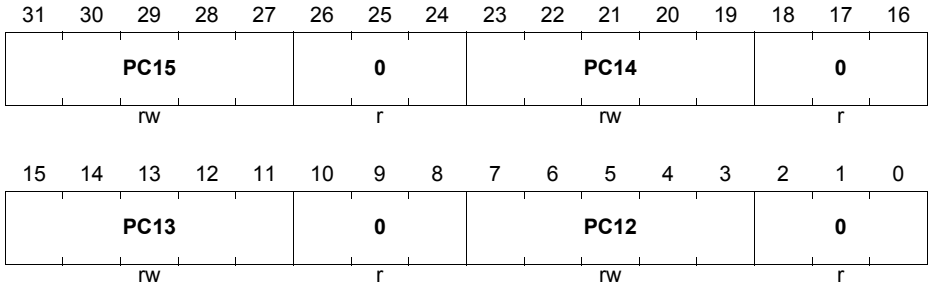


Field	Bits	Type	Description
PC8, PC9, PC10, PC11	[7:3], [15:11], [23:19], [31:27]	rw	Port Control for Port n Pin 8 to 11 This bit field determines the Port n line x functionality (x = 8-11) according to the coding table (see Table 14-5).
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P33_IOCR12
Port n Input/Output Control Register 12

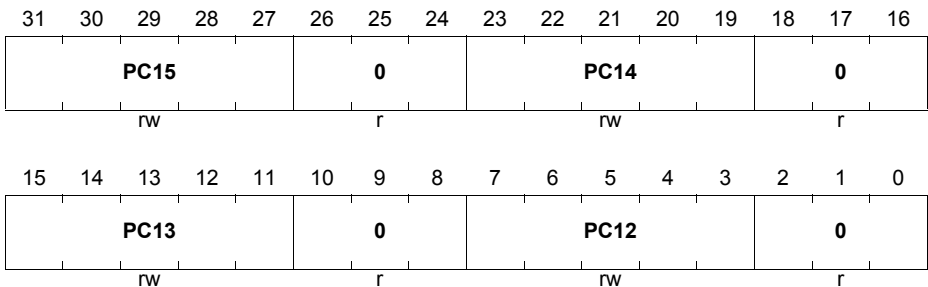
 (1C_H)

 Reset Value: 1010 1010_H


Field	Bits	Type	Description
PC12, PC13, PC14, PC15	[7:3], [15:11], [23:19], [31:27]	rw	Port Control for Port n Pin 12 to 15 This bit field determines the Port n line x functionality (x = 12-15) according to the coding table (see Table 14-5).
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

P33_IOCR12
Port n Input/Output Control Register 12

 (1C_H)

 Reset Value: 0000 0000_H


General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PC12, PC13, PC14, PC15	[7:3], [15:11], [23:19], [31:27]	rw	Port Control for Port n Pin 12 to 15 This bit field determines the Port n line x functionality (x = 12-15) according to the coding table (see Table 14-5).
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

The structure with one control bit field for each port pin located in different register bytes offers the possibility to configure the port pin functionality of a single pin with byte-oriented accesses without accessing the other PCx bit fields.

Port Control Coding

Table 14-5 describes the coding of the PCx bit fields that determine the port line functionality.

When a port line is configured as input (PCx), its hysteresis function can be activated/inactivated via its related PDx bit field (see **Table 14-6** and **Table 14-7**).

When a port line is configured as output (PCx), its speed grade can be configured via its related PDx bit field (see **Table 14-6** and **Table 14-7**).

Table 14-5 PCx Coding

PCx[4:0]	I/O	Characteristics	Selected Pull-up / Pull-down / Selected Output Function
0XX00 _B	Input	–	No input pull device connected, tri-state mode
0XX01 _B			Input pull-down device connected
0XX10 _B			Input pull-up device connected ¹⁾
0XX11 _B			No input pull device connected, tri-state mode

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Table 14-5 PCx Coding

PCx[4:0]	I/O	Characteristics	Selected Pull-up / Pull-down / Selected Output Function
10000 _B	Output	Push-pull	General-purpose output
10001 _B			Alternate output function 1
10010 _B			Alternate output function 2
10011 _B			Alternate output function 3
10100 _B			Alternate output function 4
10101 _B			Alternate output function 5
10110 _B			Alternate output function 6
10111 _B			Alternate output function 7
11000 _B		Open-drain	General-purpose output
11001 _B			Alternate output function 1
11010 _B			Alternate output function 2
11011 _B			Alternate output function 3
11100 _B			Alternate output function 4
11101 _B			Alternate output function 5
11110 _B			Alternate output function 6
11111 _B			Alternate output function 7

1) This is the default pull device setting after reset for powertrain applications.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.3.3 Pad Driver Mode Register
Overview

For port lines that are configured as output (PCx), the speed grade can be configured via the related PDx bit fields (registers Pn_PDR0/1). The speed grade defines the output driver strength and the slew rate of the faster output slopes (slope control only available for speed grades 1 and 2). For port lines configured as input (PCx), the PDx fields determines if hysteresis is active or inactive for the input function (hysteresis function is only available for MP, MP+, MPR and LP pads).

Further details about pad driver classes that are available in the TC26x B-Step are summarized in the Target Data Sheet/Data Sheet.

Table 14-6 Pad Driver Mode and Hysteresis Selection for Pads of type MP, MP+ MPR

PDx.2	PDx.1	PDx.0	Output Functionality	Input Functionality
X	0	0	Speed grade 1	Hysteresis is inactive
X	0	1	Speed grade 2	Hysteresis is active
X	1	0	Speed grade 3	Hysteresis is active
X	1	1	Speed grade 4	Hysteresis is active

Table 14-7 Pad Driver Mode and Hysteresis Selection for Pads of type LP

PDx.2	PDx.1	PDx.0	Output Functionality	Input Functionality
X	X	0	Speed grade 1	Hysteresis is inactive ¹⁾
X	X	1	Speed grade 2	Hysteresis is active

1) This feature is not available for all LP pads, for detail please see the dedicated Target Data Sheet / Data Sheet.

Note: TC26x B-Step Data Sheet describes the DC characteristics of all pad classes.

The default CMOS mode can be switched to LVDS mode in LVDS pads. For LVDSM pads, the LVDS mode can be enabled through port control by setting PDx.2 of the lower pin, CMOS mode is selected if PDx.2 of the lower pin of the pin-pair is cleared. For LVDSH pads, the LVDS mode can be enabled through LPCRx register.

Depending on whether it is 3.3V- or 5V-pad, the pad level is Automotive Level (AL). The TTL level can be enabled for both for 3.3V and 5V pads through the PLx bit in the Pn_PDRx. The coding is shown in [Table 14-8](#).

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-8 Pad Level Selection

PLx	3.3V-Pad	5V-Pad
0	Automotive Level (AL)	Automotive Level (AL)
1	TTL Level	TTL Level

Note: 3.3V-Pad corresponds to A2, F pads and 5V-Pad corresponds to LP, MP and MP+ pads in the Data Sheet, respectively.

Pad Driver Mode Registers

This is the general description of the PDR registers. Each port contains its own specific PDR registers, described additionally at each port, that can contain between one and eight PDx fields for PDR0 and PDR1 registers, respectively. Each PDx field controls 1 pin. For coding of PDx, see [Table 14-6](#). Similarly, each PLx bit controls 1 pin. For coding of PLx, see [Table 14-8](#).

The reset value of Pn_PDR0/1 registers is 3333 3333_H.

P00_PDR0

Port 00 Pad Driver Mode 0 Register (40_H) Reset Value: 3333 3333_H

P02_PDR0

Port 02 Pad Driver Mode 0 Register (40_H) Reset Value: 3333 3333_H

P10_PDR0

Port 10 Pad Driver Mode 0 Register (40_H) Reset Value: 3333 3333_H

Pn_PDR0 (n=14-15)

Port n Pad Driver Mode 0 Register(F003 A640_H+n*100_H) Reset Value: 3333 3333_H

P21_PDR0

Port 21 Pad Driver Mode 0 Register (40_H) Reset Value: 3333 3333_H

P33_PDR0

Port 33 Pad Driver Mode 0 Register (40_H) Reset Value: 3333 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL7		PD7		PL6		PD6		PL5		PD5		PL4		PD4	
r/w		r/w		r/w		r/w		r/w		r/w		r/w		r/w	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL3		PD3		PL2		PD2		PL1		PD1		PL0		PD0	
r/w		r/w		r/w		r/w		r/w		r/w		r/w		r/w	

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PD0, PD1, PD2, PD3, PD4, PD5, PD6, PD7	[2:0], [6:4], [10:8], [14:12], [18:16], [22:20], [26:24], [30:28]	rw	Pad Driver Mode for Port n Pin 0 to 7
PL0, PL1, PL2, PL3, PL4, PL5, PL6, PL7	3, 7, 11, 15, 19, 23, 27, 31	rw	Pad Level Selection for Port n Pin 0 to 7

P33_PDR1

Port 3 Pad Driver Mode 1 Register

(44_H)

Reset Value: 3333 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL15		PD15		PL14		PD14		PL13		PD13		PL12		PD12	
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL11		PD11		PL10		PD10		PL9		PD9		PL8		PD8	
rw		rw		rw		rw		rw		rw		rw		rw	

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PD8, PD9, PD10, PD11, PD12, PD13, PD14, PD15	[2:0], [6:4], [10:8], [14:12], [18:16], [22:20], [26:24], [30:28]	rw	Pad Driver Mode for Port n Pin 8 to 15
PL8, PL9, PL10, PL11, PL12, PL13, PL14, PL15	3, 7, 11, 15, 19, 23, 27, 31	rw	Pad Level Selection for Port n Pin 8 to 15

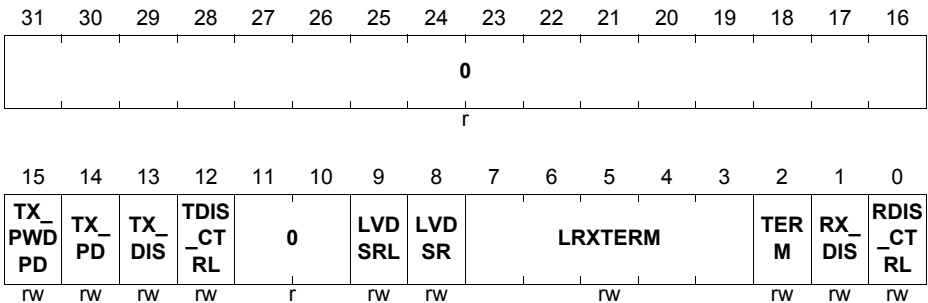
General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.3.4 LVDS Pad Control Register

The LVDS Pad Control Register controls the RX or TX functions of the LVDSH pads. For usage of RX pad, bit field [9:0] are applicable. If used for TX pad, bit field [15:12] apply. Register P21_LPCR_x (x=0-7) controls the LVDSH functions of P21._[(x*2+1):x*2] port lines.

Alternatively, the LVDS Pad Control Register also selects if the LVDSM pads are supplied by 5V or 3.3V at V_{EXT} .

Register P_x_LPCR0.1,(x=13, 22) selects the pad supply for P_x._[1:0] port lines.

Register P_x_LPCR1.1,(x=13, 22) selects the pad supply for P_x._[3:2] port lines.

P21_LPCR_x (x = 0-7)
Port 21 LVDS Pad Control Register x(00A0_H+x*0004_H) Reset Value: 0000 6046_H


Field	Bits	Type	Description
RDIS_CTRL	0	rw	LVDS RX_DIS controller The LVDS RX_DIS control function can be selected from the Port (default) or HSCT module. 0 _B Port controlled 1 _B HSCT controlled
RX_DIS	1	rw	Disable Receive LVDS Disable the receive LVDS / enable CMOS path. If this bit is set to 1 - no transfer from the LVDS sender can be received and the receiver LVDS is in low power state. 0 _B enable LVDS / disable CMOS mode 1 _B disable LVDS / enable CMOS mode

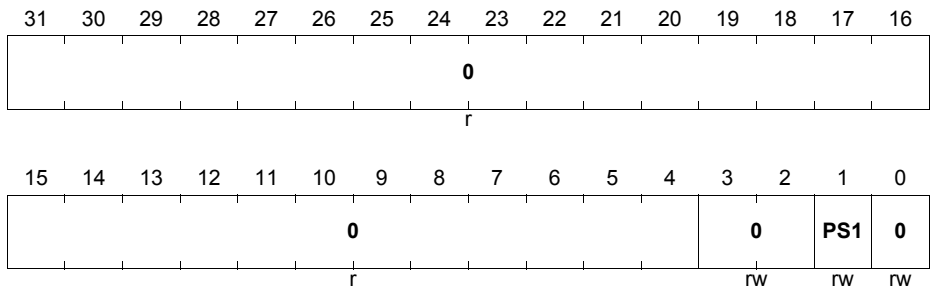
General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
TERM	2	rw	Select Receiver Termination Mode Selects a suitable internal load capacitance. 0 _B external termination - on the PCB 1 _B 100 Ω Receiver internal termination
LRXTERM	[7:3]	rw	LVDS RX Poly-resistor configuration value Programming bits for the on die poly resistor termination. The value is configured during production test. Each chip configuration on this bit field is unique and configured during production testing. <i>Note: The configuration value shall not be changed by user after start-up for a guaranteed behavior.</i>
LVDSR	8	rw	Special reduced LVDS electrical signaling mode LVDSR = 1 and LVDSRL = 0 activates LVDSR electrical. LVDSR = 1 and LVDSRL = 1 Not allowed - Software must prevent this situation! <i>Note: LVDSR = 1 and LVDSRL = 1 the special reduced electrical LVDS signaling is enabled.</i>
LVDSRL	9	rw	LVDS IEEE electrical signaling mode LVDSR = 0 and LVDSRL = 1 activates the IEEE reduced LVDS link electrical signaling specification LVDSR = 0 and LVDSRL = 0 activates the IEEE general purpose link.
TDIS_CTRL	12	rw	LVDS TX_DIS controller The LVDS TX_DIS control function can be selected from the Port (default) or HSCT module. 0 _B Port controlled 1 _B HSCT controlled
TX_DIS	13	rw	Disable Transmit LVDS Disable the transmit LVDS / enable CMOS path. If this bit is set to 1 - no transfer on LVDS data path can be initiated and the LVDS driver is disabled. 0 _B enable LVDS / disable CMOS mode 1 _B disable LVDS / enable CMOS mode

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
TX_PD	14	rw	LVDS Power Down 0 _B LVDS power on 1 _B LVDS power down (default)
TX_PWDPD	15	rw	Disable TX Power down pull down. This function disables or enables the LVDS pull down resistor. The application code must disable TX power down pull down resistor with a power up. With a LVDS Power Down configuration the pull down function must be enabled, if required. 0 _B disabled TX Power down pull down resistor. 1 _B enabled TX Power down pull down resistor.
0	[31:16], [11:10]	r	Reserved Read as 0; should be written with 0.

For P13_LPCR_x (x=0,1) and P22_LPCR_x (x=0,1) , the PS_x needs to be configured to the intended pad supply level before the LVDSM function is enabled.

P13_LPCR0
Port 13 LVDS Pad Control Register 0 (A0_H)
Reset Value: 0000 0000_H
P22_LPCR0
Port 22 LVDS Pad Control Register 0 (A0_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
PS1	1	rw	Pad Supply for pins [1:0] Selects between 5V or 3.3V supply on V _{EXT} for the LVDSM pad-pair. 0 _B 5V supply 1 _B 3.3V supply

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
0	[3:2], 0	rw	Reserved Read as 0; must be written with 0.
0	[31:4]	r	Reserved Read as 0; should be written with 0.

P13_LPCR1

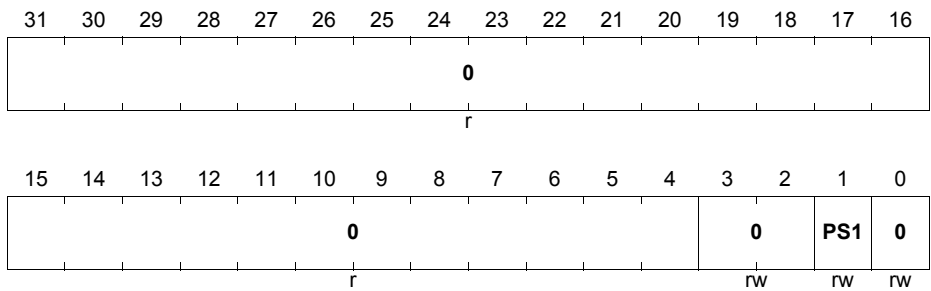
Port 13 LVDS Pad Control Register 1 (A4_H)

Reset Value: 0000 0000_H

P22_LPCR1

Port 22 LVDS Pad Control Register 1 (A4_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
PS1	1	rw	Pad Supply for pins [3:2] Selects between 5V or 3.3V supply on V_{EXT} for the LVDSM pad-pair. 0 _B 5V supply 1 _B 3.3V supply
0	[3:2], 0	rw	Reserved Read as 0; must be written with 0.
0	[31:4]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

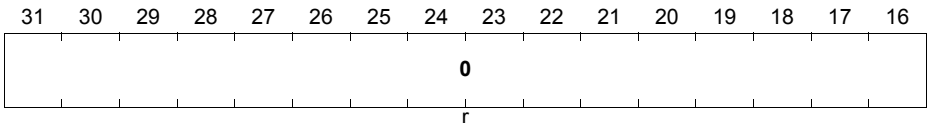
14.3.5 Pin Function Decision Control Register

Pin Function Decision Control Register

The pad structure of the TC26x B-Step GPIO lines offers the possibility to select digital input or analog ADC input functionalities. For the selection of digital input, the parameters defined for Class S pads must be met. This feature can be controlled by individual bits in the Pn_PDISC register, independently from input/output and pull-up/pull-down control functionality as programmed in the Pn_IOCRR register. One Pn_PDISC register is assigned to each port.

P40_PDISC

Port 40 Pin Function Decision Control Register(60_H) Reset Value: 0000 03CF_H



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDIS	PDIS	PDIS	PDIS	PDIS	PDIS	PDIS	PDIS	PDIS	PDIS	PDIS	PDIS	PDIS	PDIS	PDIS	PDIS
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
PDISx (x = 0-15)	x	rW	Pin Function Decision Control for Pin x This bit selects the function of the port pad. 0 _B Pad Pn.x is selected for function X. 1 _B Pad Pn.x is selected for function Y.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.3.6 Pin Controller Select Register

The pin controller select register is capable to decide the controller for all data and control functions of each shared port pin separately during run, sleep and standby modes. The 8-bit controller can handle the functions for the shared port pins in standby mode, if the controller is selected before entering the standby mode. Upon a wake-up from standby mode, the reset value does not reflect the actual latched value stored in the 8-bit controller. The value written into the PCSR before entering standby mode is kept by the 8-bit controller when standby mode is exited. The value is not available as status information.

PCSR register is also used to enable or disable the VADC Multiplexer Diagnostics (MD) feature. For pins selected for 8-bit controller, SPAREINOUT0 bit 0 decides if pull-up and pull-down function is controlled by 8-bit controller or VADC Multiplexer Diagnostics (MD) feature is enabled.

Alternatively, this register can also select Ethernet output signals through ports alternate output or fast RMII mode.

P00_PCSR

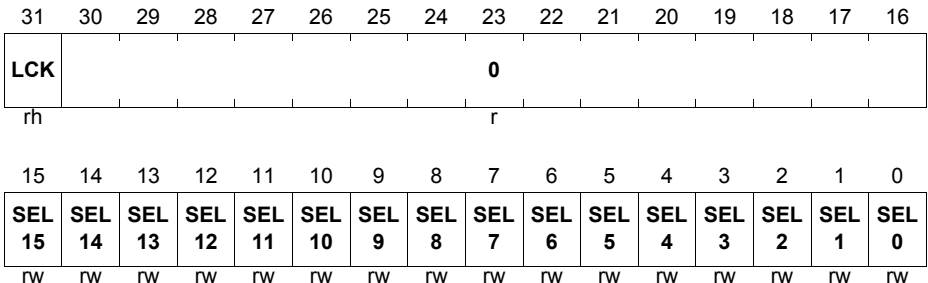
Port 00 Pin Controller Select Register (64_H)

Reset Value: 0000 0000_H

P02_PCSR

Port 02 Pin Controller Select Register (64_H)

Reset Value: 0000 0000_H



General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
SELx (x = 0-15)	x	rw	Pin Controller Select for Pin x This bit selects the controller in all data and control functions of the port pin. If applicable, this bit along with SPAREINOUT0 bit 0 can also enable or disable the VADC Multiplexer Diagnostics (MD) feature for pin x . 0 _B 32-bit controller is selected for data and control of pin x. 1 _B 8-bit controller is selected for data and control functions, pull-up or pull-down control is controlled by 8-bit controller or (if applicable) VADC Multiplexer Diagnostics (MD) feature is enabled for pin x.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus has no effect. 0 _B The register is unlocked and can be updated. 1 _B The register is locked and can not be updated.
0	[30:16]	r	Reserved Read as 0; should be written with 0.

P40_PCSR
Port 40 Pin Controller Select Register (64_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCK								0							
rh								r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
SELx (x = 0-15)	x	rw	Pin Controller Select for Pin x This bit enables or disables the VADC Multiplexer Diagnostics (MD) feature. 0 _B Disable VADC MD feature of pin x. 1 _B Enable VADC MD feature of pin x.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus has no effect. 0 _B The register is unlocked and can be updated. 1 _B The register is locked and can not be updated.
0	[30:16]	r	Reserved Read as 0; should be written with 0.

P11_PCSR
Port 11 Pin Controller Select Register (64_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL 15	SEL 14	SEL 13	SEL 12	SEL 11	SEL 10	SEL 9	SEL 8	SEL 7	SEL 6	SEL 5	SEL 4	SEL 3	SEL 2	SEL 1	SEL 0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SELx (x = 0-15)	x	rw	Pin Controller Select for Pin x This bit selects the Ethernet output signal through ports alternate output or fast RMII mode. 0 _B Ethernet output via ports alternate output of pin x. 1 _B Ethernet output via fast RMII mode of pin x.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.3.7 Port Output Register

The port output register determines the value of a GPIO pin when it is selected by Pn_IOC Rx as output. Writing a 0 to a Pn_OUT.Px (x = 0-15) bit position delivers a low level at the corresponding output pin. A high level is output when the corresponding bit is written with a 1. Note that the bits of Pn_OUT.Px can be individually set or cleared by writing appropriate values into the port output modification set register Pn_OMSR or port output modification clear register Pn_OMCR, respectively. The Pn_OUT.Px bits can also be set, cleared or toggled with register Pn_OMR within the same write operation.

P00_OUT

Port 00 Output Register (00_H) Reset Value: 0000 0000_H

P02_OUT

Port 02 Output Register (00_H) Reset Value: 0000 0000_H

Pn_OUT (n=10-11)

Port n Output Register (F003 A600_H + n*100_H) Reset Value: 0000 0000_H

Pn_OUT (n=13-15)

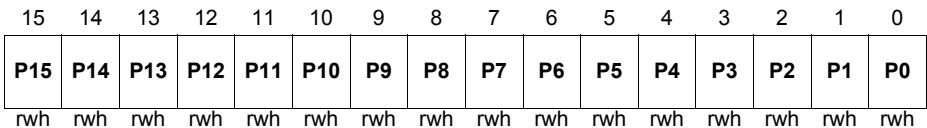
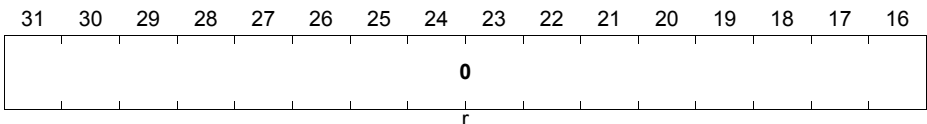
Port n Output Register (F003 A600_H + n*100_H) Reset Value: 0000 0000_H

Pn_OUT (n=20-23)

Port n Output Register (F003 AC00_H + n*100_H) Reset Value: 0000 0000_H

Pn_OUT (n=32-33)

Port n Output Register (F003 B200_H + n*100_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
Px (x = 0-15)	x	rwh	<p>Port n Output Bit x</p> <p>This bit determines the level at the output pin Pn.x if the output is selected as GPIO output.</p> <p>0_B The output level of Pn.x is 0.</p> <p>1_B The output level of Pn.x is 1.</p> <p>Pn.x can also be set or cleared by control bits of the Pn_OMSR, Pn_OMCR or Pn_OMR registers.</p>

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
0	[31:16]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.3.8 Port Output Modification Register

The port output modification register contains control bits that make it possible to individually set, clear or toggle the logic state of a single port line by manipulating the output register.

P00_OMR
Port 00 Output Modification Register (04_H)
Reset Value: 0000 0000_H
P02_OMR
Port 02 Output Modification Register (04_H)
Reset Value: 0000 0000_H
Pn_OMR (n=10-11)
Port n Output Modification Register (F003 A604_H + n*100_H)
Reset Value:
0000 0000_H
Pn_OMR (n=13-15)
Port n Output Modification Register (F003 A604_H + n*100_H)
Reset Value:
0000 0000_H
Pn_OMR (n=20-23)
Port n Output Modification Register (F003 AC04_H + n*100_H)
Reset Value:
0000 0000_H
Pn_OMR (n=32-33)
Port n Output Modification Register (F003 B204_H + n*100_H)
Reset Value:
0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PCL	PCL	PCL	PCL	PCL	PCL	PCL	PCL	PCL	PCL	PCL	PCL	PCL	PCL	PCL	PCL
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS	PS	PS	PS	PS	PS	PS	PS	PS	PS	PS	PS	PS	PS	PS	PS
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
PSx (x = 0-15)	x	w	Port n Set Bit x Setting this bit will set or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. The function of this bit is shown in Table 14-9 . 0 _B No operation 1 _B Sets or toggles Pn_OUT.Px.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PCLx (x = 0-15)	x + 16	w	Port n Clear Bit x Setting this bit will clear or toggle the corresponding bit in the port output register Pn_OUT. Read as 0. The function of this bit is shown in Table 14-9 . 0 _B No operation 1 _B Clears or toggles Pn_OUT.Px.

Table 14-9 Function of the Bits PCLx and PSx

PCLx	PSx	Function
0	0	Bit Pn_OUT.Px is not changed.
0	1	Bit Pn_OUT.Px is set.
1	0	Bit Pn_OUT.Px is reset.
1	1	Bit Pn_OUT.Px is toggled.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.3.9 Port Output Modification Set Register

The port output modification set register contains control bits that make it possible to individually set the logic state of a single port line by manipulating the output register.

P00_OMSR

Port 00 Output Modification Set Register (90_H) Reset Value: 0000 0000_H

P02_OMSR

Port 02 Output Modification Set Register (90_H) Reset Value: 0000 0000_H

Pn_OMSR (n=10-11)

Port n Output Modification Set Register (F003 A690_H + n*100_H) Reset Value: 0000 0000_H

Pn_OMSR (n=13-15)

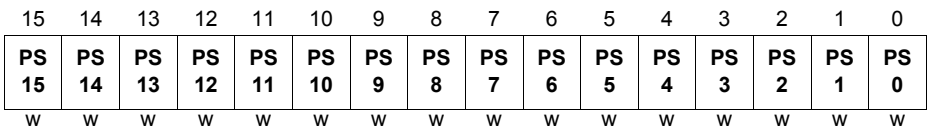
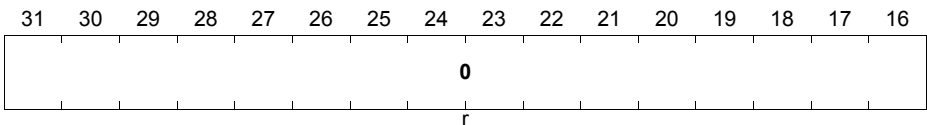
Port n Output Modification Set Register (F003 A690_H + n*100_H) Reset Value: 0000 0000_H

Pn_OMSR (n=20-23)

Port n Output Modification Set Register (F003 AC90_H + n*100_H) Reset Value: 0000 0000_H

Pn_OMSR (n=32-33)

Port n Output Modification Set Register (F003 B290_H + n*100_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
PSx (x = 0-15)	x	w	Port n Set Bit x Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0 _B No operation 1 _B Sets Pn_OUT.Px
0	[31:16]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.3.10 Port Output Modification Set Registers x

The port output modification set register x, (x = 0, 4, 8, 12) contains control bits to individually set the logic state of a single port line by manipulating the output register.

Register Pn_OMSR0 sets the logic state of Pn.[3:0] port lines

Register Pn_OMSR4 sets the logic state of Pn.[7:4] port lines

Register Pn_OMSR8 sets the logic state of Pn.[11:8] port lines

Register Pn_OMSR12 sets the logic state of Pn.[15:12] port lines

P00_OMSR0

Port 00 Output Modification Set Register 0(70_H) **Reset Value: 0000 0000_H**

P02_OMSR0

Port 02 Output Modification Set Register 0(70_H) **Reset Value: 0000 0000_H**

Pn_OMSR0 (n=10-11)

Port n Output Modification Set Register 0(F003 A670_H + n*100_H) **Reset Value: 0000 0000_H**

Pn_OMSR0 (n=13-15)

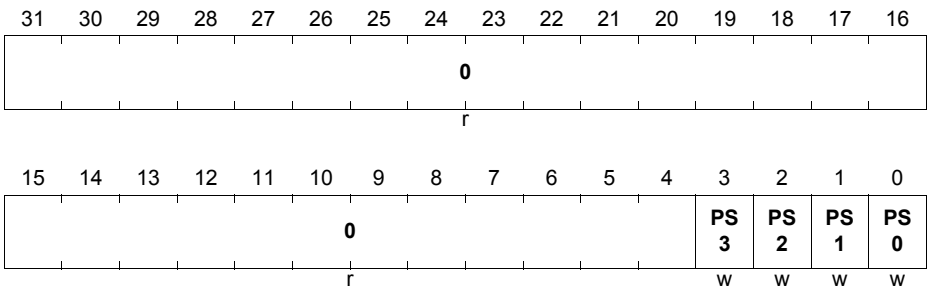
Port n Output Modification Set Register 0(F003 A670_H + n*100_H) **Reset Value: 0000 0000_H**

Pn_OMSR0 (n=20-23)

Port n Output Modification Set Register 0(F003 AC70_H + n*100_H) **Reset Value: 0000 0000_H**

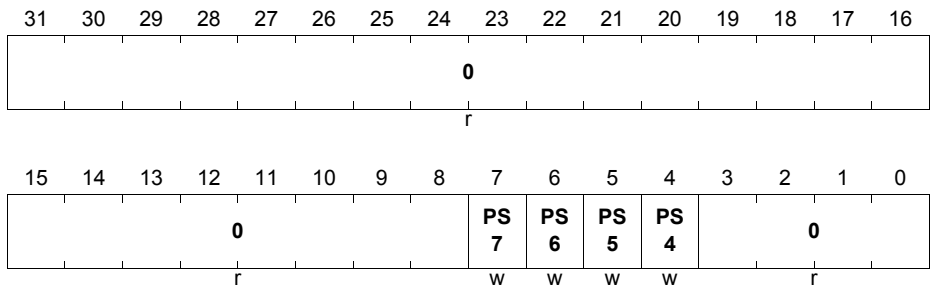
Pn_OMSR0 (n=32-33)

Port n Output Modification Set Register 0(F003 B270_H + n*100_H) **Reset Value: 0000 0000_H**



General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PSx (x = 0-3)	x	w	Port n Set Bit x Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0 _B No operation 1 _B Sets Pn_OUT.Px
0	[31:4]	r	Reserved Read as 0; should be written with 0.

P00_OMSR4
Port 00 Output Modification Set Register 4(74_H) **Reset Value: 0000 0000_H**
P02_OMSR4
Port 02 Output Modification Set Register 4(74_H) **Reset Value: 0000 0000_H**
Pn_OMSR4 (n=10-11)
Port n Output Modification Set Register 4(F003 A674_H + n*100_H) **Reset Value: 0000 0000_H**
Pn_OMSR4 (n=14-15)
Port n Output Modification Set Register 4(F003 A674_H + n*100_H) **Reset Value: 0000 0000_H**
Pn_OMSR4 (n=20-21)
Port n Output Modification Set Register 4(F003 AC74_H + n*100_H) **Reset Value: 0000 0000_H**
P32_OMSR4
Port 32 Output Modification Set Register 4(74_H) **Reset Value: 0000 0000_H**
P33_OMSR4
Port 33 Output Modification Set Register 4(74_H) **Reset Value: 0000 0000_H**


General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PSx (x = 7-4)	x	w	Port n Set Bit x Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0 _B No operation 1 _B Sets Pn_OUT.Px
0	[31:8], [3:0]	r	Reserved Read as 0; should be written with 0.

P00_OMSR8

Port 00 Output Modification Set Register 8(78_H) **Reset Value: 0000 0000_H**

P02_OMSR8

Port 02 Output Modification Set Register 8(78_H) **Reset Value: 0000 0000_H**

Pn_OMSR8 (n=10-11)

Port n Output Modification Set Register 8(F003 A678_H + n*100_H) **Reset Value: 0000 0000_H**

Pn_OMSR8 (n=14-15)

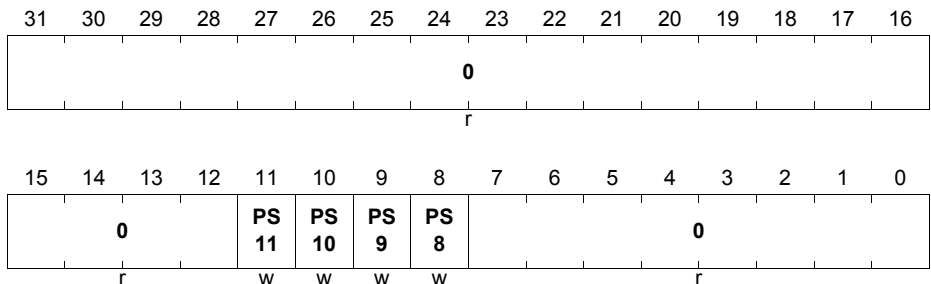
Port n Output Modification Set Register 8(F003 A678_H + n*100_H) **Reset Value: 0000 0000_H**

P20_OMSR8

Port 20 Output Modification Set Register 8(78_H) **Reset Value: 0000 0000_H**

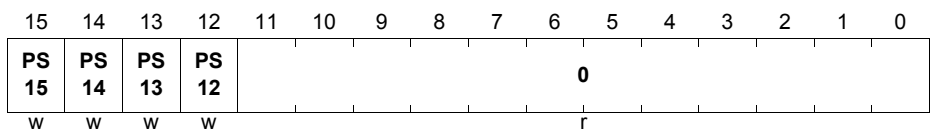
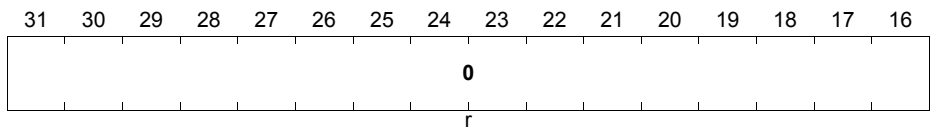
P33_OMSR8

Port 33 Output Modification Set Register 8(78_H) **Reset Value: 0000 0000_H**



General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PSx (x = 11-8)	x	w	Port n Set Bit x Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0 _B No operation 1 _B Sets Pn_OUT.Px
0	[31:12], [7:0]	r	Reserved Read as 0; should be written with 0.

P00_OMSR12
Port 00 Output Modification Set Register 12(7C_H) **Reset Value: 0000 0000_H**
P11_OMSR12
Port 11 Output Modification Set Register 12(7C_H) **Reset Value: 0000 0000_H**
P20_OMSR12
Port 20 Output Modification Set Register 12(7C_H) **Reset Value: 0000 0000_H**
P33_OMSR12
Port 33 Output Modification Set Register 12(7C_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
PSx (x = 15-12)	x	w	Port n Set Bit x Setting this bit will set the corresponding bit in the port output register Pn_OUT. Read as 0. 0 _B No operation 1 _B Sets Pn_OUT.Px
0	[31:16], [11:0]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.3.11 Port Output Modification Clear Register

The port output modification clear register contains control bits that make it possible to individually clear the logic state of a single port line by manipulating the output register.

P00_OMCR

Port 00 Output Modification Clear Register (94_H) Reset Value: 0000 0000_H

P02_OMCR

Port 02 Output Modification Clear Register (94_H) Reset Value: 0000 0000_H

Pn_OMCR (n=10-11)

Port n Output Modification Clear Register (F003 A694_H + n*100_H) Reset Value: 0000 0000_H

Pn_OMCR (n=13-15)

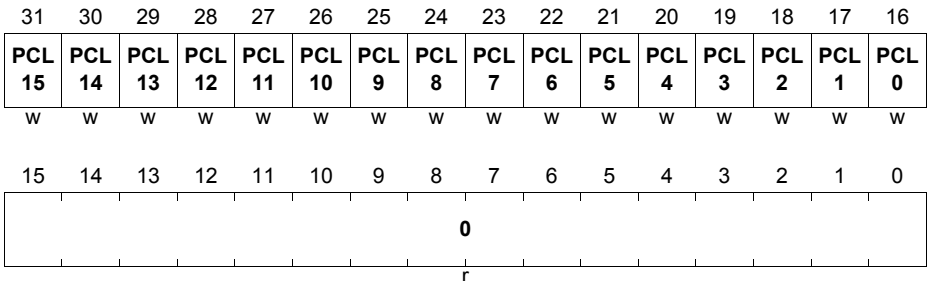
Port n Output Modification Clear Register (F003 A694_H + n*100_H) Reset Value: 0000 0000_H

Pn_OMCR (n=20-23)

Port n Output Modification Clear Register (F003 AC94_H + n*100_H) Reset Value: 0000 0000_H

Pn_OMCR (n=32-33)

Port n Output Modification Clear Register (F003 B294_H + n*100_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
PCLx (x = 0-15)	x + 16	w	Port n Clear Bit x Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0 _B No operation 1 _B Clears Pn_OUT.Px.
0	[15:0]	r	Reserved Read as 0; should be written with 0

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.3.12 Port Output Modification Clear Registers x

The port output modification clear register x, (x = 0, 4, 8, 12) contains control bits to individually clear the logic state of a single port line by manipulating the output register.

Register Pn_OMCR0 clears the logic state of Pn.[3:0] port lines

Register Pn_OMCR4 clears the logic state of Pn.[7:4] port lines

Register Pn_OMCR8 clears the logic state of Pn.[11:8] port lines

Register Pn_OMCR12 clears the logic state of Pn.[15:12] port lines

P00_OMCR0

Port 00 Output Modification Clear Register 0(80_H) **Reset Value: 0000 0000_H**

P02_OMCR0

Port 02 Output Modification Clear Register 0(80_H) **Reset Value: 0000 0000_H**

Pn_OMCR0 (n=10-11)

Port n Output Modification Clear Register 0(F003 A680_H + n*100_H) **Reset Value: 0000 0000_H**

Pn_OMCR0 (n=13-15)

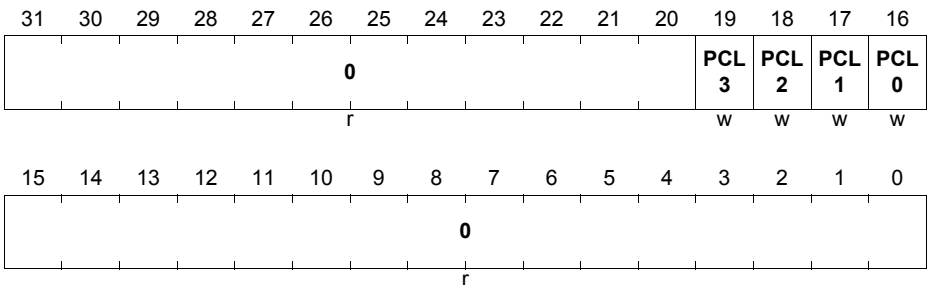
Port n Output Modification Clear Register 0(F003 A680_H + n*100_H) **Reset Value: 0000 0000_H**

Pn_OMCR0 (n=20-23)

Port n Output Modification Clear Register 0(F003 AC80_H + n*100_H) **Reset Value: 0000 0000_H**

Pn_OMCR0 (n=32-33)

Port n Output Modification Clear Register 0(F003 B280_H + n*100_H) **Reset Value: 0000 0000_H**



General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PCLx (x = 0-3)	x + 16	w	Port n Clear Bit x Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0 _B No operation 1 _B Clears Pn_OUT.Px
0	[31:20], [15:0]	r	Reserved Read as 0; should be written with 0.

P00_OMCR4

Port 00 Output Modification Clear Register 4(84_H) **Reset Value: 0000 0000_H**

P02_OMCR4

Port 02 Output Modification Clear Register 4(84_H) **Reset Value: 0000 0000_H**

Pn_OMCR4 (n=10-11)

Port n Output Modification Clear Register 4(F003 A684_H + n*100_H) **Reset Value: 0000 0000_H**

Pn_OMCR4 (n=14-15)

Port n Output Modification Clear Register 4(F003 A684_H + n*100_H) **Reset Value: 0000 0000_H**

Pn_OMCR4 (n=20-21)

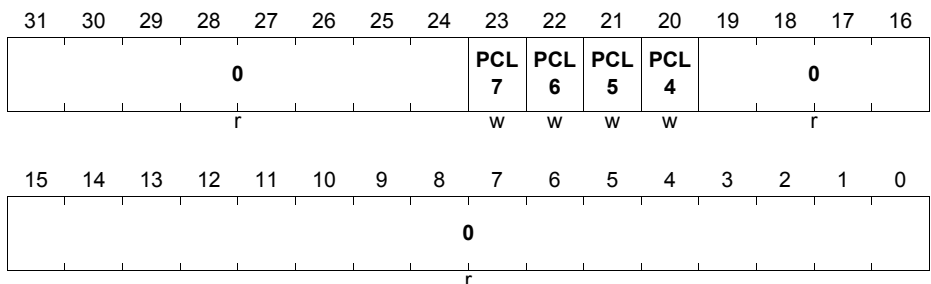
Port n Output Modification Clear Register 4(F003 AC84_H + n*100_H) **Reset Value: 0000 0000_H**

P32_OMCR4

Port 32 Output Modification Clear Register 4(84_H) **Reset Value: 0000 0000_H**

P33_OMCR4

Port 33 Output Modification Clear Register 4(84_H) **Reset Value: 0000 0000_H**



General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PCLx (x = 7-4)	x + 16	w	Port n Clear Bit x Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0 _B No operation 1 _B Clears Pn_OUT.Px
0	[31:24], [19:0]	r	Reserved Read as 0; should be written with 0.

P00_OMCR8

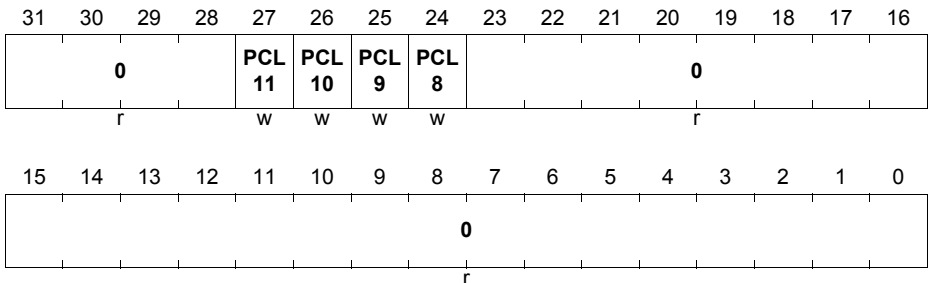
 Port 00 Output Modification Clear Register 8(88_H) Reset Value: 0000 0000_H
P02_OMCR8

 Port 02 Output Modification Clear Register 8(88_H) Reset Value: 0000 0000_H
Pn_OMCR8 (n=10-11)

 Port n Output Modification Clear Register 8(F003 A688_H + n*100_H) Reset Value: 0000 0000_H
Pn_OMCR8 (n=14-15)

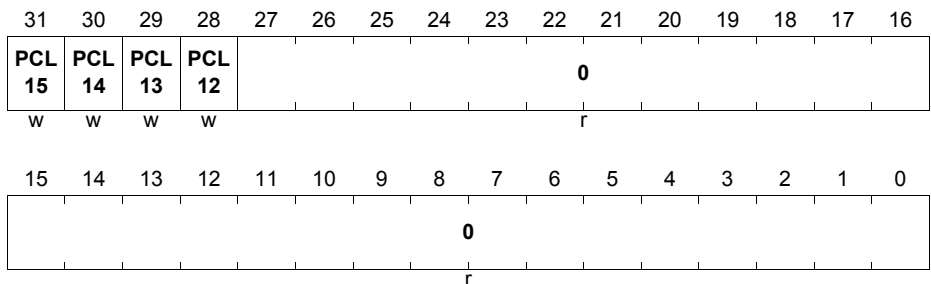
 Port n Output Modification Clear Register 8(F003 A688_H + n*100_H) Reset Value: 0000 0000_H
P20_OMCR8

 Port 20 Output Modification Clear Register 8(88_H) Reset Value: 0000 0000_H
P33_OMCR8

 Port 33 Output Modification Clear Register 8(88_H) Reset Value: 0000 0000_H


General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PCLx (x = 11-8)	x + 16	w	Port n Clear Bit x Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0 _B No operation 1 _B Clears Pn_OUT.Px
0	[31:28], [23:0]	r	Reserved Read as 0; should be written with 0.

P00_OMCR12
Port 00 Output Modification Clear Register 12(8C_H) **Reset Value: 0000 0000_H**
P11_OMCR12
Port 11 Output Modification Clear Register 12(8C_H) **Reset Value: 0000 0000_H**
P20_OMCR12
Port 20 Output Modification Clear Register 12(8C_H) **Reset Value: 0000 0000_H**
P33_OMCR12
Port 33 Output Modification Clear Register 12(8C_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
PCLx (x = 15-12)	x + 16	w	Port n Clear Bit x Setting this bit will clear the corresponding bit in the port output register Pn_OUT. Read as 0. 0 _B No operation 1 _B Clears Pn_OUT.Px
0	[27:0]	r	Reserved Read as 0; should be written with 0.

14.3.13 Emergency Stop Register

All digital GPIO lines have an emergency stop logic implemented (see [Figure 14-1](#)).

Each of these GPIO lines has its own emergency stop enable bit ENx that is located in the emergency stop register Pn_ESR of Port n. If the emergency stop signal becomes active, one of two states can be selected:

- Emergency stop function disabled (ENx = 0):
The output line remains connected (alternate function).
- Emergency stop function enabled (ENx = 1):
The mapped output function is disconnected and the safe state is entered by switching to input function with internal pull-up connected or tri-state, depending on the configured reset value of the corresponding Pn_IOCR register through PMSWCR0.TRISTREQ or setting of HWCFG[6]. (the content of the corresponding PCx bit fields in register Pn_IOCR will not be considered).

P00_ESR

Port 00 Emergency Stop Register (50_H) Reset Value: 0000 0000_H

P02_ESR

Port 02 Emergency Stop Register (50_H) Reset Value: 0000 0000_H

Pn_ESR (n=10-11)

Port n Emergency Stop Register (F003 A650_H + n*100_H) Reset Value: 0000 0000_H

Pn_ESR (n=13-15)

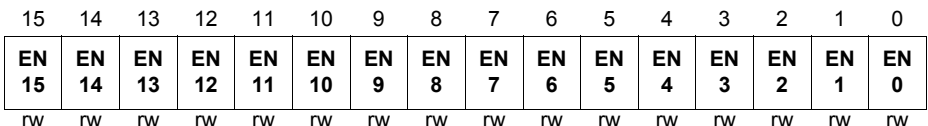
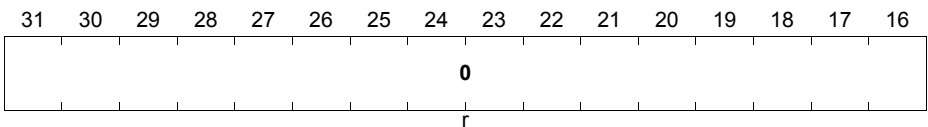
Port n Emergency Stop Register (F003 A650_H + n*100_H) Reset Value: 0000 0000_H

Pn_ESR (n=20-23)

Port n Emergency Stop Register (F003 AC50_H + n*100_H) Reset Value: 0000 0000_H

Pn_ESR (n=32-33)

Port n Emergency Stop Register (F003 B250_H + n*100_H) Reset Value: 0000 0000_H



General Purpose I/O Ports and Peripheral I/O Lines (Ports)

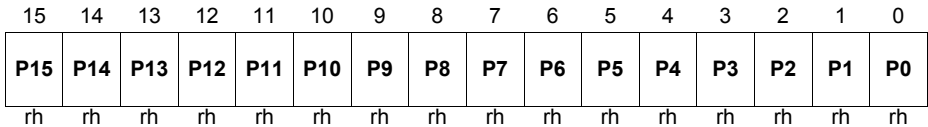
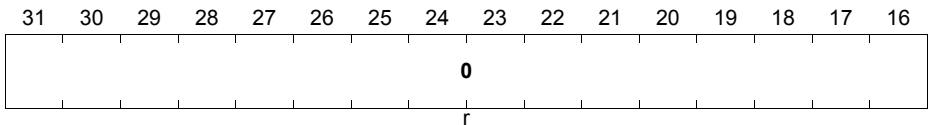
Field	Bits	Type	Description
ENx (x = 0-15)	x	rw	Emergency Stop Enable for Port n Pin x This bit enables the emergency stop function for all GPIO lines. If the emergency stop condition is met and enabled, the output selection is automatically switched from alternate output function to GPIO input function with internal pull-up device connected or tri-state. See Page 14-50 . 0 _B Emergency stop function for Pn.x is disabled. 1 _B Emergency stop function for Pn.x is enabled.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

14.3.14 Port Input Register

The logic level of a GPIO pin can be read via the read-only port input register Pn_IN. Reading the Pn_IN register always returns the current logical value at the GPIO pin independently whether the pin is selected as input or output.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P00_IN		
Port 00 Input Register	(24 _H)	Reset Value: 0000 XXXX _H
P02_IN		
Port 02 Input Register	(24 _H)	Reset Value: 0000 XXXX _H
Pn_IN (n=10-11)		
Port n Input Register	(F003 A624 _H + n*100 _H)	Reset Value: 0000 XXXX _H
Pn_IN (n=13-15)		
Port n Input Register	(F003 A624 _H + n*100 _H)	Reset Value: 0000 XXXX _H
Pn_IN (n=20-23)		
Port n Input Register	(F003 AC24 _H + n*100 _H)	Reset Value: 0000 XXXX _H
Pn_IN (n=32-33)		
Port n Input Register	(F003 B224 _H + n*100 _H)	Reset Value: 0000 XXXX _H
P40_IN		
Port 40 Input Register	(24 _H)	Reset Value: 0000 XXXX _H



Field	Bits	Type	Description
Px (x = 0-15)	x	rh	Port n Input Bit x This bit indicates the level at the input pin Pn.x. 0 _B The input level of Pn.x is 0. 1 _B The input level of Pn.x is 1.
0	[31:16]	r	Reserved Read as 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
ENx (x = 0-31)	x	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

The Access Enable Register 1 controls write¹⁾ access for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000_B, EN1 -> TAG ID 100001_B, ... ,EN31 -> TAG ID 111111_B.

P00_ACCEN1

Port 00 Access Enable Register 1 (F8_H) Reset Value: 0000 0000_H

P02_ACCEN1

Port 02 Access Enable Register 1 (F8_H) Reset Value: 0000 0000_H

Px_ACCEN1 (x=10-11)

Port x Access Enable Register 1(F003 A6F8_H + x*100_H) Reset Value: 0000 0000_H

Px_ACCEN1 (x=13-15)

Port x Access Enable Register 1(F003 A6F8_H + x*100_H) Reset Value: 0000 0000_H

Px_ACCEN1 (x=20-23)

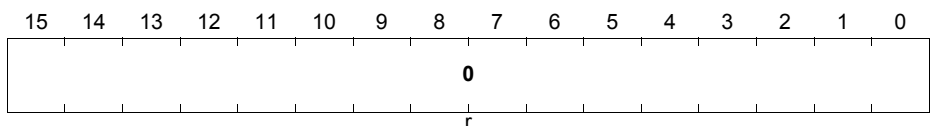
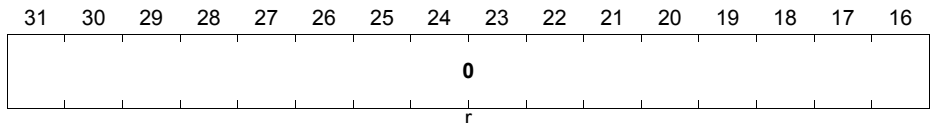
Port x Access Enable Register 1(F003 ACF8_H + x*100_H) Reset Value: 0000 0000_H

Px_ACCEN1 (x=32-33)

Port x Access Enable Register 1(F003 B2F8_H + x*100_H) Reset Value: 0000 0000_H

P40_ACCEN1

Port 40 Access Enable Register 1 (F8_H) Reset Value: 0000 0000_H



General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.4 Port 00

This section describes the Port 00 functionality.

14.4.1 Port 00 Configuration

Port 00 is a 13-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, CCU6, Ethernet, MSC, QSPI, MultiCAN+, PSI5, PSI5-S, SENT, DSADC, VADC, SCU, CIF, ASCLIN and GPT12 input and output functions.

14.4.2 Port 00 Function Table

Table 14-10 summarizes the I/O control selection functions of each Port 00 line.

Table 14-10 Port 00 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.0	I	General-purpose input	P00_IN.P0	P00_IOCRO. PC0	0XXX _B
		GTM input	TIN9		
		CCU61 input	CTRAPA		
		CCU60 input	T12HRE		
		ETH input	ETHMDIOA		
		MSC0 input	INJ00		
		CIF input	CIFD9		
	O	General-purpose output	P00_OUT.P0		1X000 _B
		GTM output	TOUT9		1X001 _B
		ASCLIN3 output	ASCLK3		1X010 _B
		ASCLIN3 output	ATX3		1X011 _B
		Reserved	–		1X100 _B
		CAN node 1 output	TXDCAN1		1X101 _B
		Reserved	–		1X110 _B
	CCU60 output	COUT63		1X111 _B	
DIRx	ETH output	ETHMDIO	HWOUT		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-10 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.1	I	General-purpose input	P00_IN.P1	P00_IOCRO. PC1	0XXXX _B
		GTM input	TIN10		
		CAN node 1 input	RXDCAN1D		
		PSI5 input	PSIRX0A		
		SENT input	SENT0B		
		CCU60 input	CC60INB		
		CCU61 input	CC60INA		
		DSADC input	DSCIN0A		
		VADC input	VADCG3.11		
		CIF input	CIFD10		
	ASCLIN3 input	ARX3E			
	O	General-purpose output	P00_OUT.P1	1X000 _B	
		GTM output	TOUT10	1X001 _B	
		ASCLIN3 output	ATX3	1X010 _B	
		Reserved	–	1X011 _B	
		DSADC output	DSCOUT0	1X100 _B	
Reserved		–	1X101 _B		
SENT output		SPC0	1X110 _B		
CCU61 output		CC60	1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Table 14-10 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.2	I	General-purpose input	P00_IN.P2	P00_IOCRO. PC2	0XXXX _B
		GTM input	TIN11		
		SENT input	SENT1B		
		DSADC input	DSDIN0A		
		VADC input	VADCG3.10		
		CIF input	CIFD11		
	O	General-purpose output	P00_OUT.P2		1X000 _B
		GTM output	TOUT11		1X001 _B
		ASCLIN3 output	ASCLK3		1X010 _B
		Reserved	–		1X011 _B
		PSI5 output	PSITX0		1X100 _B
		CAN node 3 output	TXDCAN3		1X101 _B
		Reserved	–		1X110 _B
		CCU61 output	COU60		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-10 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.3	I	General-purpose input	P00_IN.P3	P00_IOCRO. PC3	0XXXX _B
		GTM input	TIN12		
		CAN node 3 input	RXDCAN3A		
		PSI5 input	PSIRX1A		
		SENT input	SENT2B		
		CCU60 input	CC61INB		
		CCU61 input	CC61INA		
		DSADC input	DSCIN3A		
		VADC input	VADCG3.9		
		CIF input	CIFD12		
	PSI5-S input	PSISRXA			
	O	General-purpose output	P00_OUT.P3	1X000 _B	
		GTM output	TOUT12	1X001 _B	
		ASCLIN3 output	ASLSO3	1X010 _B	
		Reserved	–	1X011 _B	
		DSADC output	DSCOUT3	1X100 _B	
Reserved		–	1X101 _B		
SENT output		SPC2	1X110 _B		
CCU61 output		CC61	1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-10 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.4	I	General-purpose input	P00_IN.P4	P00_IOC4. PC4	0XXXX _B
		GTM input	TIN13		
		SCU input	REQ7		
		SENT input	SENT3B		
		DSADC input	DSDIN3A		
		DSADC input	DSSGNA		
		VADC input	VADCG3.8		
		CIF input	CIFD13		
	O	General-purpose output	P00_OUT.P4		1X000 _B
		GTM output	TOUT13		1X001 _B
		PSI5-S output	PSISTX		1X010 _B
		Reserved	–		1X011 _B
		CAN node 4 output	TXDCAN4		1X011 _B
		PSI5 output	PSITX1		1X100 _B
		VADC output	VADCG2BFL0		1X101 _B
		SENT output	SPC3		1X110 _B
CCU61 output	COUT61	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-10 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.5	I	General-purpose input	P00_IN.P5	P00_IOCRA. PC5	0XXXX _B
		GTM input	TIN14		
		PSI5 input	PSIRX2A		
		SENT input	SENT4B		
		CCU60 input	CC62INB		
		CCU61 input	CC62INA		
		DSADC input	DSCIN2A		
		VADC input	VADCG3.7		
		CAN node 4 input	RXDCAN4A		
	CIF input	CIFD14			
	O	General-purpose output	P00_OUT.P5		1X000 _B
		GTM output	TOUT14		1X001 _B
		DSADC output	DSCGPWMN		1X010 _B
		Reserved	–		1X011 _B
		DSADC output	DSCOUT2		1X100 _B
		VADC output	VADCG2BFL1		1X101 _B
SENT output		SPC4		1X110 _B	
CCU61 output	CC62		1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-10 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.6	I	General-purpose input	P00_IN.P6	P00_IOCRA. PC6	0XXXX _B
		GTM input	TIN15		
		SENT input	SENT5B		
		DSADC input	DSDIN2A		
		VADC input	VADCG3.6		
		CIF input	CIFD15		
	O	General-purpose output	P00_OUT.P6		1X000 _B
		GTM output	TOUT15		1X001 _B
		DSADC output	DSCGPWMP		1X010 _B
		VADC output	VADCG2BFL2		1X011 _B
		PSI5 output	PSITX2		1X100 _B
		VADC output	VADCEMUX10		1X101 _B
		SENT output	SPC5		1X110 _B
		CCU61 output	COUT62		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-10 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.7	I	General-purpose input	P00_IN.P7	P00_IOCRA. PC7	0XXXX _B
		GTM input	TIN16		
		CCU61 input	CC60INC		
		CCU61 input	CCPOS0A		
		CCU60 input	T12HRB		
		GPT120 input	T2INA		
		VADC input	VADCG3.5		
		CIF input	CIFCLK		
	O	General-purpose output	P00_OUT.P7		1X000 _B
		GTM output	TOUT16		1X001 _B
		Reserved	–		1X010 _B
		VADC output	VADCG2BFL3		1X011 _B
		Reserved	–		1X100 _B
		VADC output	VADCEMUX11		1X101 _B
		Reserved	–		1X110 _B
CCU61 output	CC60		1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Table 14-10 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.8	I	General-purpose input	P00_IN.P8	P00_IOC8. PC8	0XXXX _B
		GTM input	TIN17		
		CCU61 input	CC61INC		
		CCU61 input	CCPOS1A		
		CCU60 input	T13HRB		
		GPT120 input	T2EUDA		
		VADC input	VADCG3.4		
		CIF input	CIFVSNC		
	O	General-purpose output	P00_OUT.P8		1X000 _B
		GTM output	TOUT17		1X001 _B
		QSPI3 output	SLSO36		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		VADC output	VADCEMUX12		1X101 _B
		Reserved	–		1X110 _B
CCU61 output	CC61		1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-10 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P00.9	I	General-purpose input	P00_IN.P9	P00_IOCRR. PC9	0XXXX _B
		GTM input	TIN18		
		CCU61 input	CC62INC		
		CCU61 input	CCPOS2A		
		CCU60 input	T13HRC		
		CCU60 input	T12HRC		
		GPT120 input	T4EUDA		
		VADC input	VADCG3.3		
		DSADC input	DSITR3F		
		CIF input	CIFHSNC		
	O	General-purpose output	P00_OUT.P9		1X000 _B
		GTM output	TOUT18		1X001 _B
		QSPI3 output	SLSO37		1X010 _B
		ASCLIN3 output	ARTS3		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		CCU61 output	CC62		1X111 _B
P00.10	I	General-purpose input	P00_IN.P10	P00_IOCRR. PC10	0XXXX _B
		GTM input	TIN19		
		VADC input	VADCG3.2		
	O	General-purpose output	P00_OUT.P10		1X000 _B
		GTM output	TOUT19		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B
		CCU61 output	COOUT63		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-10 Port 00 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
P00.11	I	General-purpose input	P00_IN.P11	P00_IOCRR8. PC11	0XXXX _B	
		GTM input	TIN20			
		CCU60 input	CTRAPA			
		CCU61 input	T12HRE			
		VADC input	VADCG3.1			
	O	General-purpose output	P00_OUT.P11			1X000 _B
		GTM output	TOUT20			1X001 _B
		Reserved	–			1X010 _B
		Reserved	–			1X011 _B
		DSADC output	DSCOUT0			1X100 _B
		Reserved	–			1X101 _B
		Reserved	–			1X110 _B
		Reserved	–			1X111 _B
		P00.12	I			General-purpose input
GTM input	TIN21					
VADC input	VADCG3.0					
ASCLIN3 input	ACTS3A					
O	General-purpose output		P00_OUT.P12	1X000 _B		
	GTM output		TOUT21	1X001 _B		
	Reserved		–	1X010 _B		
	Reserved		–	1X011 _B		
	Reserved		–	1X100 _B		
	Reserved		–	1X101 _B		
	Reserved		–	1X110 _B		
	Reserved		–	1X111 _B		
	CCU61 output		COOUT63			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.4.3 Port 00 Registers

The following registers are available on Port 00:

Table 14-11 Port 00 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P00_OUT	Port 00 Output Register	0000 _H	Page 14-68 ²⁾
P00_OMR	Port 00 Output Modification Register	0004 _H	Page 14-68 ²⁾
P00_IOCRO	Port 00 Input/Output Control Register 0	0010 _H	Page 14-14
P00_IOCRR4	Port 00 Input/Output Control Register 4	0014 _H	Page 14-16
P00_IOCRR8	Port 00 Input/Output Control Register 8	0018 _H	Page 14-18
P00_IOCRR12	Port 00 Input/Output Control Register 12	001C _H	Page 14-69 ²⁾
P00_IN	Port 00 Input Register	0024 _H	Page 14-70 ²⁾
P00_PDR0	Port 00 Pad Driver Mode 0 Register	0040 _H	Page 14-25
P00_PDR1	Port 00 Pad Driver Mode 1 Register	0044 _H	Page 14-71 ²⁾
P00_ESR	Port 00 Emergency Stop Register	0050 _H	Page 14-71 ²⁾
P00_PCSR	Port 02 Pin Controller Select Register	0064 _H	Page 14-72 ²⁾
P00_OMSR0	Port 00 Output Modification Set Register 0	0070 _H	Page 14-41
P00_OMSR4	Port 00 Output Modification Set Register 4	0074 _H	Page 14-42
P00_OMSR8	Port 00 Output Modification Set Register 8	0078 _H	Page 14-43
P00_OMSR12	Port 00 Output Modification Set Register 12	007C _H	Page 14-68 ²⁾
P00_OMCR0	Port 00 Output Modification Clear Register 0	0080 _H	Page 14-46
P00_OMCR4	Port 00 Output Modification Clear Register 4	0084 _H	Page 14-47
P00_OMCR8	Port 00 Output Modification Clear Register 8	0088 _H	Page 14-48
P00_OMCR12	Port 00 Output Modification Clear Register 12	008C _H	Page 14-68 ²⁾
P00_OMSR	Port 00 Output Modification Set Register	0090 _H	Page 14-68 ²⁾
P00_OMCR	Port 00 Output Modification Clear Register	0094 _H	Page 14-68 ²⁾
P00_ACCEN1	Port 00 Access Enable Register 1	00F8 _H	Page 14-53
P00_ACCEN0	Port 00 Access Enable Register 0	00FC _H	Page 14-54

1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))

2) These registers are listed and noted here in the Port 00 section because they differ from the general port register description given in [Section 14.3](#).

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.4.3.1 Port 00 Output Register

The basic P00_OUT register functionality is described on [Page 14-36](#). Port lines P00.[15:13] are not connected. Reading the P00_OUT bits P[15:13] returns the value that was last written. These connected bits can be also set/reset by the corresponding bits in P00_OMR, P00_OMSR, P00_OMCR, P00_OMSRx (x=0,4,8,12), P00_OMCRx (x=0,4,8,12).

14.4.3.2 Port 00 Output Modification Register

The basic P00_OMR register functionality is described on [Page 14-38](#). However, port lines P00.[15:13] are not connected. The P00_OMR bits PS[15:13] and PCL[15:13] have no direct effect on port lines but only on register bits P00_OUT.P[15:13].

14.4.3.3 Port 00 Output Modification Set Register

The basic P00_OMSR register functionality is described on [Page 14-40](#). However, port lines P00.[15:13] are not connected. The P00_OMSR bits PS[15:13] have no direct effect on port lines but only on register bits P00_OUT.P[15:13].

14.4.3.4 Port 00 Output Modification Set Register 12

The basic P00_OMSR12 register functionality is described on [Page 14-44](#). However, port lines P00.[15:13] are not connected. The P00_OMSR12 bits PS[15:13] have no direct effect on port lines but only on register bits P00_OUT.P[15:13].

14.4.3.5 Port 00 Output Modification Clear Register

The basic P00_OMCR register functionality is described on [Page 14-45](#). However, port lines P00.[15:13] are not connected. The P00_OMCR bits PCL[15:13] have no direct effect on port lines but only on register bits P00_OUT.P[15:13].

14.4.3.6 Port 00 Output Modification Clear Register 12

The basic P00_OMCR12 register functionality is described on [Page 14-49](#). However, port lines P00.[15:13] are not connected. The P00_OMCR12 bits PCL[15:13] have no direct effect on port lines but only on register bits P00_OUT.P[15:13].

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.4.3.7 Port 00 Input/Output Control Register 12

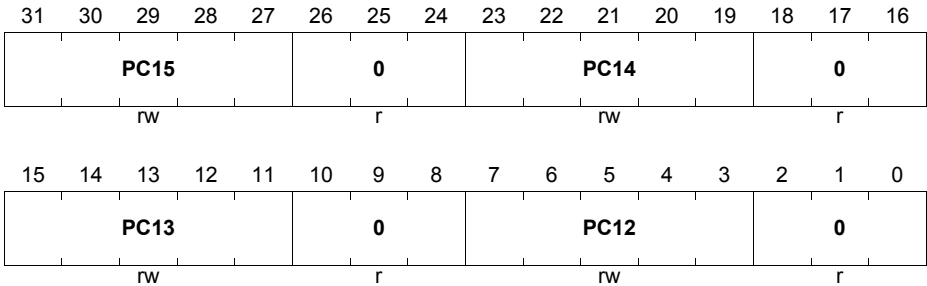
The PC13, PC14 and PC15 bit fields in register P00_IOC12 are not connected.

P00_IOC12

Port 00 Input/Output Control Register 12

(1C_H)

Reset Value: 1010 1010_H

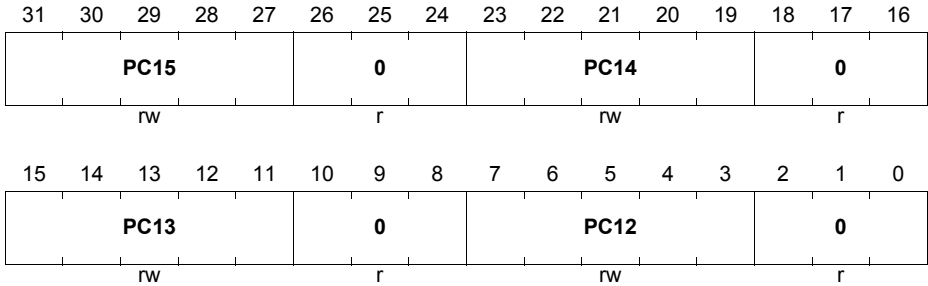


Field	Bits	Type	Description
PC13	[15:11]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
PC14	[23:19]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
PC15	[31:27]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
PC12	[7:3]	rw	Port Control for Port 00.12 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P00_IOCR12
Port 00 Input/Output Control Register 12

 (1C_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
PC13	[15:11]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC14	[23:19]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC15	[31:27]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC12	[7:3]	rw	Port Control for Port 00.12 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

14.4.3.8 Port 00 Input Register

The basic P00_IN register functionality is described on [Page 14-52](#). However, port lines P00.[15:13] are not connected. Therefore, bits P[15:13] in register P00_IN are always read as 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.4.3.9 Port 00 Pad Driver Mode 1 Register

P00_PDR1
Port 00 Pad Driver Mode 1 Register (44_H)
Reset Value: 3333 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL15	PD15		PL14	PD14		PL13	PD13		PL12	PD12					
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL11	PD11		PL10	PD10		PL9	PD9		PL8	PD8					
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
PD8	[2:0]	rw	Pad Driver Mode for Port 00 Pin 8
PD9	[6:4]	rw	Pad Driver Mode for Port 00 Pin 9
PD10	[10:8]	rw	Pad Driver Mode for Port 00 Pin 10
PD11	[14:12]	rw	Pad Driver Mode for Port 00 Pin 11
PD12	[18:16]	rw	Pad Driver Mode for Port 00 Pin 12
PL8	3	rw	Pad Level Selection for Port 00 Pin 8
PL9	7	rw	Pad Level Selection for Port 00 Pin 9
PL10	11	rw	Pad Level Selection for Port 00 Pin 10
PL11	15	rw	Pad Level Selection for Port 00 Pin 11
PL12	19	rw	Pad Level Selection for Port 00 Pin 12
PD13, PD14, PD15	[22:20], [26:24], [30:28]	rw	Reserved Read as 000 _B after reset; returns value that was written.
PL13, PL14, PL15	23, 27, 31	rw	Reserved Read as 0 _B after reset; returns value that was written.

14.4.3.10 Port 00 Emergency Stop Register

The basic P00_ESR register functionality is described on [Page 14-50](#). Port lines P00.[15:13] are not connected. Reading the P00_ESR bits EN[15:13] returns the value that was last written.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.4.3.11 Port 00 Pin Controller Select Register

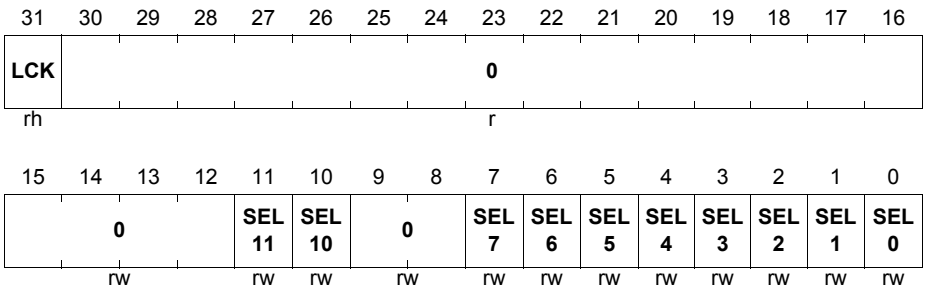
P00_PCSR selects the 32-bit or 8-bit controller for all data and control functions of each shared port pin separately during run, sleep and standby modes.

For pins used for 8-bit controller and VADC Multiplexer Diagnostics (MD) feature, SPAREINOUT0 bit 0 decides if input pull-up and pull-down is controlled by 8-bit controller or VADC Multiplexer Diagnostics (MD) feature is enabled.

P00_PCSR

Port 00 Pin Controller Select Register (64_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SELx (x = 1-0)	0 + x	rw	<p>Pin Controller Select for Pin x</p> <p>This bit selects the controller in all data and control functions of the port pin.</p> <p>0_B 32-bit controller is selected for data and control of pin x.</p> <p>1_B 8-bit controller is selected for data and control functions of pin x.</p>
SELx (x = 7-4)	0 + x	rw	<p>Pin Controller Select for Pin x</p> <p>This bit selects the controller in all data and control functions of the port pin.</p> <p>0_B 32-bit controller is selected for data and control of pin x.</p> <p>1_B 8-bit controller is selected for data and control functions of pin x.</p>

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
SELx (x = 3-2)	0 + x	rw	<p>Pin Controller Select for Pin x</p> <p>This bit selects the controller in all data and control functions of the port pin. This bit along with SPAREINOUT0 bit 0 decides if pull-up and pull-down is selected through 8-bit controller or VADC Multiplexer Diagnostics (MD) is enabled feature for pin x .</p> <p>For the respective analog signal connection, see analog connections table in VADC chapter.</p> <p>0_B 32-bit controller is selected for data and control of pin x. Disables VADC MD feature.</p> <p>1_B 8-bit controller is selected for data and control of pin x. If SPAREINOUT0 bit 0=0, VADC MD feature is enabled and pull-up and pull-down is not controlled by 8-bit controller.</p> <p>If SPAREINOUT0.bit 0=1, pull-up and pull-down is controlled by 8-bit controller, and VADC MD feature is disabled.</p>
SELx (x = 11-10)	0 + x	rw	<p>Pin Controller Select for Pin x</p> <p>This bit enables or disables the VADC Multiplexer Diagnostics (MD) feature.</p> <p>For the respective analog signal connection, see analog connections table in VADC chapter.</p> <p>0_B Disables VADC MD feature.</p> <p>1_B Enables VADC MD feature.</p>
LCK	31	rh	<p>Lock Status</p> <p>This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus has no effect.</p> <p>0_B The register is unlocked and can be updated.</p> <p>1_B The register is locked and can not be updated.</p>
0	[9:8], [15:12]	rw	<p>Reserved</p> <p>Read as 0; returns value last written, must be written with 0.</p>
0	[30:16]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.5 Port 02

This section describes the Port 02 functionality.

14.5.1 Port 02 Configuration

Port 02 is a 9-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, SCU, CCU6, ASCLIN, QSPI, DSADC, MultiCAN+, ERAY, PSI5, MSC, I2C, SENT, GPT12, VADC, CIF, PSI5-S and Ethernet functions.

14.5.2 Port 02 Function Table

Table 14-12 summarizes the I/O control selection functions of each Port 02 line.

Table 14-12 Port 02 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.0	I	General-purpose input	P02_IN.P0	P02_IOCR0. PC0	0XXXX _B
		GTM input	TIN0		
		SCU input	REQ6		
		CCU60 input	CC60INA		
		CCU61 input	CC60INB		
		CIF input	CIFD0		
		ASCLIN2 input	ARX2G		
	O	General-purpose output	P02_OUT.P0		1X000 _B
		GTM output	TOUT0		1X001 _B
		ASCLIN2 output	ATX2		1X010 _B
		QSPI3 output	SLSO31		1X011 _B
		DSADC output	DSCGPWMN		1X100 _B
		CAN node 0 output	TXDCAN0		1X101 _B
		ERAY output	TXDA		1X110 _B
CCU60 output	CC60		1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-12 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
P02.1	I	General-purpose input	P02_IN.P1	P02_IOCRO. PC1	0XXXX _B	
		GTM input	TIN1			
		ASCLIN2 input	ARX2B			
		CAN node 0 input	RXDCAN0A			
		ERAY input	RXDA2			
		CIF input	CIFD1			
		SCU input	REQ14			
	O	General-purpose output	P02_OUT.P1		1X000 _B	
		GTM output	TOUT1		1X001 _B	
		Reserved	–		1X010 _B	
		QSPI3 output	SLSO32		1X011 _B	
		DSADC output	DSCGPWMP		1X100 _B	
		Reserved	–		1X101 _B	
		Reserved	–		1X110 _B	
		CCU60 output	COU60		1X111 _B	
P02.2	I	General-purpose input	P02_IN.P2	P02_IOCRO. PC2	0XXXX _B	
		GTM input	TIN2			
		CCU60 input	CC61INA			
		CCU61 input	CC61INB			
		CIF input	CIFD2			
	O	General-purpose output	P02_OUT.P2		1X000 _B	
		GTM output	TOUT2		1X001 _B	
		ASCLIN1 output	ATX1		1X010 _B	
		QSPI3 output	SLSO33		1X011 _B	
		PSI5 output	PSITX0		1X100 _B	
		CAN node 2 output	TXDCAN2		1X101 _B	
		ERAY output	TXDB		1X110 _B	
				CCU60 output	CC61	

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Table 14-12 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.3	I	General-purpose input	P02_IN.P3	P02_IOCR0. PC3	0XXXX _B
		GTM input	TIN3		
		CAN node 2 input	RXDCAN2B		
		ERAY input	RXDB2		
		PSI5 input	PSIRX0B		
		MSC1 input	SDI11		
		CIF input	CIFD3		
		ASCLIN1 input	ARX1G		
	O	General-purpose output	P02_OUT.P3		1X000 _B
		GTM output	TOUT3		1X001 _B
		ASCLIN2 output	ASLSO2		1X010 _B
		QSPI3 output	SLSO34		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
CCU60 output	COUT61		1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-12 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.4	I	General-purpose input	P02_IN.P4	P02_IOCR4. PC4	0XXXX _B
		GTM input	TIN4		
		QSPI3 input	SLSI3A		
		TTCAN input	ECTT1		
		CAN node 0 input	RXDCAN0D		
		CCU60 input	CC62INA		
		CCU61 input	CC62INB		
		I2C0 input	SDA0A		
		CIF input	CIFD4		
	O	General-purpose output	P02_OUT.P4	1X000 _B	
		GTM output	TOUT4	1X001 _B	
		ASCLIN2	ASCLK2	1X010 _B	
		QSPI3 output	SLSO30	1X011 _B	
		PSI5-S output	PSISCLK	1X100 _B	
		I2C0 output	SDA0	1X101 _B	
ERAY output	TXENA	1X110 _B			
CCU60 output	CC62	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-12 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.5	I	General-purpose input	P02_IN.P5	P02_IOCR4. PC5	0XXXX _B
		GTM input	TIN5		
		QSPI3 input	MRST3A		
		TTCAN input	ECTT2		
		PSI5 input	PSIRX1B		
		SENT input	SENT3C		
		I2C0 input	SCL0A		
		CIF input	CIFD5		
		PSI5-S input	PSISRXB		
	O	General-purpose output	P02_OUT.P5	1X000 _B	
		GTM output	TOUT5	1X001 _B	
		CAN node 0 output	TXDCAN0	1X010 _B	
		QSPI3 output	MRST3	1X011 _B	
		Reserved	–	1X100 _B	
		I2C0 output	SCL0	1X101 _B	
ERAY output	TXENB	1X110 _B			
CCU60 output	COUT62	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-12 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.6	I	General-purpose input	P02_IN.P6	P02_IOCR4. PC6	0XXXX _B
		GTM input	TIN6		
		QSPI3 input	MTR3A		
		SENT input	SENT2C		
		CCU60 input	CC60INC		
		CCU60 input	CCPOS0A		
		CCU61 input	T12HRB		
		GPT120 input	T3INA		
	CIF input	CIFD6			
	O	General-purpose output	P02_OUT.P6		1X000 _B
		GTM output	TOUT6		1X001 _B
		PSI5-S output	PSISTX		1X010 _B
		QSPI3 output	MTR3		1X011 _B
		PSI5 output	PSITX1		1X100 _B
		VADC output	VADCEMUX00		1X101 _B
Reserved		–		1X110 _B	
CCU60 output	CC60		1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-12 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.7	I	General-purpose input	P02_IN.P7	P02_IOCR4. PC7	0XXXX _B
		GTM input	TIN7		
		QSPI3 input	SCLK3A		
		PSI5 input	PSIRX2B		
		SENT input	SENT1C		
		CCU60 input	CC61INC		
		CCU60 input	CCPOS1A		
		CCU61 input	T13HRB		
		GPT120 input	T3EUDA		
		DSADC input	DSCIN3B		
	CIF input	CIFD7			
	O	General-purpose output	P02_OUT.P7		1X000 _B
		GTM output	TOUT7		1X001 _B
		Reserved	–		1X010 _B
		QSPI3 output	SCLK3		1X011 _B
		DSADC output	DSCOUT3		1X100 _B
		VADC output	VADCEMUX01		1X101 _B
SENT output		SPC1		1X110 _B	
CCU60 output	CC61		1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-12 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.8	I	General-purpose input	P02_IN.P8	P02_IOCR8. PC8	0XXXX _B
		GTM input	TIN8		
		SENT input	SENT0C		
		CCU60 input	CC62INC		
		CCU60 input	CCPOS2A		
		CCU61 input	T12HRC		
		CCU61 input	T13HRC		
		GPT120 input	T4INA		
		DSADC input	DSDIN3B		
		CIF input	CIFD8		
	DSADC input	DSITR3E			
	O	General-purpose output	P02_OUT.P8		1X000 _B
		GTM output	TOUT8		1X001 _B
		QSPI3 output	SLSO35		1X010 _B
		Reserved	–		1X011 _B
		PSI5 output	PSITX2		1X100 _B
		VADC output	VADCEMUX02		1X101 _B
ETH output		ETHMDC		1X110 _B	
CCU60 output	CC62		1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.5.3 Port 02 Registers

The following registers are available on Port 02:

Table 14-13 Port 02 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P02_OUT	Port 02 Output Register	0000 _H	Page 14-83 ²⁾
P02_OMR	Port 02 Output Modification Register	0004 _H	Page 14-83 ²⁾
P02_IOCRO	Port 02 Input/Output Control Register 0	0010 _H	Page 14-14
P02_IOCRR4	Port 02 Input/Output Control Register 4	0014 _H	Page 14-16
P02_IOCRR8	Port 02 Input/Output Control Register 8	0018 _H	Page 14-85 ²⁾
P02_IN	Port 02 Input Register	0024 _H	Page 14-86 ²⁾
P02_PDR0	Port 02 Pad Driver Mode 0 Register	0040 _H	Page 14-25
P02_PDR1	Port 02 Pad Driver Mode 1 Register	0044 _H	Page 14-87 ²⁾
P02_ESR	Port 02 Emergency Stop Register	0050 _H	Page 14-87 ²⁾
P02_PCSR	Port 02 Pin Controller Select Register	0064 _H	Page 14-88 ²⁾
P02_OMSR0	Port 02 Output Modification Set Register 0	0070 _H	Page 14-41
P02_OMSR4	Port 02 Output Modification Set Register 4	0074 _H	Page 14-42
P02_OMSR8	Port 02 Output Modification Set Register 8	0078 _H	Page 14-83 ²⁾
P02_OMCR0	Port 02 Output Modification Clear Register 0	0080 _H	Page 14-46
P02_OMCR4	Port 02 Output Modification Clear Register 4	0084 _H	Page 14-47
P02_OMCR8	Port 02 Output Modification Clear Register 8	0088 _H	Page 14-84
P02_OMSR	Port 02 Output Modification Set Register	0090 _H	Page 14-83 ²⁾
P02_OMCR	Port 02 Output Modification Clear Register	0094 _H	Page 14-83 ²⁾

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Table 14-13 Port 02 Registers (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾ Address	Description see
P02_ACCEN1	Port 02 Access Enable Register 1	00F8 _H	Page 14-53
P02_ACCEN0	Port 02 Access Enable Register 0	00FC _H	Page 14-54

- 1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))
- 2) These registers are listed and noted here in the Port 02 section because they differ from the general port register description given in [Section 14.3](#).

14.5.3.1 Port 02 Output Register

The basic P02_OUT register functionality is described on [Page 14-36](#). Port lines P02.[15:9] are not connected. Reading the P02_OUT bits P[15:12] always return 0. Reading P[11:9] returns the value that was last written. These connected bits can be also set/reset by the corresponding bits in P02_OMR, P02_OMSR, P02_OMCR, P02_OMSRx (x=0,4,8), P02_OMCRx (x=0,4,8).

14.5.3.2 Port 02 Output Modification Register

The basic P02_OMR register functionality is described on [Page 14-38](#). However, port lines P02.[15:9] are not connected. The P02_OMR bits PS[11:9] and PCL[11:9] have no direct effect on port lines but only on register bits P02_OUT.P[11:9].

14.5.3.3 Port 02 Output Modification Set Register

The basic P02_OMSR register functionality is described on [Page 14-40](#). However, port lines P02.[15:9] are not connected. The P02_OMSR bits PS[11:9] have no direct effect on port lines but only on register bits P02_OUT.P[11:9].

14.5.3.4 Port 02 Output Modification Set Register 8

The basic P02_OMSR8 register functionality is described on [Page 14-43](#). However, port lines P02.[11:9] are not connected. The P02_OMSR8 bits PS[11:9] have no direct effect on port lines but only on register bits P02_OUT.P[11:9].

14.5.3.5 Port 02 Output Modification Clear Register

The basic P02_OMCR register functionality is described on [Page 14-45](#). However, port lines P02.[15:9] are not connected. The P02_OMCR bits PCL[11:9] have no direct effect on port lines but only on register bits P02_OUT.P[11:9].

14.5.3.6 Port 02 Output Modification Clear Register 8

The basic P02_OMCR8 register functionality is described on [Page 14-48](#). However, port lines P02.[11:9] are not connected. The P02_OMCR8 bits PCL[11:9] have no direct effect on port lines but only on register bits P02_OUT.P[11:9].

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.5.3.7 Port 02 Input/Output Control Register 8

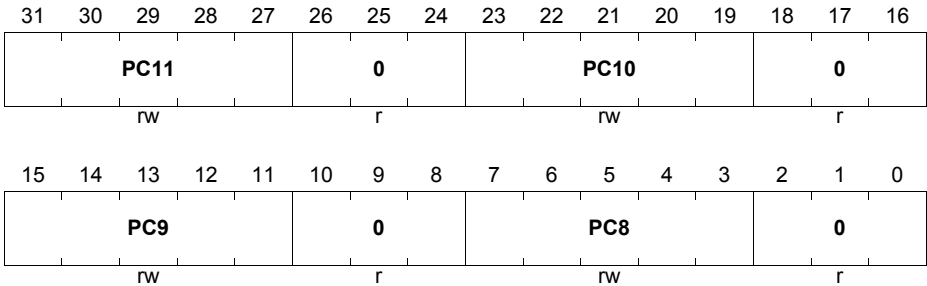
The PC[11:9] bit fields in register P02_IOC8 are not connected.

P02_IOC8

Port 02 Input/Output Control Register 8

(18_H)

Reset Value: 1010 1010_H

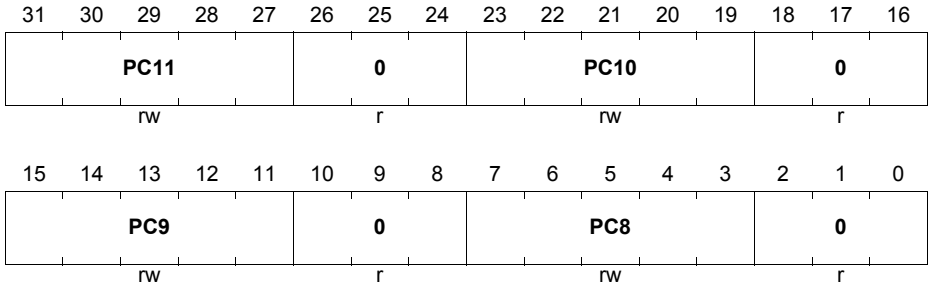


Field	Bits	Type	Description
PC8	[7:3]	rw	Port Control for Port 02 Pin 8 Coding see Table 14-5 .
PC9, PC10, PC11	[15:11], [23:19], [31:27]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P02_IOCR8
Port 02 Input/Output Control Register 8

 (18_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
PC8	[7:3]	rw	Port Control for Port 02 Pin 8 Coding see Table 14-5 .
PC9, PC10, PC11	[15:11], [23:19], [31:27]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

14.5.3.8 Port 02 Input Register

The basic P02_IN register functionality is described on [Page 14-52](#). However, port lines P02.[15:9] are not connected. Therefore, bits P[15:9] in register P02_IN are always read as 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.5.3.9 P02 Pad Driver Mode 1 Register

P02_PDR1

 Port 02 Pad Driver Mode 1 Register (44_H)

 Reset Value: 0000 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL15	PD15		PL14	PD14		PL13	PD13		PL12	PD12					
r	r		r	r		r	r		r	r					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL11	PD11		PL10	PD10		PL9	PD9		PL8	PD8					
rw	rw		rw	rw		rw	rw		rw	rw					

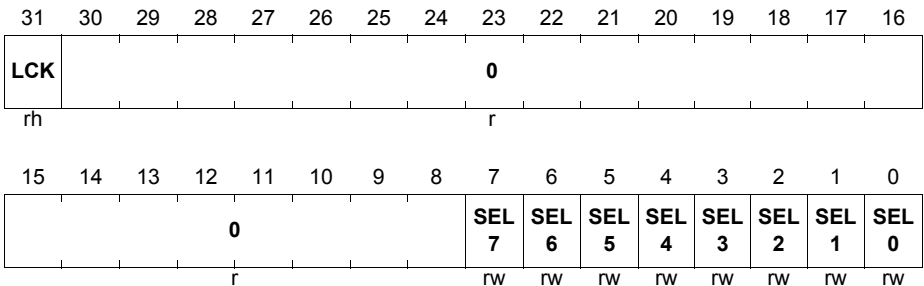
Field	Bits	Type	Description
PD8	[2:0]	rw	Pad Driver Mode for Port n Pin 8
PL8	3	rw	Pad Level Selection for Port n Pin 8
PD9, PD10, PD11	[6:4], [10:8], [14:12]	rw	Reserved Read as 000 _B after reset; returns value that was written, have to be written with 0.
PD12, PD13, PD14, PD15	[18:16], [22:20], [26:24], [30:28]	r	Reserved Read as 000 _B after reset; should be written as 0.
PL9, PL10, PL11	7, 11, 15	rw	Reserved Read as 0 _B after reset; returns value that was written, have to be written with 0.
PL12, PL13, PL14, PL15	19, 23, 27, 31	r	Reserved Read as 0 _B after reset; should be written as 0.

14.5.3.10 Port 02 Emergency Stop Register

The basic P02_ESR register functionality is described on [Page 14-50](#). Port lines P02.[15:9] are not connected. Reading the P02_ESR bits EN[15:12] always returns 0. Reading the P02_ESR bits EN[11:9] returns value that was last written.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.5.3.11 Port 02 Pin Controller Select Register

P02_PCSR selects the 32-bit or 8-bit controller for all data and control functions of each shared port pin separately during run, sleep and standby modes.

P02_PCSR
Port 02 Pin Controller Select Register (64_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SELx (x = 7-0)	0 + x	rw	Pin Controller Select for Pin x This bit selects the controller in all data and control functions of the port pin. 0 _B 32-bit controller is selected for data and control of pin x. 1 _B 8-bit controller is selected for data and control functions.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus has no effect. 0 _B The register is unlocked and can be updated. 1 _B The register is locked and can not be updated.
0	[30:8]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.6 Port 10

This section describes the Port 10 functionality.

14.6.1 Port 10 Configuration

Port 10 is a 9-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, QSPI, CCU6, ASCLIN, MSC, SCU, VADC, MultiCAN+ and GPT12 functions.

After the required state is latched in based on HWCFGx during reset, the corresponding pins can be used as GPIO. For the different start-up configuration, please refer to the firmware and SCU chapters.

14.6.2 Port 10 Function Table

Table 14-14 summarizes the I/O control selection functions of each Port 10 line.

Table 14-14 Port 10 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P10.0	I	General-purpose input	P10_IN.P0	P10_IOCRO. PC0	0XXXX _B
		GTM input	TIN102		
		GPT120 input	T6EUDB		
	O	General-purpose output	P10_OUT.P0		1X000 _B
		GTM output	TOUT102		1X001 _B
		Reserved	–		1X010 _B
		QSPI1 output	SLSO110		1X011 _B
		Reserved	–		1X100 _B
		VADC output	VADCG3BFLO		1X101 _B
		Reserved	–		1X110 _B
Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-14 Port 10 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.				
				Reg./Bit Field	Value			
P10.1	I	General-purpose input	P10_IN.P1	P10_IOCRO. PC1	0XXXX _B			
		GTM input	TIN103					
		QSPI1 input	MRST1A					
		GPT120 input	T5EUDB					
	O	General-purpose output	P10_OUT.P1			1X000 _B		
		GTM output	TOUT103			1X001 _B		
		QSPI1 output	MTR1			1X010 _B		
		QSPI1 output	MRST1			1X011 _B		
		MSC0	EN01			1X100 _B		
		VADC output	VADCG3BFL1			1X101 _B		
		MSC0 output	END03			1X110 _B		
		Reserved	–			1X111 _B		
	P10.2	I	General-purpose input			P10_IN.P2	P10_IOCRO. PC2	0XXXX _B
			GTM input			TIN104		
QSPI1 input			SCLK1A					
SCU input			REQ2					
GPT120 input			T6INB					
MSC0 input			SDI01					
CAN node 2 input			RXDCAN2E					
O		General-purpose output	P10_OUT.P2	1X000 _B				
		GTM output	TOUT104	1X001 _B				
		Reserved	–	1X010 _B				
		QSPI1 output	SCLK1	1X011 _B				
		MSC0	EN00	1X100 _B				
		VADC output	VADCG3BFL2	1X101 _B				
		MSC0 output	END02	1X110 _B				
Reserved	–	1X111 _B						

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-14 Port 10 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
P10.3	I	General-purpose input	P10_IN.P3	P10_IOCRO. PC3	0XXXX _B	
		GTM input	TIN105			
		QSPI1 input	MTSR1A			
		SCU input	REQ3			
		GPT120 input	T5INB			
	O	General-purpose output	P10_OUT.P3			1X000 _B
		GTM output	TOUT105			1X001 _B
		VADC output	VADCG3BFL3			1X010 _B
		QSPI1 output	MTSR1			1X011 _B
		MSC0 output	EN00			1X100 _B
		MSC0 output	END02			1X101 _B
		CAN node 2 output	TXDCAN2			1X110 _B
		Reserved	–			1X111 _B
	P10.4	I	General-purpose input			P10_IN.P4
GTM input			TIN106			
QSPI1 input			MTSR1C			
GPT120 input			T3INB			
CCU60 input			CCPOS0C			
O		General-purpose output	P10_OUT.P4	1X000 _B		
		GTM output	TOUT106	1X001 _B		
		Reserved	–	1X010 _B		
		QSPI1 output	SLSO18	1X011 _B		
		QSPI1 output	MTSR1	1X100 _B		
		MSC0 output	EN00	1X101 _B		
		MSC0 output	END02	1X110 _B		
		Reserved	–	1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-14 Port 10 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P10.5	I	General-purpose input	P10_IN.P5	P10_IOCRA. PC5	0XXXX _B
		GTM input	TIN107		
		SCU input	HWCFG4		
		MSC0 input	INJ01		
		CAN node 4 input	RXDCAN4B		
	O	General-purpose output	P10_OUT.P5		1X000 _B
		GTM output	TOUT107		1X001 _B
		ASCLIN2 output	ATX2		1X010 _B
		QSPI3 output	SLSO38		1X011 _B
		QSPI1 output	SLSO19		1X100 _B
		GPT120 output	T6OUT		1X101 _B
		ASCLIN2 output	ASLSO2		1X110 _B
		Reserved	–		1X111 _B
P10.6	I	General-purpose input	P10_IN.P6	P10_IOCRA. PC6	0XXXX _B
		GTM input	TIN108		
		QSPI3 input	MTR3B		
		SCU input	HWCFG5		
		ASCLIN2 input	ARX2D		
	O	General-purpose output	P10_OUT.P6		1X000 _B
		GTM output	TOUT108		1X001 _B
		ASCLIN2 output	ASCLK2		1X010 _B
		QSPI3 output	MTR3		1X011 _B
		GPT120 output	T3OUT		1X100 _B
		Reserved	–		1X101 _B
		CAN node 4 output	TXDCAN4		1X101 _B
		QSPI1 output	MRST1		1X110 _B
		VADC output	VADCG3BFL0		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-14 Port 10 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P10.7	I	General-purpose input	P10_IN.P7	P10_IOCRR4. PC7	0XXXX _B
		GTM input	TIN109		
		ASCLIN2 input	ACTS2A		
		QSPI3 input	MRST3B		
		SCU input	REQ4		
		CCU60 input	CCPOS1C		
		GPT120 input	T3EUDB		
	O	General-purpose output	P10_OUT.P7		1X000 _B
		GTM output	TOUT109		1X001 _B
		Reserved	–		1X010 _B
		QSPI3 output	MRST3		1X011 _B
		VADC output	VADCG3BFL1		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
Reserved	–		1X111 _B		
P10.8	I	General-purpose input	P10_IN.P8	P10_IOCRR8. PC8	0XXXX _B
		GTM input	TIN110		
		QSPI3 input	SCLK3B		
		SCU input	REQ5		
		CCU60 input	CCPOS2C		
		GPT120 input	T4INB		
	O	General-purpose output	P10_OUT.P8		1X000 _B
		GTM output	TOUT110		1X001 _B
		ASCLIN2 output	ARTS2		1X010 _B
		QSPI3 output	SCLK3		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.6.3 Port 10 Registers

The following registers are available on Port 10:

Table 14-15 Port 10 Registers

Register Short Name	Register Long Name	Offset⁽¹⁾ Address	Description see
P10_OUT	Port 10 Output Register	0000 _H	Page 14-94⁽²⁾
P10_OMR	Port 10 Output Modification Register	0004 _H	Page 14-95⁽²⁾
P10_IOCRO	Port 10 Input/Output Control Register 0	0010 _H	Page 14-14
P10_IOCRR4	Port 10 Input/Output Control Register 4	0014 _H	Page 14-17
P10_IOCRR8	Port 10 Input/Output Control Register 8	0018 _H	Page 14-96⁽²⁾
P10_IN	Port 10 Input Register	0024 _H	Page 14-97⁽²⁾
P10_PDR0	Port 10 Pad Driver Mode 0 Register	0040 _H	Page 14-25
P10_PDR1	Port 10 Pad Driver Mode 1 Register	0044 _H	Page 14-98⁽²⁾
P10_ESR	Port 10 Emergency Stop Register	0050 _H	Page 14-98⁽²⁾
P10_OMSR0	Port 10 Output Modification Set Register 0	0070 _H	Page 14-41
P10_OMSR4	Port 10 Output Modification Set Register 4	0074 _H	Page 14-42
P10_OMSR8	Port 10 Output Modification Set Register 8	0078 _H	Page 14-95⁽²⁾
P10_OMCR0	Port 10 Output Modification Clear Register 0	0080 _H	Page 14-46
P10_OMCR4	Port 10 Output Modification Clear Register 4	0084 _H	Page 14-47
P10_OMCR8	Port 10 Output Modification Clear Register 8	0088 _H	Page 14-95⁽²⁾
P10_OMSR	Port 10 Output Modification Set Register	0090 _H	Page 14-95⁽²⁾
P10_OMCR	Port 10 Output Modification Clear Register	0094 _H	Page 14-95⁽²⁾
P10_ACCEN1	Port 10 Access Enable Register 1	00F8 _H	Page 14-53
P10_ACCEN0	Port 10 Access Enable Register 0	00FC _H	Page 14-54

1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))

2) These registers are listed and noted here in the Port 10 section because they differ from the general port register description given in [Section 14.3](#).

14.6.3.1 Port 10 Output Register

The basic P10_OUT register functionality is described on [Page 14-36](#). Port lines P10.[15:9] are not connected. Reading the P10_OUT bits P[15:12] always return 0. Reading P[11:9] returns the value that was last written. These connected bits can be also

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

set/reset by the corresponding bits in P10_OMR, P10_OMSR, P10_OMCR, P10_OMSRx (x=0,4,8), P10_OMCRx (x=0,4,8).

14.6.3.2 Port 10 Output Modification Register

The basic P10_OMR register functionality is described on [Page 14-38](#). However, port lines P10.[15:9] are not connected. The P10_OMR bits PS[11:9] and PCL[11:9] have no direct effect on port lines but only on register bits P10_OUT.P[11:9].

14.6.3.3 Port 10 Output Modification Set Register

The basic P10_OMSR register functionality is described on [Page 14-40](#). However, port lines P10.[15:9] are not connected. The P10_OMSR bits PS[11:9] have no direct effect on port lines but only on register bits P10_OUT.P[11:9].

14.6.3.4 Port 10 Output Modification Set Register 8

The basic P10_OMSR8 register functionality is described on [Page 14-43](#). However, port lines P10.[11:9] are not connected. The P10_OMSR8 bits PS[11:9] have no direct effect on port lines but only on register bits P10_OUT.P[11:9].

14.6.3.5 Port 10 Output Modification Clear Register

The basic P10_OMCR register functionality is described on [Page 14-45](#). However, port lines P10.[15:9] are not connected. The P10_OMCR bits PCL[11:9] have no direct effect on port lines but only on register bits P10_OUT.P[11:9].

14.6.3.6 Port 10 Output Modification Clear Register 8

The basic P10_OMCR8 register functionality is described on [Page 14-48](#). However, port lines P10.[11:9] are not connected. The P10_OMCR8 bits PCL[11:9] have no direct effect on port lines but only on register bits P10_OUT.P[11:9].

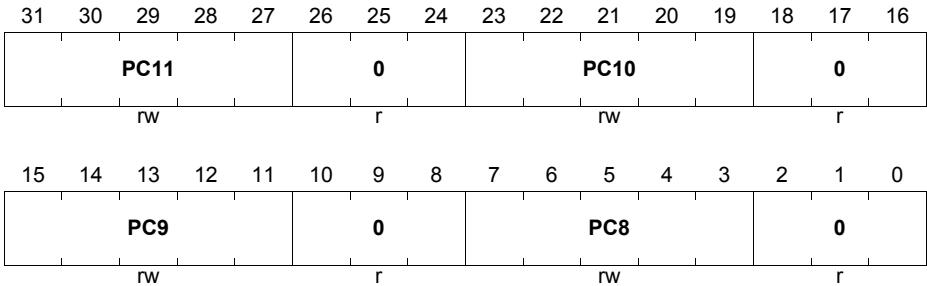
General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.6.3.7 Port 10 Input/Output Control Register 8

The PC[11:9] bit fields in register P10_IOC8 are not connected.

P10_IOC8
Port 10 Input/Output Control Register 8

 (18_H)

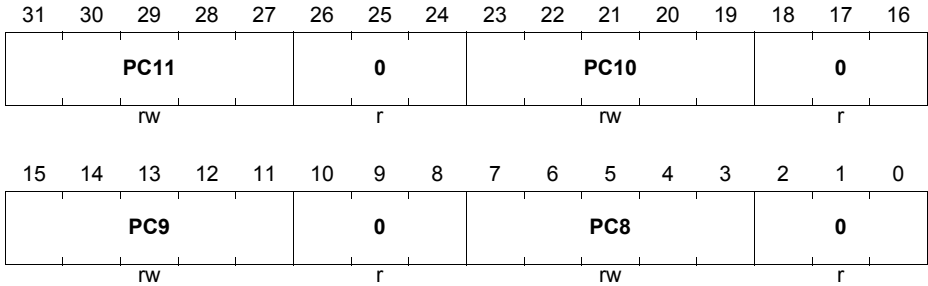
 Reset Value: 1010 1010_H


Field	Bits	Type	Description
PC9, PC10, PC11	[15:11], [23:19], [31:27]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
PC8	[7:3]	rw	Port Control for Port 10 Pin 8 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P10_IOC8
Port 10 Input/Output Control Register 8

 (18_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
PC9, PC10, PC11	[15:11], [23:19], [31:27]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC8	[7:3]	rw	Port Control for Port 10 Pin 8 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

14.6.3.8 Port 10 Input Register

The basic P10_IN register functionality is described on [Page 14-52](#). However, port lines P10.[15:9] are not connected. Therefore, bits P[15:9] in register P10_IN are always read as 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.6.3.9 Port 10 Pad Driver Mode 1 Register

P10_PDR1

 Port 10 Pad Driver Mode 1 Register (44_H)

 Reset Value: 0000 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL15	PD15		PL14	PD14		PL13	PD13		PL12	PD12					
r	r		r	r		r	r		r	r		r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL11	PD11		PL10	PD10		PL9	PD9		PL8	PD8					
r/w	r/w		r/w	r/w		r/w	r/w		r/w	r/w		r/w			

Field	Bits	Type	Description
PD8	[2:0]	r/w	Pad Driver Mode for Port 10 Pin 8
PL8	3	r/w	Pad Level Selection for Port 10 Pin 8
PD9, PD10, PD11	[6:4], [10:8], [14:12]	r/w	Reserved Read as 000 _B after reset; returns value that was written, have to be written with 0.
PD12, PD13, PD14, PD15	[18:16], [22:20], [26:24], [30:28]	r	Reserved Read as 000 _B after reset; should be written with 0.
PL9, PL10, PL11	7, 11, 15	r/w	Reserved Read as 0 _B after reset; returns value that was written, have to be written with 0.
PL12, PL13, PL14, PL15	19, 23, 27, 31	r	Reserved Read as 0 _B after reset; should be written with 0.

14.6.3.10 Port 10 Emergency Stop Register

The basic P10_ESR register functionality is described on [Page 14-50](#). Port lines P10.[15:9] are not connected. Reading the P10_ESR bits EN[15:12] always returns 0. Reading the P10_ESR bits EN[11:9] returns value that was last written.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.7 Port 11

This section describes the Port 11 functionality.

14.7.1 Port 11 Configuration

Port 11 is a 7-bit bi-directional general-purpose I/O Flexport that can be alternatively used for GTM, QSPI, MSC, Ethernet, CCU6, ERAY, MultiCAN+ and SCU functions.

14.7.2 Port 11 Function Table

Table 14-16 summarizes the I/O control selection functions of each Port 11 line.

Table 14-16 Port 11 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P11.2	I	General-purpose input	P11_IN.P2	P11_IOCR0. PC2	0XXXX _B
		GTM input	TIN95		
	O	General-purpose output	P11_OUT.P2		1X000 _B
		GTM output	TOUT95		1X001 _B
		MSC0 output	END03		1X010 _B
		QSPI0 output	SLSO05		1X011 _B
		QSPI1 output	SLSO15		1X100 _B
		MSC0 output	EN01		1X101 _B
		ETH output	ETHTXD1		1X110 _B
		CCU60 output	COOUT63		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-16 Port 11 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.			
				Reg./Bit Field	Value		
P11.3	I	General-purpose input	P11_IN.P3	P11_IOCRO. PC3	0XXXX _B		
		GTM input	TIN96				
		QSPI1 input	MRST1B				
		MSC0 input	SDI03				
	O	General-purpose output	P11_OUT.P3		1X000 _B		
		GTM output	TOUT96		1X001 _B		
		Reserved	–		1X010 _B		
		QSPI1 output	MRST1		1X011 _B		
		ERAY output	TXDA		1X100 _B		
		Reserved	–		1X101 _B		
		ETH output	ETHTXD0		1X110 _B		
		CCU60 output	COOUT62		1X111 _B		
	P11.6	I	General-purpose input		P11_IN.P6	P11_IOCRO4. PC6	0XXXX _B
			GTM input		TIN97		
QSPI1 input			SCLK1B				
O		General-purpose output	P11_OUT.P6	1X000 _B			
		GTM output	TOUT97	1X001 _B			
		ERAY output	$\overline{\text{TXENB}}$	1X010 _B			
		QSPI1 output	SCLK1	1X011 _B			
		ERAY output	$\overline{\text{TXENA}}$	1X100 _B			
		MSC0 output	FCLP0	1X101 _B			
		ETH output	ETHTXEN	1X110 _B			
		CCU60 output	COOUT61	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-16 Port 11 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P11.9	I	General-purpose input	P11_IN.P9	P11_IOCR8. PC9	0XXXX _B
		GTM input	TIN98		
		QSPI1 input	MTSR1B		
		ERAY input	RXDA1		
		ETH input	ETHRXD1		
	O	General-purpose output	P11_OUT.P9		1X000 _B
		GTM output	TOUT98		1X001 _B
		Reserved	–		1X010 _B
		QSPI1 output	MTSR1		1X011 _B
		Reserved	–		1X100 _B
		MSC0 output	SOP0		1X101 _B
		Reserved	–		1X110 _B
		CCU60 output	COUT60		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-16 Port 11 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P11.10	I	General-purpose input	P11_IN.P10	P11_IOC8. PC10	0XXXX _B
		GTM input	TIN99		
		QSPI1 input	SLSI1A		
		CAN node 3 input	RXDCAN3D		
		ERAY input	RXDB1		
		ETH input	ETHRXD0		
		MSC0 input	SDI00		
		ASCLIN1 input	ARX1E		
		SCU input	REQ12		
	O	General-purpose output	P11_OUT.P10	1X000 _B	
		GTM output	TOUT99	1X001 _B	
		Reserved	–	1X010 _B	
		QSPI0 output	SLSO03	1X011 _B	
		QSPI1 output	SLSO13	1X100 _B	
		Reserved	–	1X101 _B	
		Reserved	–	1X110 _B	
CCU60 output		CC62	1X111 _B		
P11.11	I	General-purpose input	P11_IN.P11	P11_IOC8. PC11	0XXXX _B
		GTM input	TIN100		
		ETH input (RMII mode)	ETHCRSDVA		
	O	General-purpose output	P11_OUT.P11	1X000 _B	
		GTM output	TOUT100	1X001 _B	
		MSC0 output	END02	1X010 _B	
		QSPI0 output	SLSO04	1X011 _B	
		QSPI1 output	SLSO14	1X100 _B	
		MSC0 output	EN00	1X101 _B	
		ERAY output	TXENB	1X110 _B	
		CCU60 output	CC61	1X111 _B	

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Table 14-16 Port 11 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P11.12	I	General-purpose input	P11_IN.P12	P11_IOC12. PC12	0XXXX _B
		GTM input	TIN101		
		ETH input	ETHRXCLKA		
		Ethernet input	ETHTXCLKB ¹⁾		
	O	General-purpose output	P11_OUT.P12		1X000 _B
		GTM output	TOUT101		1X001 _B
		ASCLIN1 output	ATX1		1X010 _B
		GTM output	GTMCLK2		1X011 _B
		ERAY output	TXDB	1X100 _B	
		CAN node 3 output	TXDCAN3	1X101 _B	
		SCU output	EXTCLK1	1X110 _B	
		CCU60 output	CC60	1X111 _B	

1) Not for application but for SW development and test

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.7.3 Port 11 Registers

The following registers are available on Port 11:

Table 14-17 Port 11 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P11_OUT	Port 11 Output Register	0000 _H	Page 14-105 2)
P11_OMR	Port 11 Output Modification Register	0004 _H	Page 14-105 2)
P11_IOCRO	Port 11 Input/Output Control Register 0	0010 _H	Page 14-108 2)
P11_IOCRR4	Port 11 Input/Output Control Register 4	0014 _H	Page 14-110 2)
P11_IOCRR8	Port 11 Input/Output Control Register 8	0018 _H	Page 14-112 2)
P11_IOCRR12	Port 11 Input/Output Control Register 12	001C _H	Page 14-114 2)
P11_IN	Port 11 Input Register	0024 _H	Page 14-115 2)
P11_PDR0	Port 11 Pad Driver Mode 0 Register	0040 _H	Page 14-116 2)
P11_PDR1	Port 11 Pad Driver Mode 1 Register	0044 _H	Page 14-117 2)
P11_ESR	Port 11 Emergency Stop Register	0050 _H	Page 14-117 2)
P11_PCSR	Port 11 Pin Controller Select Register	0064 _H	Page 14-118 2)
P11_OMSR0	Port 11 Output Modification Set Register 0	0070 _H	Page 14-106 2)
P11_OMSR4	Port 11 Output Modification Set Register 4	0074 _H	Page 14-106 2)
P11_OMSR8	Port 11 Output Modification Set Register 8	0078 _H	Page 14-106 2)
P11_OMSR12	Port 11 Output Modification Set Register 12	007C _H	Page 14-106 2)

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-17 Port 11 Registers (cont'd)

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P11_OMCR0	Port 11 Output Modification Clear Register 0	0080 _H	Page 14-106 ²⁾
P11_OMCR4	Port 11 Output Modification Clear Register 4	0084 _H	Page 14-107 ²⁾
P11_OMCR8	Port 11 Output Modification Clear Register 8	0088 _H	Page 14-107 ²⁾
P11_OMCR12	Port 11 Output Modification Clear Register 12	008C _H	Page 14-107 ²⁾
P11_OMSR	Port 11 Output Modification Set Register	0090 _H	Page 14-106 ²⁾
P11_OMCR	Port 11 Output Modification Clear Register	0094 _H	Page 14-106 ²⁾
P11_ACCEN1	Port 11 Access Enable Register 1	00F8 _H	Page 14-53
P11_ACCEN0	Port 11 Access Enable Register 0	00FC _H	Page 14-54

1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))

2) These registers are listed and noted here in the Port 11 section because they differ from the general port register description given in [Section 14.3](#).

14.7.3.1 Port 11 Output Register

The basic P10_OUT register functionality is described on [Page 14-36](#). Port lines P11.[15:13], P11.[8:7], P11.[5:4], P11.[1:0] are not connected. Reading the P11_OUT P[15:13], [8:7], [5:4], [1:0] bits return the value that was last written. These connected bits can be also set/reset by the corresponding bits in P10_OMR, P11_OMSR, P11_OMCR, P11_OMSRx (x=0,4,8,12), P11_OMCRx (x=0,4,8,12).

14.7.3.2 Port 11 Output Modification Register

The basic P11_OMR register functionality is described on [Page 14-38](#). However, port lines P11.[15:13], P11.[8:7], P11.[5:4], P11.[1:0] are not connected. The P11_OMR bits PS[15:13], PS[8:7], PS[5:4], PS[1:0] and PCL[15:13], PCL[8:7], PCL[5:4], PCL[1:0] have no direct effect on port lines but only on register bits P11_OUT.P[15:13], P[8:7], P[5:4], P[1:0].

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.7.3.3 Port 11 Output Modification Set Register

The basic P11_OMSR register functionality is described on [Page 14-40](#). However, port lines P11.[15:13], P11.[8:7], P11.[5:4], P11.[1:0] are not connected. The P11_OMSR bits PS[15:13], PS[8:7], PS[5:4], PS[1:0] have no direct effect on port lines but only on register bits P11_OUT.P[15:13], P[8:7],P[5:4],P[1:0].

14.7.3.4 Port 11 Output Modification Set Register 0

The basic P11_OMSR0 register functionality is described on [Page 14-41](#). However, port lines P11.[1:0] are not connected. The P11_OMSR0 bits PS[1:0] have no direct effect on port lines but only on register bits P11_OUT.P[1:0].

14.7.3.5 Port 11 Output Modification Set Register 4

The basic P11_OMSR4 register functionality is described on [Page 14-42](#). However, port lines P11.[5:4], P11.7 are not connected. The P11_OMSR4 bits PS[5:4], PS7 have no direct effect on port lines but only on register bits P11_OUT.P[5:4], P11_OUT.P7.

14.7.3.6 Port 11 Output Modification Set Register 8

The basic P11_OMSR8 register functionality is described on [Page 14-43](#). However, port lines P11.8 is not connected. The P11_OMSR8 bits PS8 has no direct effect on port lines but only on register bits P11_OUT.P8.

14.7.3.7 Port 11 Output Modification Set Register 12

The basic P11_OMSR12 register functionality is described on [Page 14-44](#). However, port lines P11.[15:13] are not connected. The P11_OMSR12 bits PS[15:13] have no direct effect on port lines but only on register bits P11_OUT.P[15:13].

14.7.3.8 Port 11 Output Modification Clear Register

The basic P11_OMCR register functionality is described on [Page 14-45](#). However, port lines P11.[15:13], P11.[8:7], P11.[5:4], P11.[1:0] are not connected. The P11_OMCR bits PCL[15:13], PCL[8:7], PCL[5:4], PCL[1:0] have no direct effect on port lines but only on register bits P11_OUT.P[15:13], P[8:7],P[5:4],P[1:0].

14.7.3.9 Port 11 Output Modification Clear Register 0

The basic P11_OMCR0 register functionality is described on [Page 14-46](#). However, port lines P11.[1:0] are not connected. The P11_OMCR0 bits PCL[1:0] have no direct effect on port lines but only on register bits P11_OUT.P[1:0].

General Purpose I/O Ports and Peripheral I/O Lines (Ports)**14.7.3.10 Port 11 Output Modification Clear Register 4**

The basic P11_OMCR4 register functionality is described on [Page 14-47](#). However, port lines P11.[5:4], P11.7 are not connected. The P11_OMCR4 bits PCL[5:4], PCL7 have no direct effect on port lines but only on register bits P11_OUT.P[5:4], P11_OUT.P7.

14.7.3.11 Port 11 Output Modification Clear Register 8

The basic P11_OMCR8 register functionality is described on [Page 14-48](#). However, port lines P11.8 is not connected. The P11_OMCR8 bits PCL8 has no direct effect on port lines but only on register bits P11_OUT.P8.

14.7.3.12 Port 11 Output Modification Clear Register 12

The basic P11_OMCR12 register functionality is described on [Page 14-49](#). However, port lines P11.[15:13] are not connected. The P11_OMCR12 bits PCL[15:13] have no direct effect on port lines but only on register bits P11_OUT.P[15:13].

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.7.3.13 Port 11 Input/Output Control Register 0

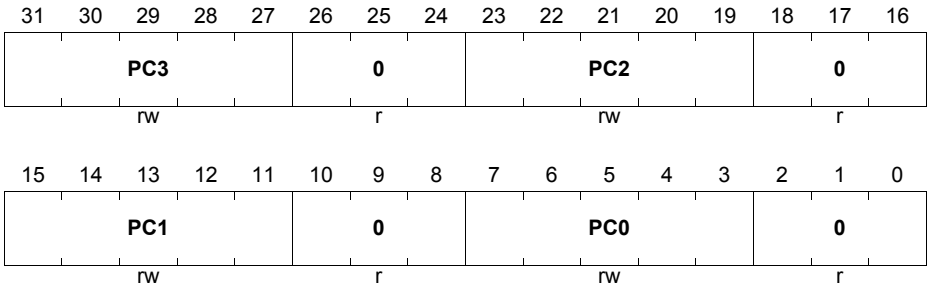
The PC[1:0] bit fields in register P11_IOCRO are not connected.

P11_IOCRO

Port 11 Input/Output Control Register 0

(10_H)

Reset Value: 1010 1010_H

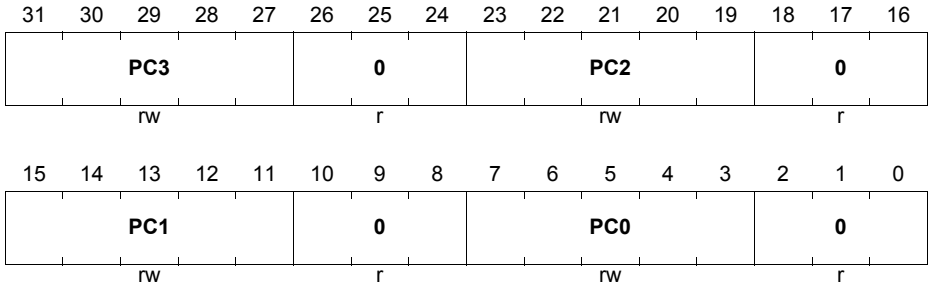


Field	Bits	Type	Description
PC2	[23:19]	rw	Port Control for Port 11 Pin 2
PC3	[31:27]	rw	Port Control for Port 11 Pin 3
PC0, PC1	[7:3], [15:11]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P11_IOCRO
Port 11 Input/Output Control Register 0

 (10_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
PC2	[23:19]	rw	Port Control for Port 11 Pin 2
PC3	[31:27]	rw	Port Control for Port 11 Pin 3
PC0, PC1	[7:3], [15:11]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

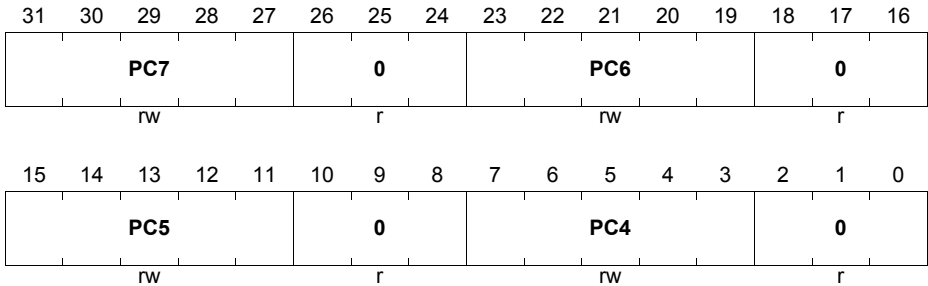
General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.7.3.14 Port 11 Input/Output Control Register 4

The PC[5:4] and PC7 bit fields in register P11_IOCR4 are not connected.

P11_IOCR4
Port 11 Input/Output Control Register 4

 (14_H)

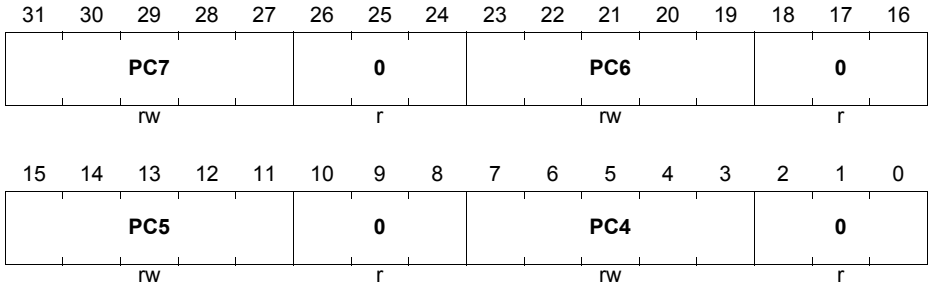
 Reset Value: 1010 1010_H


Field	Bits	Type	Description
PC4, PC5, PC7	[7:3], [15:11], [31:27]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
PC6	[23:19]	rw	Port Control for Port 11 Pin 6 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P11_IOCRA4
Port 11 Input/Output Control Register 4

 (14_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
PC4, PC5, PC7	[7:3], [15:11], [31:27]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC6	[23:19]	rw	Port Control for Port 11 Pin 6 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

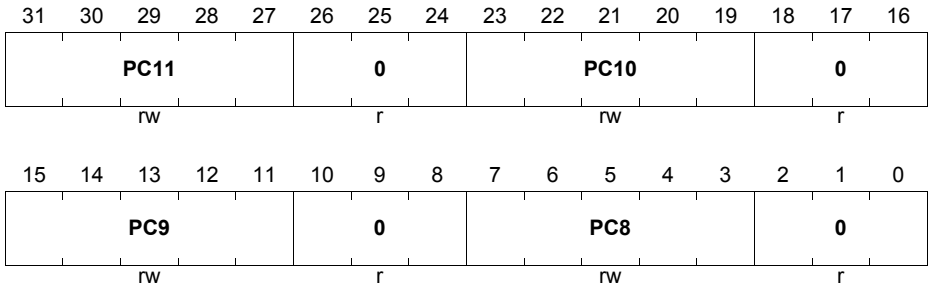
General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.7.3.15 Port 11 Input/Output Control Register 8

The PC8 bit field in register P11_IOC8 is not connected.

P11_IOC8
Port 11 Input/Output Control Register 8

 (18_H)

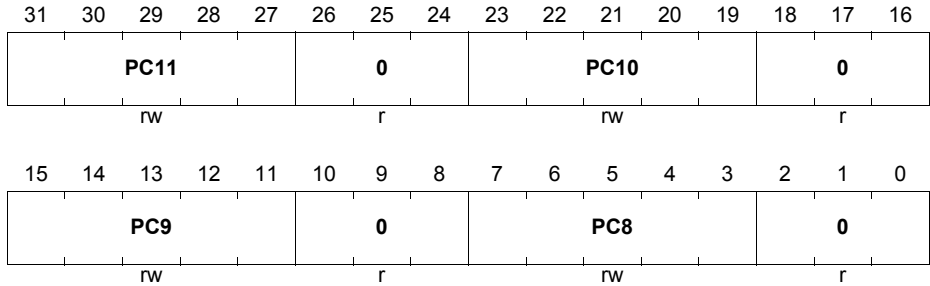
 Reset Value: 1010 1010_H


Field	Bits	Type	Description
PC8	[7:3]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
PC9	[15:11]	rw	Port Control for Port 11 Pin 9 (coding see Table 14-5)
PC10	[23:19]	rw	Port Control for Port 11 Pin10 (coding see Table 14-5)
PC11	[31:27]	rw	Port Control for Port 11 Pin 11 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P11_IOC8
Port 11 Input/Output Control Register 8

 (18_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
PC8	[7:3]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC9	[15:11]	rw	Port Control for Port 11 Pin 9 (coding see Table 14-5)
PC10	[23:19]	rw	Port Control for Port 11 Pin10 (coding see Table 14-5)
PC11	[31:27]	rw	Port Control for Port 11 Pin 11 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

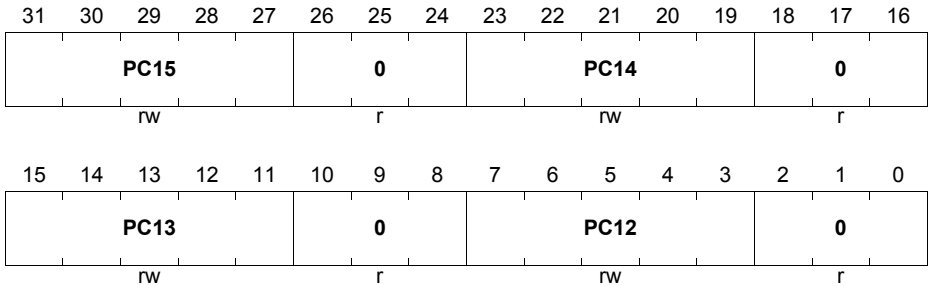
General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.7.3.16 Port 11 Input/Output Control Register 12

The PC13, PC14 and PC15 bit fields in register P00_IOCR12 are not connected.

P11_IOCR12
Port 11 Input/Output Control Register 12

 (1C_H)

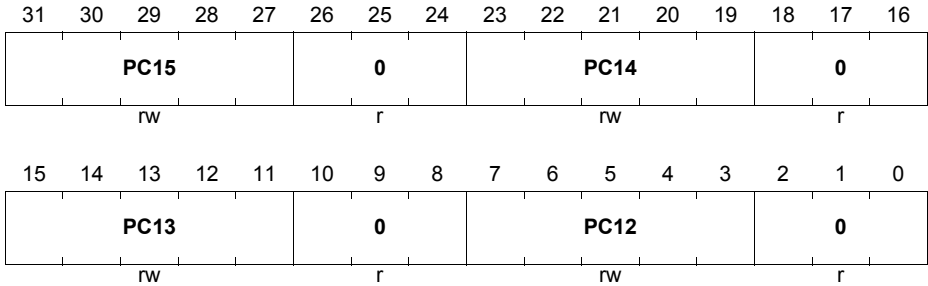
 Reset Value: 1010 1010_H


Field	Bits	Type	Description
PC13, PC14, PC15	[15:11], [23:19], [31:27]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
PC12	[7:3]	rw	Port Control for Port 11.12 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P11_IOC12
Port 11 Input/Output Control Register 12

 (1C_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
PC13, PC14, PC15	[15:11], [23:19], [31:27]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC12	[7:3]	rw	Port Control for Port 11.12 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

14.7.3.17 Port 11 Input Register

The basic P11_IN register functionality is described on [Page 14-52](#). However, port lines P11.[15:13] are not connected. Therefore, bits P[15:9] in register P10_IN are always read as 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.7.3.18 Port 11 Pad Driver Mode 0 Register

The basic P11_PDR0 register functionality is described on [Page 14-25](#).

P11_PDR0
Port 11 Pad Driver Mode 0 Register (40_H)
Reset Value: 3333 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL7	PD7			PL6	PD6			PL5	PD5			PL4	PD4		
rw		rw			rw		rw			rw		rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL3	PD3			PL2	PD2			PL1	PD1			PL0	PD0		
rw		rw			rw		rw			rw		rw			

Field	Bits	Type	Description
PD2	[10:8]	rw	Pad Driver Mode for Port 11 Pin 2
PD3	[14:12]	rw	Pad Driver Mode for Port 11 Pin 3
PD6	[26:24]	rw	Pad Driver Mode for Port 11 Pin 6
PD7	[30:28]	rw	Pad Driver Mode for Port 11 Pin 7
PL2	11	rw	Pad Level Selection for Port 11 Pin 2
PL3	15	rw	Pad Level Selection for Port 11 Pin 3
PL6	27	rw	Pad Level Selection for Port 11 Pin 6
PL7	31	rw	Pad Level Selection for Port 11 Pin 7
PD0, PD1, PD4, PD5	[2:0], [6:4], [18:16], [22:20]	rw	Reserved Read as 000 _B after reset; returns value that was written.
PL0, PL1, PL4, PL5	3, 7, 19, 23	rw	Reserved Read as 0 _B after reset; returns value that was written.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.7.3.19 Port 11 Pad Driver Mode 1 Register

P11_PDR1
Port 11 Pad Driver Mode 1 Register (44_H)
Reset Value: 3333 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL15	PD15		PL14	PD14		PL13	PD13		PL12	PD12					
rw		rw		rw		rw		rw		rw		rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL11	PD11		PL10	PD10		PL9	PD9		PL8	PD8					
rw		rw		rw		rw		rw		rw		rw			

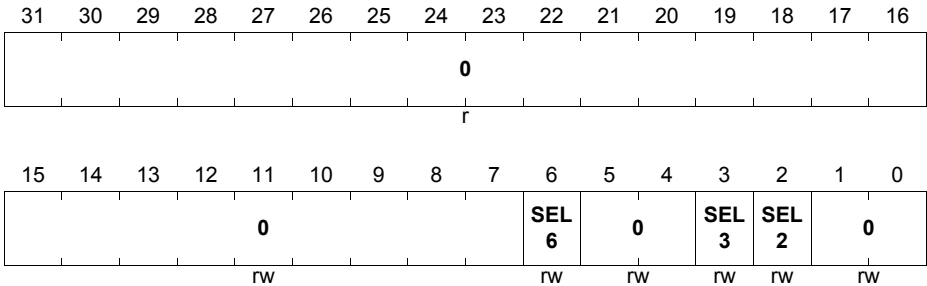
Field	Bits	Type	Description
PD9	[6:4]	rw	Pad Driver Mode for Port 15 Pin 9
PD10	[10:8]	rw	Pad Driver Mode for Port 15 Pin 10
PD11	[14:12]	rw	Pad Driver Mode for Port 15 Pin 11
PD12	[18:16]	rw	Pad Driver Mode for Port 15 Pin 12
PL9	7	rw	Pad Level Selection for Port 15 Pin 9
PL10	11	rw	Pad Level Selection for Port 15 Pin 10
PL11	15	rw	Pad Level Selection for Port 15 Pin 11
PL12	19	rw	Pad Level Selection for Port 15 Pin 12
PD8, PD13, PD14, PD15	[2:0], [22:20], [26:24], [30:28]	rw	Reserved Read as 000 _B after reset; returns value that was written.
PL8, PL13, PL14, PL15	3, 23, 27, 31	rw	Reserved Read as 0 _B after reset; returns value that was written.

14.7.3.20 Port 11 Emergency Stop Register

The basic P11_ESR register functionality is described on [Page 14-50](#). Port lines P11.[15:13], P11.[8:7], P11.[5:4], P11.[1:0] are not connected. Reading the P11_ESR bits EN[15:13], EN[8:7], EN[5:4], EN[1:0] returns value that was last written.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.7.3.21 Port 11 Pin Controller Select Register

P11_PCSR selects Ethernet output signals through ports alternate output or fast RMII mode.

P11_PCSR
Port 11 Pin Controller Select Register (64_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SEL2	2	rw	Pin Controller Select for Pin 2 This bit selects the Ethernet output signal through ports alternate output or fast RMII mode. 0 _B ETHTXD1 output via ports alternate output of pin 2. 1 _B ETHTXD1 output via fast RMII mode of pin 2.
SEL3	3	rw	Pin Controller Select for Pin 3 This bit selects the Ethernet output signal through ports IOCR or fast RMII mode. 0 _B ETHTXD0 output via ports alternate output of pin 3. 1 _B ETHTXD0 output via fast RMII mode of pin 3.
SEL6	6	rw	Pin Controller Select for Pin 6 This bit selects the Ethernet output signal through ports IOCR or fast RMII mode. 0 _B ETHTXEN output via ports alternate output of pin 6. 1 _B ETHTXEN output via fast RMII mode of pin 6.
0	[1:0], [5:4]	rw	Reserved Read as 0; returns value last written, must be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
0	[31:16], [15:7]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.8 Port 13

This section describes the Port 13 functionality.

Port 13 is an 4-bit GPIO port. Pins associated to it be used in two ways:

- as a CMOS Port where each pin outputs one signal, as any other port (only exception - no open drain mode available), and
- as an output LVDS port where a pin pair (two pins) outputs one differential MSC signal.

The switching between the CMOS/LVDS modes is done via the P13_PDR0 register.

The switching between Input/Ouput and Pull-Up/Pull-Down control is done via the IOCR register.

Attention: In the LVDS mode the IOCR.PCx bit field of each pin of the LVDS pair must be programed as output, that is 1xxxx_B.

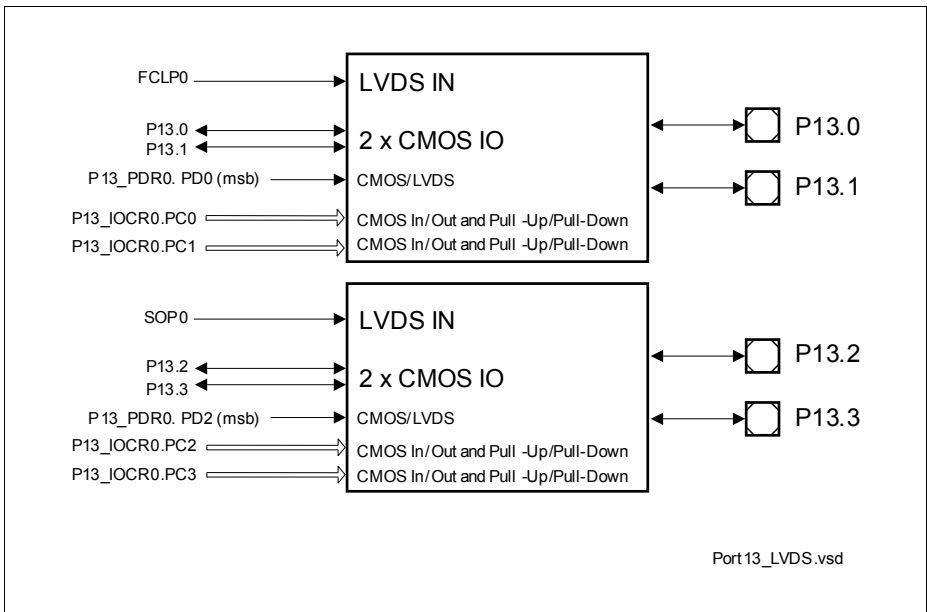


Figure 14-3 Port 13 Pad Connections

14.8.1 Port 13 Configuration

Port 13 is a 4-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, GPT12, QSPI, MSC, I2C and CCU6 functions.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.8.2 Port 13 Function Table

Table 14-18 summarizes the I/O control selection functions of each Port 13 line.

Table 14-18 Port 13 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P13.0	I	General-purpose input	P13_IN.P0	P13_IOCRO. PC0	0XXXX _B
		GTM input	TIN91		
	O	General-purpose output	P13_OUT.P0		1X000 _B
		GTM output	TOUT91		1X001 _B
		MSC0 output	END03		1X010 _B
		QSPI2 output	SCLK2N		1X011 _B
		MSC0 output	EN01		1X100 _B
		MSC0 output	FCLN0		1X101 _B
		MSC0 output	FCLND0		1X110 _B
		Reserved	–		1X111 _B
		CAN node 4 output	TXDCAN4		1X111 _B
P13.1	I	General-purpose input	P13_IN.P1	P13_IOCRO. PC1	0XXXX _B
		GTM input	TIN92		
		I2C0 input	SCL0B		
		CAN node 4 input	RXDCAN4C		
	O	General-purpose output	P13_OUT.P1		1X000 _B
		GTM output	TOUT92		1X001 _B
		Reserved	–		1X010 _B
		QSPI2 output	SCLK2P		1X011 _B
		Reserved	–		1X100 _B
		MSC0 output	FCLP0		1X101 _B
		I2C0 output	SCL0		1X110 _B
		Reserved	–		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-18 Port 13 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.			
				Reg./Bit Field	Value		
P13.2	I	General-purpose input	P13_IN.P2	P13_IOCR0. PC2	0XXXX _B		
		GTM input	TIN93				
		GPT120 input	CAPINA				
		I2C0 input	SDA0B				
	O	General-purpose output	P13_OUT.P2		1X000 _B		
		GTM output	TOUT93		1X001 _B		
		Reserved	–		1X010 _B		
		QSPI2 output	MTSR2N		1X011 _B		
		MSC0 output	FCLP0		1X100 _B		
		MSC0 output	SON0		1X101 _B		
		I2C0 output	SDA0		1X110 _B		
		MSC0 output	SOND0		1X111 _B		
	P13.3	I	General-purpose input		P13_IN.P3	P13_IOCR0. PC3	0XXXX _B
			GTM input		TIN94		
O		General-purpose output	P13_OUT.P3	1X000 _B			
		GTM output	TOUT94	1X001 _B			
		Reserved	–	1X010 _B			
		QSPI2 output	MTSR2P	1X011 _B			
		Reserved	–	1X100 _B			
		MSC0 output	SOP0	1X101 _B			
		Reserved	–	1X110 _B			
		Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.8.3 Port 13 Registers

The following registers are available on Port 13:

Table 14-19 Port 13 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P13_OUT	Port 13 Output Register	0000 _H	Page 14-124 ²⁾
P13_OMR	Port 13 Output Modification Register	0004 _H	Page 14-124 ²⁾
P13_IOCRO	Port 13 Input/Output Control Register 0	0010 _H	Page 14-14
P13_IN	Port 13 Input Register	0024 _H	Page 14-124 ²⁾
P13_PDR0	Port 13 Pad Driver Mode 0 Register	0040 _H	Page 14-125 ²⁾
P13_ESR	Port 13 Emergency Stop Register	0050 _H	Page 14-128 ²⁾
P13_OMSR0	Port 13 Output Modification Set Register 0	0070 _H	Page 14-41
P13_OMCR0	Port 13 Output Modification Clear Register 0	0080 _H	Page 14-46
P13_OMSR	Port 13 Output Modification Set Register	0090 _H	Page 14-124 ²⁾
P13_OMCR	Port 13 Output Modification Clear Register	0094 _H	Page 14-124 ²⁾
P13_LPCR0	Port 13 LVDS Pad Control Register 0	00A0 _H	Page 14-127 ²⁾
P13_LPCR1	Port 13 LVDS Pad Control Register 1	00A4 _H	Page 14-127 ²⁾
P13_ACCEN1	Port 13 Access Enable Register 1	00F8 _H	Page 14-54
P13_ACCEN0	Port 13 Access Enable Register 0	00FC _H	Page 14-53

1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))

2) These registers are listed and noted here in the Port 13 section because they differ from the general port register description given in [Section 14.3](#).

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.8.3.1 Port 13 Output Register

The basic P13_OUT register functionality is described on [Page 14-36](#). Port line P13.[15:4] are not connected. Reading the P13_OUT bit P[15:4] always return 0. These connected bits can be also set/reset by the corresponding bits in P13_OMR, P13_OMSR, P13_OMCR, P13_OMSR0, P13_OMCR0.

14.8.3.2 Port 13 Output Modification Register

The basic P13_OMR register functionality is described on [Page 14-38](#). However, port lines P13.[15:4] are not connected.

14.8.3.3 Port 13 Output Modification Set Register

The basic P13_OMSR register functionality is described on [Page 14-40](#). However, port lines P13.[15:4] are not connected.

14.8.3.4 Port 13 Output Modification Clear Register

The basic P13_OMCR register functionality is described on [Page 14-45](#). However, port lines P13.[15:4] are not connected.

14.8.3.5 Port 13 Input Register

The basic P13_IN register functionality is described on [Page 14-52](#). However, port lines P13.[15:4] are not connected. Therefore, bits P[15:4] in register P13_IN are always read as 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.8.3.6 Port 13 Pad Driver Mode 0 Register

 The basic P13_PDR0 register functionality is described on [Page 14-25](#).

P13_PDR0
Port 13 Pad Driver Mode 0 Register (40_H)
Reset Value: 0000 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL7		PD7		PL6		PD6		PL5		PD5		PL4		PD4	
r		r		r		r		r		r		r		r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL3		PD3		PL2		PD2		PL1		PD1		PL0		PD0	
rw		rw		rw		rw		rw		rw		rw		rw	

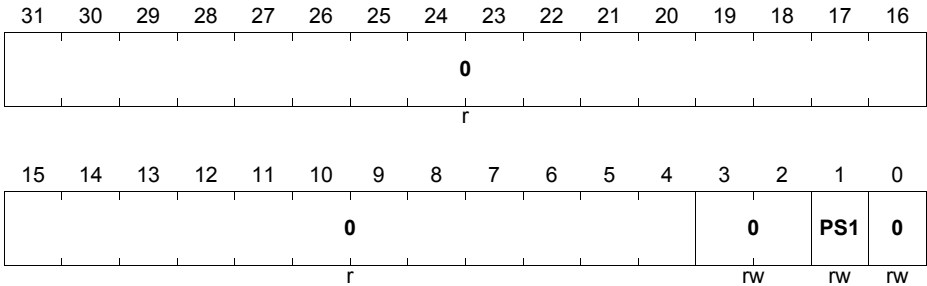
Field	Bits	Type	Description
PD0	[2:0]	rw	Pad Driver Mode for Port 13 Pin [1:0] The MSB of PDx switches between CMOS (default) and LVDS modes. 0XX _B CMOS input or output (depending on IOCRx setting). 1XX _B LVDS output (pull-up or pull-down must be switched off via IOCRx)
PD2	[10:8]	rw	Pad Driver Mode for Port 13 Pin [3:2] The MSB of PDx switches between CMOS (default) and LVDS modes. 0XX _B CMOS input or output (depending on IOCRx setting). 1XX _B LVDS output (pull-up or pull-down must be switched off via IOCRx)
PD1, PD3	[6:4], [14:12]	rw	Pad Driver Mode for Port 13 Pin 1 and 3
PL0, PL1, PL2, PL3	3, 7, 11, 15	rw	Pad Level Selection for Port 13 Pin 0 to 3
PD4, PD5, PD6, PD7	[18:16], [22:20], [26:24], [30:28]	r	Reserved Read as 000 _B after reset; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PL4, PL5, PL6, PL7	19, 23, 27, 31	r	Reserved Read as 0 _B after reset; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.8.3.7 Port 13 LVDS Pad Control Register

For P13_LPCR_x($x=0-1$), PS1 needs to be configured to the intended pad supply level before the LVDSM function is enabled.

P13_LPCR0
Port 13 LVDS Pad Control Register 0 (A0_H)
Reset Value: 0000 0000_H


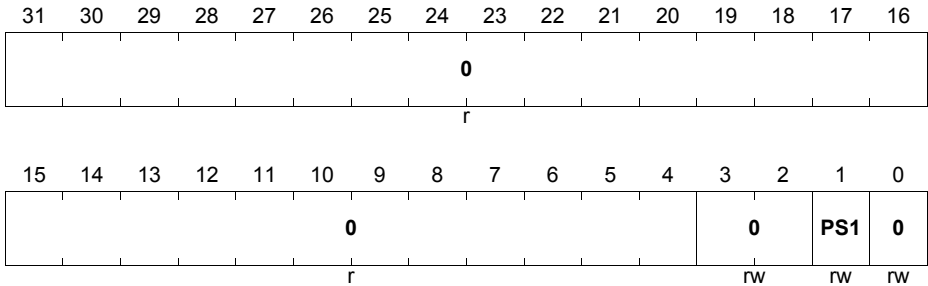
Field	Bits	Type	Description
PS1	1	rw	Pad Supply for pins [1:0] Selects between 5V or 3.3V supply on V_{EXT} for the LVDSM pad-pair. 0_B 5V supply 1_B 3.3V supply
0	[3:2], 0	rw	Reserved Read as 0; must be written with 0.
0	[31:4]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P13_LPCR1

Port 13 LVDS Pad Control Register 1 (A4_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
PS1	1	rw	Pad Supply for pins [3:2] Selects between 5V or 3.3V supply on V_{EXT} for the LVDSM pad-pair. 0 _B 5V supply 1 _B 3.3V supply
0	[3:2], 0	rw	Reserved Read as 0; must be written with 0.
0	[31:4]	r	Reserved Read as 0; should be written with 0.

14.8.3.8 Port 13 Emergency Stop Register

The basic P13_ESR register functionality is described on [Page 14-50](#). Port lines P13.[15:4] are not connected. Reading the P13_ESR bits EN[15:4] always returns 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.9 Port 14

This section describes the Port 14 functionality.

14.9.1 Port 14 Configuration

Port 14 is a 11-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, ASCLIN, ERAY, MultiCAN+, CCU6, SCU, QSPI and MSC functions.

After the required state is latched in based on HWCFGx during reset, the corresponding pins can be used as GPIO. For the different start-up configuration, please refer to the firmware and SCU chapters.

14.9.2 Port 14 Function Table

Table 14-20 summarizes the I/O control selection functions of each Port 14 line.

Table 14-20 Port 14 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P14.0	I	General-purpose input	P14_IN.P0	P14_IOCRO. PC0	0XXXX _B
		GTM input	TIN80		
	O	General-purpose output	P14_OUT.P0		1X000 _B
		GTM output	TOUT80		1X001 _B
		ASCLIN0 output	ATX0		1X010 _B
		ERAY output	TXDA		1X011 _B
		ERAY output	TXDB		1X100 _B
		CAN node 1 output	TXDCAN1		1X101 _B
		ASCLIN0 output	ASCLK0		1X110 _B
		CCU60 output	COU62		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-20 Port 14 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P14.1	I	General-purpose input	P14_IN.P1	P14_IOCRO.PC1	0XXXX _B
		GTM input	TIN81		
		ASCLIN0 input	ARX0A		
		CAN node 1 input	RXDCAN1B		
		ERAY input	RXDA3		
		ERAY input	RXDB3		
		SCU input	EVRWUPA		
		SCU input	REQ15		
	O	General-purpose output	P14_OUT.P1	1X000 _B	
		GTM output	TOUT81	1X001 _B	
		ASCLIN0 output	ATX0	1X010 _B	
		Reserved	–	1X011 _B	
		Reserved	–	1X100 _B	
		Reserved	–	1X101 _B	
		Reserved	–	1X110 _B	
CCU60 output	COOUT63	1X111 _B			
P14.2	I	General-purpose input	P14_IN.P2	P14_IOCRO.PC2	0XXXX _B
		GTM input	TIN82		
		SCU input	HWCFG2_EVR13		
	O	General-purpose output	P14_OUT.P2	1X000 _B	
		GTM output	TOUT82	1X001 _B	
		ASCLIN2 output	ATX2	1X010 _B	
		QSPI2 output	SLSO21	1X011 _B	
		Reserved	–	1X100 _B	
		Reserved	–	1X101 _B	
		ASCLIN2 output	ASCLK2	1X110 _B	
		Reserved	–	1X111 _B	

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-20 Port 14 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
P14.3	I	General-purpose input	P14_IN.P3	P14_IOCRR0. PC3	0XXXX _B	
		GTM input	TIN83			
		ASCLIN2 input	ARX2A			
		SCU input	REQ10			
		SCU input	HWCFG3_BMI			
		MSC0 input	SDI02			
	O	General-purpose output	P14_OUT.P3			1X000 _B
		GTM output	TOUT83			1X001 _B
		ASCLIN2 output	ATX2			1X010 _B
		QSPI2 output	SLSO23			1X011 _B
		ASCLIN1 output	ASLSO1			1X100 _B
		ASCLIN3 output	ASLSO3			1X101 _B
		Reserved	–			1X110 _B
		Reserved	–			1X111 _B
P14.4	I	General-purpose input	P14_IN.P4	P14_IOCRR4. PC4	0XXXX _B	
		GTM input	TIN84			
		SCU input	HWCFG6			
	O	General-purpose output	P14_OUT.P4			1X000 _B
		GTM output	TOUT84			1X001 _B
		Reserved	–			1X010 _B
		Reserved	–			1X011 _B
		Reserved	–			1X100 _B
		Reserved	–			1X101 _B
		Reserved	–			1X110 _B
		Reserved	–			1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-20 Port 14 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./I/O Line	Port I/O Control Select.				
				Reg./Bit Field	Value			
P14.5	I	General-purpose input	P14_IN.P5	P14_IOCRA4. PC5	0XXXX _B			
		GTM input	TIN85					
		SCU input	HWCFG1_EVR33					
	O	General-purpose output	P14_OUT.P5			1X000 _B		
		GTM output	TOUT85			1X001 _B		
		Reserved	–			1X010 _B		
		Reserved	–			1X011 _B		
		Reserved	–			1X100 _B		
		Reserved	–			1X101 _B		
		ERAY output	TXDB			1X110 _B		
		Reserved	–			1X111 _B		
	P14.6	I	General-purpose input			P14_IN.P6	P14_IOCRA4. PC6	0XXXX _B
			GTM input			TIN86		
			SCU input			HWCFG0_DCLDO		
O		General-purpose output	P14_OUT.P6	1X000 _B				
		GTM output	TOUT86	1X001 _B				
		Reserved	–	1X010 _B				
		QSPI2 output	SLSO22	1X011 _B				
		Reserved	–	1X100 _B				
		Reserved	–	1X101 _B				
		ERAY output	TXENB	1X110 _B				
		Reserved	–	1X111 _B				

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-20 Port 14 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./I/O Line	Port I/O Control Select.			
				Reg./Bit Field	Value		
P14.7	I	General-purpose input	P14_IN.P7	P14_IOCR4. PC7	0XXXX _B		
		GTM input	TIN87				
		ERAY input	RXDB0				
	O	General-purpose output	P14_OUT.P7		1X000 _B		
		GTM output	TOUT87		1X001 _B		
		ASCLIN0 output	ARTS0		1X010 _B		
		QSPI2 output	SLSO24		1X011 _B		
		Reserved	–		1X100 _B		
		Reserved	–		1X101 _B		
		Reserved	–		1X110 _B		
		Reserved	–		1X111 _B		
	P14.8	I	General-purpose input		P14_IN.P8	P14_IOCR8. PC8	0XXXX _B
			GTM input		TIN88		
			ASCLIN1 input		ARX1D		
CAN node 2 input			RXDCAN2D				
ERAY input			RXDA0				
O		General-purpose output	P14_OUT.P8	1X000 _B			
		GTM output	TOUT88	1X001 _B			
		Reserved	–	1X010 _B			
		Reserved	–	1X011 _B			
		Reserved	–	1X100 _B			
		Reserved	–	1X101 _B			
		Reserved	–	1X110 _B			
		Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-20 Port 14 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./I/O Line	Port I/O Control Select.			
				Reg./Bit Field	Value		
P14.9	I	General-purpose input	P14_IN.P9	P14_IOC8. PC9	0XXXX _B		
		GTM input	TIN89				
		ASCLIN0 input	ACTS0A				
	O	General-purpose output	P14_OUT.P9		1X000 _B		
		GTM output	TOUT89		1X001 _B		
		MSC0 output	END03		1X010 _B		
		MSC0 output	EN01		1X011 _B		
		Reserved	–		1X100 _B		
		ERAY output	$\overline{\text{TXENB}}$		1X101 _B		
		ERAY output	$\overline{\text{TXENA}}$		1X110 _B		
		Reserved	–		1X111 _B		
	P14.10	I	General-purpose input		P14_IN.P10	P14_IOC8. PC10	0XXXX _B
			GTM input		TIN90		
		O	General-purpose output		P14_OUT.P10		1X000 _B
GTM output			TOUT90	1X001 _B			
MSC0 output			END02	1X010 _B			
MSC0 output			EN00	1X011 _B			
ASCLIN1 output			ATX1	1X100 _B			
CAN node 2 output			TXDCAN2	1X101 _B			
ERAY output			TXDA	1X110 _B			
Reserved			–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.9.3 Port 14 Registers

The following registers are available on Port 14:

Table 14-21 Port 14 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P14_OUT	Port 14 Output Register	0000 _H	Page 14-136 ²⁾
P14_OMR	Port 14 Output Modification Register	0004 _H	Page 14-136 ²⁾
P14_IOCRO	Port 14 Input/Output Control Register 0	0010 _H	Page 14-14
P14_IOCRR4	Port 14 Input/Output Control Register 4	0014 _H	Page 14-17
P14_IOCRR8	Port 14 Input/Output Control Register 8	0018 _H	Page 14-137 ²⁾
P14_IN	Port 14 Input Register	0024 _H	Page 14-138 ²⁾
P14_PDR0	Port 14 Pad Driver Mode 0 Register	0040 _H	Page 14-25
P14_PDR1	Port 14 Pad Driver Mode 1 Register	0044 _H	Page 14-139 ²⁾
P14_ESR	Port 14 Emergency Stop Register	0050 _H	Page 14-140 ²⁾
P14_OMSR0	Port 14 Output Modification Set Register 0	0070 _H	Page 14-41
P14_OMSR4	Port 14 Output Modification Set Register 4	0074 _H	Page 14-42
P14_OMSR8	Port 14 Output Modification Set Register 8	0078 _H	Page 14-136 ²⁾
P14_OMCR0	Port 14 Output Modification Clear Register 0	0080 _H	Page 14-46
P14_OMCR4	Port 14 Output Modification Clear Register 4	0084 _H	Page 14-47
P14_OMCR8	Port 14 Output Modification Clear Register 8	0088 _H	Page 14-136 ²⁾
P14_OMSR	Port 14 Output Modification Set Register	0090 _H	Page 14-136 ²⁾
P14_OMCR	Port 14 Output Modification Clear Register	0094 _H	Page 14-136 ²⁾
P14_ACCEN1	Port 14 Access Enable Register 1	00F8 _H	Page 14-54
P14_ACCEN0	Port 14 Access Enable Register 0	00FC _H	Page 14-53

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

- 1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))
- 2) These registers are listed and noted here in the Port 14 section because they differ from the general port register description given in [Section 14.3](#).

14.9.3.1 Port 14 Output Register

The basic P14_OUT register functionality is described on [Page 14-36](#). Port lines P14.[15:11] are not connected. Reading the P14_OUT bits P[15:12] always return 0. Reading P11 returns the value that was last written. These connected bits can be also set/reset by the corresponding bits in P14_OMR, P14_OMSR, P14_OMCR, P14_OMSRx (x=0,4,8), P14_OMCRx (x=0,4,8).

14.9.3.2 Port 14 Output Modification Register

The basic P14_OMR register functionality is described on [Page 14-38](#). However, port lines P14.[15:11] are not connected. The P14_OMR bits PS11 and PCL11 has no direct effect on port lines but only on register bits P14_OUT.P11.

14.9.3.3 Port 14 Output Modification Set Register

The basic P14_OMSR register functionality is described on [Page 14-40](#). However, port lines P14.[15:11] are not connected. The P14_OMSR bits PS11 has no direct effect on port lines but only on register bits P14_OUT.P11.

14.9.3.4 Port 14 Output Modification Set Register 8

The basic P14_OMSR8 register functionality is described on [Page 14-43](#). However, port lines P14.11 is not connected. The P14_OMSR8 bit PS11 has no direct effect on port line but only on register bit P14_OUT.P11.

14.9.3.5 Port 14 Output Modification Clear Register

The basic P14_OMCR register functionality is described on [Page 14-45](#). However, port lines P14.11 are not connected. The P14_OMCR bits PCL11 has no direct effect on port lines but only on register bits P14_OUT.P11.

14.9.3.6 Port 14 Output Modification Clear Register 8

The basic P14_OMCR8 register functionality is described on [Page 14-48](#). However, port lines P14.11 is not connected. The P14_OMCR8 bit PCL11 has no direct effect on port lines but only on register bit P14_OUT.P11.

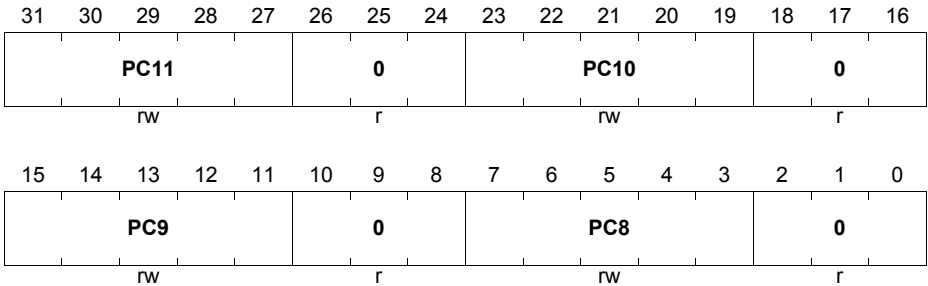
General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.9.3.7 Port 14 Input/Output Control Register 8

The PC11 bit field in register P14_IOC8 is not connected.

P14_IOC8
Port 14 Input/Output Control Register 8

 (18_H)

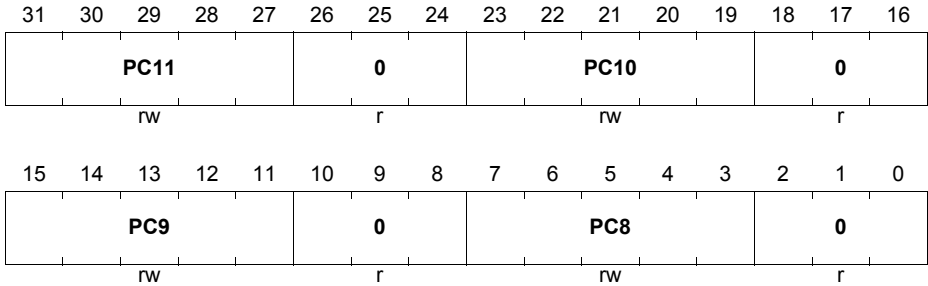
 Reset Value: 1010 1010_H


Field	Bits	Type	Description
PC11	[31:27]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
PC8, PC9, PC10	[7:3], [15:11], [23:19]	rw	Port Control for Port 14 Pin 8 to 10 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P14_IOC8
Port 14 Input/Output Control Register 8

 (18_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
PC11	[31:27]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC8, PC9, PC10	[7:3], [15:11], [23:19]	rw	Port Control for Port 14 Pin 8 to 10 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

14.9.3.8 Port 14 Input Register

The basic P14_IN register functionality is described on [Page 14-52](#). However, port lines P14.[15:11] are not connected. Therefore, bits P[15:11] in register P14_IN are always read as 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.9.3.9 Port 14 Pad Driver Mode 1 Register

P14_PDR1
Port 14 Pad Driver Mode 1 Register (44_H)
Reset Value: 0000 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL15	PD15		PL14	PD14		PL13	PD13		PL12	PD12					
r	r		r	r		r	r		r	r		r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL11	PD11		PL10	PD10		PL9	PD9		PL8	PD8					
rw	rw		rw	rw		rw	rw		rw	rw		rw			

Field	Bits	Type	Description
PD8, PD9, PD10	[2:0], [6:4], [10:8]	rw	Pad Driver Mode for Port 14 Pin 8 to 10
PL8, PL9, PL10	3, 7, 11	rw	Pad Level Selection for Port 14 Pin 8 to 10
PD11	[14:12]	rw	Reserved Read as 000 _B after reset; returns value that was written.
PD12, PD13, PD14, PD15	[18:16], [22:20], [26:24], [30:28]	r	Reserved Read as 000 _B after reset; should be written with 0.
PL11	15	rw	Reserved Read as 0 _B after reset; returns value that was written.
PL12, PL13, PL14, PL15	19, 23, 27, 31	r	Reserved Read as 0 _B after reset; should be written with 0.

14.9.3.10 Port 14 Emergency Stop Register

The basic P14_ESR register functionality is described on [Page 14-50](#). Port lines P14.[15:11] are not connected. Reading the P14_ESR bits EN[15:12] always returns 0. Reading P14_ESR bits EN11 returns value that was last written.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.10 Port 15

This section describes the Port 15 functionality.

14.10.1 Port 15 Configuration

Port 15 is a 9-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, ASCLIN, MultiCAN+, SCU, QSPI, CCU6 and I2C functions.

14.10.2 Port 15 Function Table

Table 14-22 summarizes the I/O control selection functions of each Port 15 line.

Table 14-22 Port 15 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P15.0	I	General-purpose input	P15_IN.P0	P15_IOCRO. PC0	0XXXX _B
		GTM input	TIN71		
	O	General-purpose output	P15_OUT.P0		1X000 _B
		GTM output	TOUT71		1X001 _B
		ASCLIN1 output	ATX1		1X010 _B
		QSPI0 output	SLSO013		1X011 _B
		Reserved	–		1X100 _B
		CAN node 2 output	TXDCAN2		1X101 _B
		ASCLIN1 output	ASCLK1		1X110 _B
Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-22 Port 15 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
P15.1	I	General-purpose input	P15_IN.P1	P15_IOCRO. PC1	0XXXX _B	
		GTM input	TIN72			
		ASCLIN1 input	ARX1A			
		CAN node 2 input	RXDCAN2A			
		SCU input	EVRWUPB			
		QSPI2 input	SLSI2B			
		SCU input	REQ16			
	O	General-purpose output	P15_OUT.P1			1X000 _B
		GTM output	TOUT72			1X001 _B
		ASCLIN1 output	ATX1			1X010 _B
		QSPI2 output	SLSO25			1X011 _B
		Reserved	–			1X100 _B
		Reserved	–			1X101 _B
		Reserved	–			1X110 _B
Reserved	–	1X111 _B				
P15.2	I	General-purpose input	P15_IN.P2	P15_IOCRO. PC2	0XXXX _B	
		GTM input	TIN73			
		QSPI2 input	SLSI2A			
		QSPI2 input	MRST2E			
		QSPI2 input	HSIC2INA			
	O	General-purpose output	P15_OUT.P2			1X000 _B
		GTM output	TOUT73			1X001 _B
		ASCLIN0 output	ATX0			1X010 _B
		QSPI2 output	SLSO20			1X011 _B
		Reserved	–			1X100 _B
		CAN node 1 output	TXDCAN1			1X101 _B
		ASCLIN0 output	ASCLK0			1X110 _B
		Reserved	–			1X111 _B
		Reserved	–			–

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-22 Port 15 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
P15.3	I	General-purpose input	P15_IN.P3	P15_IOCRO. PC3	0XXXX _B	
		GTM input	TIN74			
		ASCLIN0 input	ARX0B			
		QSPI2 input	SCLK2A			
		CAN node 1 input	RXDCAN1A			
		QSPI2 input	HSIC2INB			
	O	General-purpose output	P15_OUT.P3			1X000 _B
		GTM output	TOUT74			1X001 _B
		ASCLIN0 output	ATX0			1X010 _B
		QSPI2 output	SCLK2			1X011 _B
		MSC0 output	END03			1X100 _B
		MSC0	EN01			1X101 _B
		Reserved	–			1X110 _B
		Reserved	–			1X111 _B
P15.4	I	General-purpose input	P15_IN.P4	P15_IOCRO4. PC4	0XXXX _B	
		GTM input	TIN75			
		QSPI2 input	MRST2A			
		SCU input	REQ0			
		I2C0 input	SCL0C			
		O	General-purpose output			P15_OUT.P4
	GTM output		TOUT75			1X001 _B
	ASCLIN1 output		ATX1			1X010 _B
	QSPI2 output		MRST2			1X011 _B
	Reserved		–			1X100 _B
	Reserved		–			1X101 _B
	I2C0 output		SCL0			1X110 _B
	CCU60 output		CC62			1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-22 Port 15 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P15.5	I	General-purpose input	P15_IN.P5	P15_IOCRA. PC5	0XXXX _B
		GTM input	TIN76		
		ASCLIN1 input	ARX1B		
		QSPI2 input	MTRSR2A		
		I2C0 input	SDA0C		
		SCU input	REQ13		
	O	General-purpose output	P15_OUT.P5		1X000 _B
		GTM output	TOUT76		1X001 _B
		ASCLIN1 output	ATX1		1X010 _B
		QSPI2 output	MTRSR2		1X011 _B
		MSC0 output	END02		1X100 _B
		MSC0	EN00		1X101 _B
		I2C0 output	SDA0		1X110 _B
		CCU60 output	CC61		1X111 _B
P15.6	I	General-purpose input	P15_IN.P6	P15_IOCRA. PC6	0XXXX _B
		GTM input	TIN77		
		QSPI2 input	MTRSR2B		
	O	General-purpose output	P15_OUT.P6		1X000 _B
		GTM output	TOUT77		1X001 _B
		ASCLIN3 output	ATX3		1X010 _B
		QSPI2 output	MTRSR2		1X011 _B
		Reserved	–		1X100 _B
		QSPI2 output	SCLK2		1X101 _B
		ASCLIN3 output	ASCLK3		1X110 _B
		CCU60 output	CC60		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-22 Port 15 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.				
				Reg./Bit Field	Value			
P15.7	I	General-purpose input	P15_IN.P7	P15_IOCRA. PC7	0XXXX _B			
		GTM input	TIN78					
		ASCLIN3 input	ARX3A					
		QSPI2 input	MRST2B					
	O	General-purpose output	P15_OUT.P7			1X000 _B		
		GTM output	TOUT78			1X001 _B		
		ASCLIN3 output	ATX3			1X010 _B		
		QSPI2 output	MRST2			1X011 _B		
		Reserved	–			1X100 _B		
		Reserved	–			1X101 _B		
		Reserved	–			1X110 _B		
		CCU60 output	COOUT60			1X111 _B		
	P15.8	I	General-purpose input			P15_IN.P8	P15_IOCRA. PC8	0XXXX _B
			GTM input			TIN79		
QSPI2 input			SCLK2B					
SCU input			REQ1					
O		General-purpose output	P15_OUT.P8	1X000 _B				
		GTM output	TOUT79	1X001 _B				
		Reserved	–	1X010 _B				
		QSPI2 output	SCLK2	1X011 _B				
		Reserved	–	1X100 _B				
		Reserved	–	1X101 _B				
		ASCLIN3 output	ASCLK3	1X110 _B				
		CCU60 output	COOUT61	1X111 _B				

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.10.3 Port 15 Registers

The following registers are available on Port 15:

Table 14-23 Port 15 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P15_OUT	Port 15 Output Register	0000 _H	Page 14-147 2)
P15_OMR	Port 15 Output Modification Register	0004 _H	Page 14-147 2)
P15_IOCRO	Port 15 Input/Output Control Register 0	0010 _H	Page 14-14
P15_IOCRR4	Port 15 Input/Output Control Register 4	0014 _H	Page 14-17
P15_IOCRR8	Port 15 Input/Output Control Register 8	0018 _H	Page 14-148 2)
P15_IN	Port 15 Input Register	0024 _H	Page 14-149 2)
P15_PDR0	Port 15 Pad Driver Mode 0 Register	0040 _H	Page 14-25
P15_PDR1	Port 15 Pad Driver Mode 1 Register	0044 _H	Page 14-150 2)
P15_ESR	Port 15 Emergency Stop Register	0050 _H	Page 14-150 2)
P15_OMSR0	Port 15 Output Modification Set Register 0	0070 _H	Page 14-41
P15_OMSR4	Port 15 Output Modification Set Register 4	0074 _H	Page 14-42
P15_OMSR8	Port 15 Output Modification Set Register 8	0078 _H	Page 14-147 2)
P15_OMCR0	Port 15 Output Modification Clear Register 0	0080 _H	Page 14-46
P15_OMCR4	Port 15 Output Modification Clear Register 4	0084 _H	Page 14-47
P15_OMCR8	Port 15 Output Modification Clear Register 8	0088 _H	Page 14-147 2)
P15_OMSR	Port 15 Output Modification Set Register	0090 _H	Page 14-147 2)
P15_OMCR	Port 15 Output Modification Clear Register	0094 _H	Page 14-147 2)
P15_ACCEN1	Port 15 Access Enable Register 1	00F8 _H	Page 14-54
P15_ACCEN0	Port 15 Access Enable Register 0	00FC _H	Page 14-53

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

- 1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))
- 2) These registers are listed and noted here in the Port 15 section because they differ from the general port register description given in [Section 14.3](#).

14.10.3.1 Port 15 Output Register

The basic P15_OUT register functionality is described on [Page 14-36](#). Port lines P15.[15:9] are not connected. Reading the P15_OUT bits P[15:12] always return 0, reading P[11:9] returns the value that was last written. These connected bits can be also set/reset by the corresponding bits in P15_OMR, P15_OMSR, P15_OMCR, P15_OMSRx (x=0,4,8), P15_OMCRx (x=0,4,8).

14.10.3.2 Port 15 Output Modification Register

The basic P15_OMR register functionality is described on [Page 14-38](#). However, port lines P15.[15:9] are not connected. The P15_OMR bits PS[11:9] and PCL[11:9] have no direct effect on port lines but only on register bits P15_OUT.P[11:9].

14.10.3.3 Port 15 Output Modification Set Register

The basic P15_OMSR register functionality is described on [Page 14-40](#). However, port lines P15.[15:9] are not connected. The P15_OMSR bits PS[11:9] have no direct effect on port lines but only on register bits P15_OUT.P[11:9].

14.10.3.4 Port 15 Output Modification Set Register 8

The basic P15_OMSR8 register functionality is described on [Page 14-43](#). However, port lines P15.[11:9] are not connected. The P15_OMSR8 bits PS[11:9] have no direct effect on port lines but only on register bits P15_OUT.P[11:9].

14.10.3.5 Port 15 Output Modification Clear Register

The basic P15_OMCR register functionality is described on [Page 14-45](#). However, port lines P15.[15:9] are not connected. The P15_OMCR bits PCL[11:9] have no direct effect on port lines but only on register bits P15_OUT.P[11:9].

14.10.3.6 Port 15 Output Modification Clear Register 8

The basic P15_OMCR8 register functionality is described on [Page 14-48](#). However, port lines P15.[11:9] are not connected. The P15_OMCR8 bits PCL[11:9] have no direct effect on port lines but only on register bits P15_OUT.P[11:9].

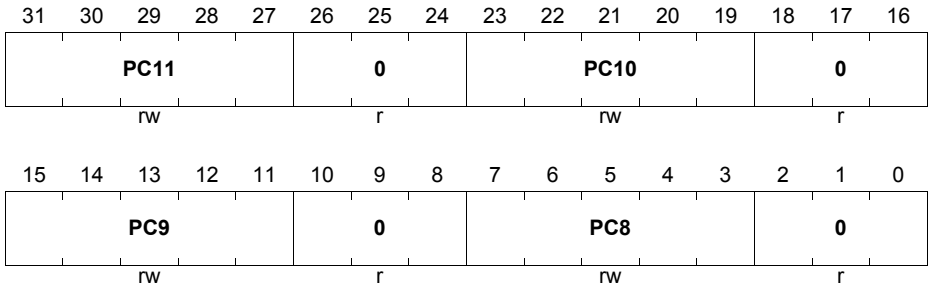
General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.10.3.7 Port 15 Input/Output Control Register 8

The PC[11:9] bit fields in register P15_IOC8 are not connected.

P15_IOC8
Port 15 Input/Output Control Register 8

 (18_H)

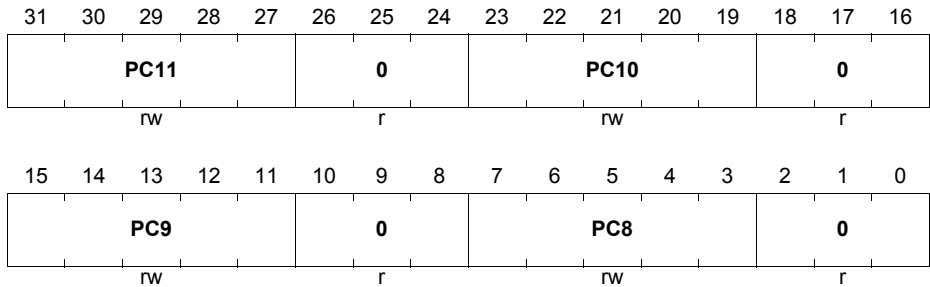
 Reset Value: 1010 1010_H


Field	Bits	Type	Description
PC9, PC10, PC11	[15:11], [23:19], [31:27]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
PC8	[7:3]	rw	Port Control for Port 15 Pin 8 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P15_IOC8
Port 15 Input/Output Control Register 8

 (18_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
PC9, PC10, PC11	[15:11], [23:19], [31:27]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC8	[7:3]	rw	Port Control for Port 15 Pin 8 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

14.10.3.8 Port 15 Input Register

The basic P15_IN register functionality is described on [Page 14-52](#). However, port lines P15.[15:9] are not connected. Therefore, bits P[15:9] in register P15_IN are always read as 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.10.3.9 Port 15 Pad Driver Mode 1 Register

P15_PDR1

 Port 15 Pad Driver Mode 1 Register (44_H)

 Reset Value: 0000 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL15	PD15		PL14	PD14		PL13	PD13		PL12	PD12					
r	r		r	r		r	r		r	r		r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL11	PD11		PL10	PD10		PL9	PD9		PL8	PD8					
rw	rw		rw	rw		rw	rw		rw	rw		rw			

Field	Bits	Type	Description
PD8	[2:0]	rw	Pad Driver Mode for Port 15 Pin 8
PL8	3	rw	Pad Level Selection for Port 15 Pin 8
PD9, PD10, PD11	[6:4], [10:8], [14:12]	rw	Reserved Read as 000 _B after reset; returns value that was written, have to be written with 0.
PD12, PD13, PD14, PD15	[18:16], [22:20], [26:24], [30:28]	r	Reserved Read as 000 _B after reset; should be written with 0.
PL9, PL10, PL11	7, 11, 15	rw	Reserved Read as 0 _B after reset; returns value that was written, have to be written with 0.
PL12, PL13, PL14, PL15	19, 23, 27, 31	r	Reserved Read as 0 _B after reset; should be written with 0.

14.10.3.10 Port 15 Emergency Stop Register

The basic P15_ESR register functionality is described on [Page 14-50](#). Port lines P15.[15:9] are not connected. Reading the P15_ESR bits EN[15:12] always returns 0 and should be written with 0. Reading the P15_ESR bits EN[11:9] returns value that was last written.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.11 Port 20

This section describes the Port 20 functionality.

14.11.1 Port 20 Configuration

Port 20 is a 13-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, MultiCAN+, SCU, OCDS, ASCLIN, HSCT, CCU6, and QSPI functions.

14.11.2 Port 20 Function Table

Table 14-24 summarizes the I/O control selection functions of each Port 20 line.

Table 14-24 Port 20 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P20.0	I	General-purpose input	P20_IN.P0	P20_IOCRO.PC0	0XXXX _B
		GTM input	TIN59		
		CAN node 3 input	RXDCAN3C		
		SCU input	REQ9		
		HSCT input	SYSCLK		
		OCDS input	$\overline{\text{TGI0}}$		
		GPT120 input	T6EUDA		
	O	General-purpose output	P20_OUT.P0		1X000 _B
		GTM output	TOUT59		1X001 _B
		ASCLIN3 output	ATX3		1X010 _B
		ASCLIN3 output	ASCLK3		1X011 _B
		Reserved	–		1X100 _B
		HSCT output	SYSCLK		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B
DIRx	OCDS; ENx	$\overline{\text{TGO0}}$	HW_OUT		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-24 Port 20 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P20.1	I	General-purpose input	P20_IN.P1	P20_IOCRO. PC1	0XXXX B
		GTM input	TIN60		
		OCDS input	$\overline{\text{TGI1}}$		
	O	General-purpose output	P20_OUT.P1		1X000 _B
		GTM output	TOUT60		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B
Reserved		–	1X111 _B		
DIRx	OCDS; ENx	$\overline{\text{TGO1}}$	HW_OUT		
P20.2	I	General-purpose input	P20_IN.P2	P20_IOCRO. PC2	0XXXX B
		OCDS input	$\overline{\text{TESTMODE}}$		
	O	Output function not available	–		1X000 _B
		Output function not available	–		1X001 _B
		Output function not available	–		1X010 _B
		Output function not available	–		1X011 _B
		Output function not available	–		1X100 _B
		Output function not available	–		1X101 _B
		Output function not available	–		1X110 _B
		Output function not available	–		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-24 Port 20 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.			
				Reg./Bit Field	Value		
P20.3	I	General-purpose input	P20_IN.P3	P20_IOCRO. PC3	0XXXX _B		
		GTM input	TIN61				
		ASCLIN3 input	ARX3C				
		GPT120 input	T6INA				
	O	General-purpose output	P20_OUT.P3		1X000 _B		
		GTM output	TOUT61		1X001 _B		
		ASCLIN3 output	ATX3		1X010 _B		
		QSPI0 output	SLSO09		1X011 _B		
		QSPI2 output	SLSO29		1X100 _B		
		CAN node 3 output	TXDCAN3		1X101 _B		
		Reserved	–		1X110 _B		
		Reserved	–		1X111 _B		
	P20.6	I	General-purpose input		P20_IN.P6	P20_IOCRR4. PC6	0XXXX _B
			GTM input		TIN62		
O		General-purpose output	P20_OUT.P6	1X000 _B			
		GTM output	TOUT62	1X001 _B			
		ASCLIN1 output	ARTS1	1X010 _B			
		QSPI0 output	SLSO08	1X011 _B			
		QSPI2 output	SLSO28	1X100 _B			
		Reserved	–	1X101 _B			
		Reserved	–	1X110 _B			
		Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-24 Port 20 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.			
				Reg./Bit Field	Value		
P20.7	I	General-purpose input	P20_IN.P7	P20_IOCR4. PC7	0XXXX _B		
		GTM input	TIN63				
		ASCLIN1 input	ACTS1A				
		CAN node 0 input	RXDCAN0B				
	O	General-purpose output	P20_OUT.P7		1X000 _B		
		GTM output	TOUT63		1X001 _B		
		Reserved	–		1X010 _B		
		Reserved	–		1X011 _B		
		Reserved	–		1X100 _B		
		Reserved	–		1X101 _B		
		SCU output	WDT1LCK		1X110 _B		
		CCU61 output	COOUT63		1X111 _B		
	P20.8	I	General-purpose input		P20_IN.P8	P20_IOCR8. PC8	0XXXX _B
			GTM input		TIN64		
O		General-purpose output	P20_OUT.P8	1X000 _B			
		GTM output	TOUT64	1X001 _B			
		ASCLIN1 output	ASLSO1	1X010 _B			
		QSPI0 output	SLSO00	1X011 _B			
		QSPI1 output	SLSO10	1X100 _B			
		CAN node 0 output	TXDCAN0	1X101 _B			
		SCU output	WDT0LCK	1X110 _B			
		CCU61 output	CC60	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-24 Port 20 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P20.9	I	General-purpose input	P20_IN.P9	P20_IOCR8. PC9	0XXXX _B
		GTM input	TIN65		
		ASCLIN1 input	ARX1C		
		CAN node 3 input	RXDCAN3E		
		SCU input	REQ11		
		QSPI0 input	SLSI0B		
	O	General-purpose output	P20_OUT.P9		1X000 _B
		GTM output	TOUT65		1X001 _B
		Reserved	–		1X010 _B
		QSPI0 output	SLSO01		1X011 _B
		QSPI1 output	SLSO11		1X100 _B
		Reserved	–		1X101 _B
		SCU output	WDTSLCK		1X110 _B
		CCU61 output	CC61		1X111 _B
P20.10	I	General-purpose input	P20_IN.P10	P20_IOCR8. PC10	0XXXX _B
		GTM input	TIN66		
	O	General-purpose output	P20_OUT.P10		1X000 _B
		GTM output	TOUT66		1X001 _B
		ASCLIN1 output	ATX1		1X010 _B
		QSPI0 output	SLSO06		1X011 _B
		QSPI2 output	SLSO27		1X100 _B
		CAN node 3 output	TXDCAN3		1X101 _B
		ASCLIN1 output	ASCLK1		1X110 _B
		CCU61 output	CC62		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-24 Port 20 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P20.11	I	General-purpose input	P20_IN.P11	P20_IOCR8. PC11	0XXXX _B
		GTM input	TIN67		
		QSPI0 input	SCLK0A		
	O	General-purpose output	P20_OUT.P11		1X000 _B
		GTM output	TOUT67		1X001 _B
		Reserved	–		1X010 _B
		QSPI0 output	SCLK0		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
CCU61 output	COOUT60	1X111 _B			
P20.12	I	General-purpose input	P20_IN.P12	P20_IOCR12 PC12	0XXXX _B
		GTM input	TIN68		
		QSPI0 input	MRST0A		
	O	General-purpose output	P20_OUT.P12		1X000 _B
		GTM output	TOUT68		1X001 _B
		Reserved	–		1X010 _B
		QSPI0 output	MRST0		1X011 _B
		QSPI0 output	MTSR0		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
CCU61 output	COOUT61	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-24 Port 20 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P20.13	I	General-purpose input	P20_IN.P13	P20_IOC12 · PC13	0XXXX _B
		GTM input	TIN69		
		QSPI0 input	SLSI0A		
	O	General-purpose output	P20_OUT.P13		1X000 _B
		GTM output	TOUT69		1X001 _B
		Reserved	–		1X010 _B
		QSPI0 output	SLSO02		1X011 _B
		QSPI1 output	SLSO12		1X100 _B
		QSPI0 output	SCLK0		1X101 _B
		Reserved	–		1X110 _B
CCU61 output	COU62	1X111 _B			
P20.14	I	General-purpose input	P20_IN.P14	P20_IOC12 · PC14	0XXXX _B
		GTM input	TIN70		
		QSPI0 input	MTSR0A		
	O	General-purpose output	P20_OUT.P14		1X000 _B
		GTM output	TOUT70		1X001 _B
		Reserved	–		1X010 _B
		QSPI0 output	MTSR0		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.11.3 Port 20 Registers

The following registers are available on Port 20:

Table 14-25 Port 20 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P20_OUT	Port 20 Output Register	0000 _H	Page 14-159 2)
P20_OMR	Port 20 Output Modification Register	0004 _H	Page 14-159 2)
P20_IOCR0	Port 20 Input/Output Control Register 0	0010 _H	Page 14-161 2)
P20_IOCR4	Port 20 Input/Output Control Register 4	0014 _H	Page 14-163 2)
P20_IOCR8	Port 20 Input/Output Control Register 8	0018 _H	Page 14-19
P20_IOCR12	Port 20 Input/Output Control Register 12	001C _H	Page 14-165 2)
P20_IN	Port 20 Input Register	0024 _H	Page 14-166 2)
P20_PDR0	Port 20 Pad Driver Mode 0 Register	0040 _H	Page 14-167
P20_PDR1	Port 20 Pad Driver Mode 1 Register	0044 _H	Page 14-168 2)
P20_ESR	Port 20 Emergency Stop Register	0050 _H	Page 14-169 2)
P20_OMSR0	Port 20 Output Modification Set Register 0	0070 _H	Page 14-159 2)
P20_OMSR4	Port 20 Output Modification Set Register 4	0074 _H	Page 14-160
P20_OMSR8	Port 20 Output Modification Set Register 8	0078 _H	Page 14-43
P20_OMSR12	Port 20 Output Modification Set Register 12	007C _H	Page 14-160 2)
P20_OMCR0	Port 20 Output Modification Clear Register 0	0080 _H	Page 14-160 2)
P20_OMCR4	Port 20 Output Modification Clear Register 4	0084 _H	Page 14-160
P20_OMCR8	Port 20 Output Modification Clear Register 8	0088 _H	Page 14-48
P20_OMCR12	Port 20 Output Modification Clear Register 12	008C _H	Page 14-160 2)

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-25 Port 20 Registers (cont'd)

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P20_OMSR	Port 20 Output Modification Set Register	0090 _H	Page 14-159 ²⁾
P20_OMCR	Port 20 Output Modification Clear Register	0094 _H	Page 14-160 ²⁾
P20_ACCEN1	Port 20 Access Enable Register 1	00F8 _H	Page 14-54
P20_ACCEN0	Port 20 Access Enable Register 0	00FC _H	Page 14-53

1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))

2) These registers are listed and noted here in the Port 20 section because they differ from the general port register description given in [Section 14.3](#).

14.11.3.1 Port 20 Output Register

The basic P20_OUT register functionality is described on [Page 14-36](#). Port line P20.[5:4] and P20.15 are not connected. Reading the P20_OUT bits P[5:4] and P15 returns the value that was last written (0 after reset). These connected bits can be also set/reset by the corresponding bits in P20_OMR, P20_OMSR, P20_OMCR, P20_OMSRx (x=4,12), P20_OMCRx (x=4,12). P20_OUT.P2 does not determine the level at P20.2, returns the value that was last written.

14.11.3.2 Port 20 Output Modification Register

The basic P20_OMR register functionality is described on [Page 14-38](#). However, port lines P20.[5:4] and P20.15 are not connected. The P20_OMR bits PS[5:4], PS15 and PCL[5:4], PCL15 have no direct effect on port lines but only on register bits P20_OUT P[5:4] and P15. P20_OMR bit PS2 and PCL2 has no effect on P20_OUT.P2.

14.11.3.3 Port 20 Output Modification Set Register

The basic P20_OMSR register functionality is described on [Page 14-40](#). However, port lines P20.15 is not connected. The P20_OMSR bits PS15 has no direct effect on port line but only on register bits P20_OUT.P15. P20_OMSR bit PS2 has no effect on P20_OUT.P2.

14.11.3.4 Port 20 Output Modification Set Register 0

The basic P20_OMSR0 register functionality is described on [Page 14-41](#). P20_OMSR0 bit PS2 has no effect on P20_OUT.P2.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.11.3.5 Port 20 Output Modification Set Register 4

The basic P20_OMSR4 register functionality is described on [Page 14-42](#). However, port lines P20.[5:4] are not connected. The P20_OMSR4 bits PS[5:4] has no direct effect on port lines but only on register bits P20_OUT.P[5:4].

14.11.3.6 Port 20 Output Modification Set Register 12

The basic P20_OMSR12 register functionality is described on [Page 14-44](#). However, port lines P20.15 is not connected. The P20_OMSR12 bits PS15 has no direct effect on port lines but only on register bits P20_OUT.P15.

14.11.3.7 Port 20 Output Modification Clear Register

The basic P20_OMCR register functionality is described on [Page 14-45](#). However, port lines P20.[5:4] and P20.15 are not connected. The P20_OMCR bits PCL[5:4], PCL15 has no direct effect on port lines but only on register bits P20_OUT.P[5:4] and P15. P20_OMCR bit PCL2 has no effect on P20_OUT.P2.

14.11.3.8 Port 20 Output Modification Clear Register 0

The basic P20_OMCR0 register functionality is described on [Page 14-46](#). P20_OMCR0 bit PCL2 has no effect on P20_OUT.P2.

14.11.3.9 Port 20 Output Modification Clear Register 4

The basic P20_OMCR4 register functionality is described on [Page 14-47](#). However, port lines P20.[5:4] are not connected. The P20_OMCR4 bits PCL[5:4] has no direct effect on port lines but only on register bits P20_OUT.P[5:4].

14.11.3.10Port 20 Output Modification Clear Register 12

The basic P20_OMCR12 register functionality is described on [Page 14-49](#). However, port lines P20.15 are not connected. The P20_OMCR12 bits PCL15 has no direct effect on port lines but only on register bits P20_OUT.P15.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.11.3.11 Port 20 Input/Output Control Register 0

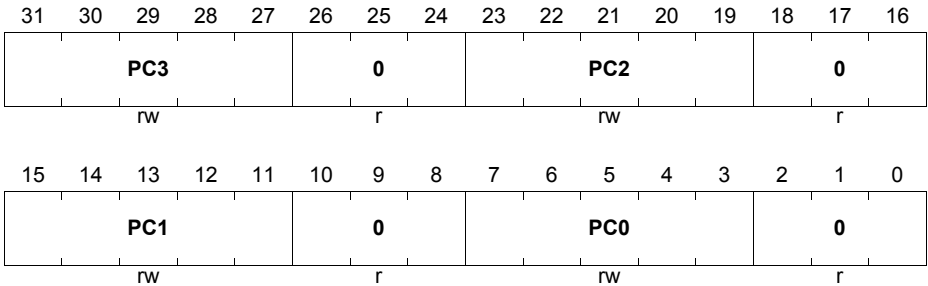
The PC2 bit field in register P20_IOCRO is not connected.

P20_IOCRO

Port 20 Input/Output Control Register 0

(10_H)

Reset Value: 1010 1010_H



Field	Bits	Type	Description
PC0	[7:3]	rw	Port Control for Port n Pin 0
PC1	[15:11]	rw	Port Control for Port n Pin 1
PC3	[31:27]	rw	Port Control for Port n Pin 3
PC2	[23:19]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P20_IOCR0
Port 20 Input/Output Control Register 0

 (10_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC3				0				PC2				0			
rw				r				rw				r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC1				0				PC0				0			
rw				r				rw				r			

Field	Bits	Type	Description
PC0	[7:3]	rw	Port Control for Port n Pin 0
PC1	[15:11]	rw	Port Control for Port n Pin 1
PC3	[31:27]	rw	Port Control for Port n Pin 3
PC2	[23:19]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.11.3.12 Port 20 Input/Output Control Register 4

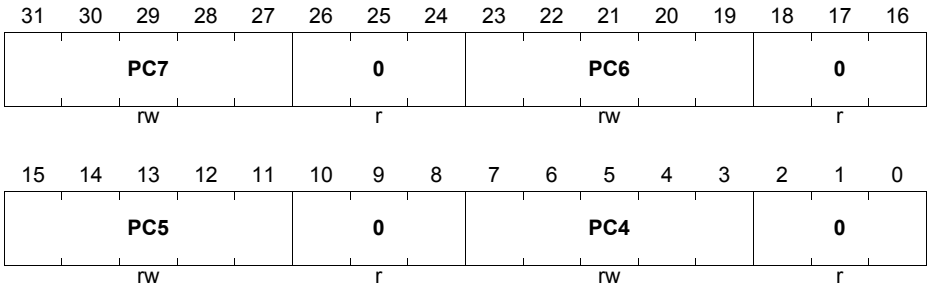
The PC[5:4] bit fields in register P20_IOC4 are not connected.

P20_IOC4

Port 20 Input/Output Control Register 4

(14_H)

Reset Value: 1010 1010_H

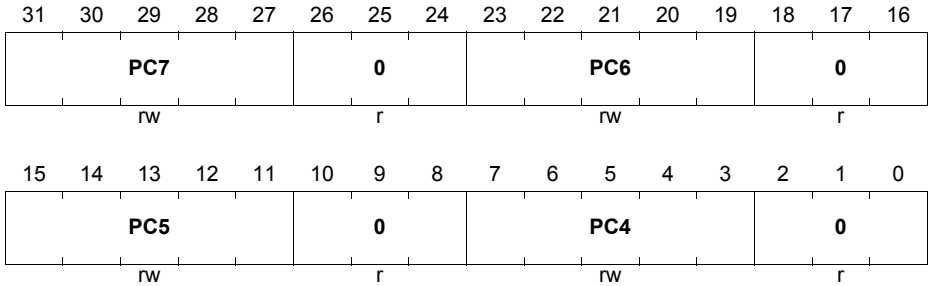


Field	Bits	Type	Description
PC4, PC5	[7:3], [15:11]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
PC6	[23:19]	rw	Port Control for Port 20 Pin 6 (coding see Table 14-5)
PC7	[31:27]	rw	Port Control for Port 20 Pin 7 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P20_IOC4
Port 20 Input/Output Control Register 4

 (14_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
PC4, PC5	[7:3], [15:11]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC6	[23:19]	rw	Port Control for Port 20 Pin 6 (coding see Table 14-5)
PC7	[31:27]	rw	Port Control for Port 20 Pin 7 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.11.3.13 Port 20 Input/Output Control Register 12

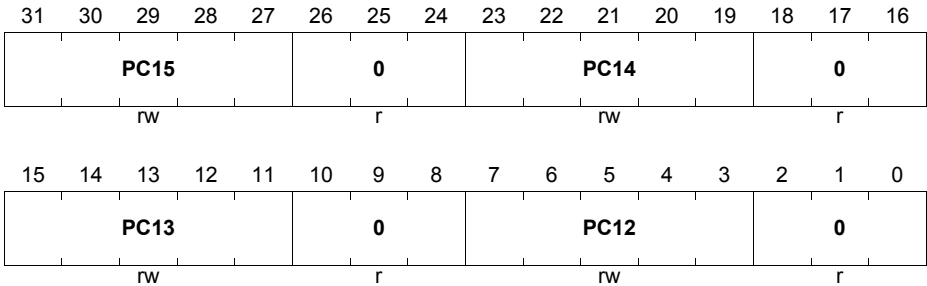
The PC15 bit fields in register P20_IOCR12 is not connected.

P20_IOCR12

Port 20 Input/Output Control Register 12

(1C_H)

Reset Value: 1010 1010_H



Field	Bits	Type	Description
PC15	[31:27]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
PC12	[7:3]	rw	Port Control for Port 20 Pin 12 (coding see Table 14-5)
PC13	[15:11]	rw	Port Control for Port 20 Pin 13 (coding see Table 14-5)
PC14	[23:19]	rw	Port Control for Port 20 Pin 14 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P20_IOC12
Port 20 Input/Output Control Register 12

 (1C_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC15				0				PC14				0			
rw				r				rw				r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC13				0				PC12				0			
rw				r				rw				r			

Field	Bits	Type	Description
PC15	[31:27]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC12	[7:3]	rw	Port Control for Port 20 Pin 12 (coding see Table 14-5)
PC13	[15:11]	rw	Port Control for Port 20 Pin 13 (coding see Table 14-5)
PC14	[23:19]	rw	Port Control for Port 20 Pin 14 (coding see Table 14-5)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

14.11.3.14Port 20 Input Register

The basic P20_IN register functionality is described on [Page 14-52](#). However, port lines P20.[5:4] and P20.15 are not connected. Therefore, bits P[5:4] and P15 in register P20_IN are always read as 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

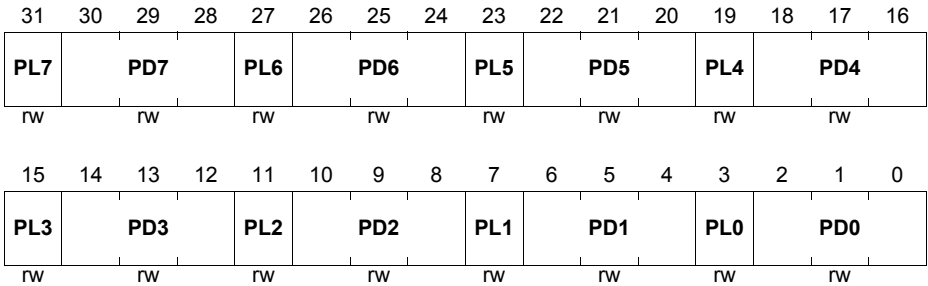
14.11.3.15 Port 20 Pad Driver Mode 0 Register

The basic P20_PDR0 register functionality is described on [Page 14-25](#).

P20_PDR0

Port 20 Pad Driver Mode 0 Register (40_H)

Reset Value: 3333 3333_H



Field	Bits	Type	Description
PD0	[2:0]	rw	Pad Driver Mode for Port 20 Pin 0
PD1	[6:4]	rw	Pad Driver Mode for Port 20 Pin 1
PD3	[14:12]	rw	Pad Driver Mode for Port 20 Pin 3
PD6	[26:24]	rw	Pad Driver Mode for Port 20 Pin 6
PD7	[30:28]	rw	Pad Driver Mode for Port 20 Pin 7
PL0	3	rw	Pad Level Selection for Port 20 Pin 0
PL1	7	rw	Pad Level Selection for Port 20 Pin 1
PL3	15	rw	Pad Level Selection for Port 20 Pin 3
PL6	27	rw	Pad Level Selection for Port 20 Pin 6
PL7	31	rw	Pad Level Selection for Port 20 Pin 7
PD2, PD4, PD5	[10:8], [18:16], [22:20]	rw	Reserved Read as 000 _B after reset; returns value that was written.
PL2, PL4, PL5	11, 19, 23	rw	Reserved Read as 0 _B after reset; returns value that was written.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.11.3.16 Port 20 Pad Driver Mode 1 Register

P20_PDR1
Port 20 Pad Driver Mode 1 Register (44_H)
Reset Value: 3333 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL15	PD15		PL14	PD14		PL13	PD13		PL12	PD12					
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL11	PD11		PL10	PD10		PL9	PD9		PL8	PD8					
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
PD8	[2:0]	rw	Pad Driver Mode for Port 20 Pin 8
PD9	[6:4]	rw	Pad Driver Mode for Port 20 Pin 9
PD10	[10:8]	rw	Pad Driver Mode for Port 20 Pin 10
PD11	[14:12]	rw	Pad Driver Mode for Port 20 Pin 11
PD12	[18:16]	rw	Pad Driver Mode for Port 20 Pin 12
PD13	[22:20]	rw	Pad Driver Mode for Port 20 Pin 13
PD14	[26:24]	rw	Pad Driver Mode for Port 20 Pin 14
PL8	3	rw	Pad Level Selection for Port 20 Pin 8
PL9	7	rw	Pad Level Selection for Port 20 Pin 9
PL10	11	rw	Pad Level Selection for Port 20 Pin 10
PL11	15	rw	Pad Level Selection for Port 20 Pin 11
PL12	19	rw	Pad Level Selection for Port 20 Pin 12
PL13	23	rw	Pad Level Selection for Port 20 Pin 13
PL14	27	rw	Pad Level Selection for Port 20 Pin 14
PD15	[30:28]	rw	Reserved Read as 000 _B after reset; returns value that was written.
PL15	31	rw	Reserved Read as 0 _B after reset; returns value that was written.

14.11.3.17 Port 20 Emergency Stop Register

The basic P20_ESR register functionality is described on [Page 14-50](#). Port lines P20.2, P20.[5:4] and P20.15 are not connected. Reading the EN2, EN[5:4], EN15 returns value that was last written.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.12 Port 21

This section describes the Port 21 functionality.

Port 21 is an 8-bit GPIO port. Pins associated to it be used in two ways:

- as a CMOS Port where each pin outputs one signal, as any other port (only exception - no open drain mode available), and
- as an output LVDS port

The switching between the CMOS/LVDS modes is done via the P21_LPCR_x register.

The switching between Input/Output and Pull-Up/Pull-Down control is done via the IOCR register.

Attention: *In the LVDS mode the IOCR.PC_x bit field of each pin of the LVDS pair must be programmed as output, that is 1xxxx_B.*

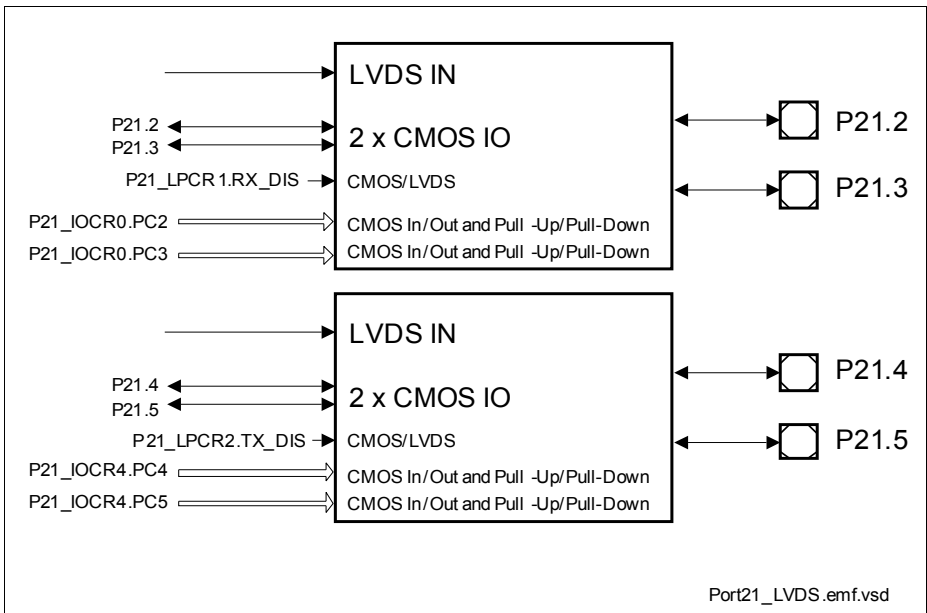


Figure 14-4 Port 21 Pad Connections

14.12.1 Port 21 Configuration

Port 21 is a 8-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, Ethernet, ASCLIN, QSPI, SCU, CCU6, HSCT, OCDS and GPT12 functions.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.12.2 Port 21 Function Table

Table 14-26 summarizes the I/O control selection functions of each Port 21 line.

Table 14-26 Port 21 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P21.0	I	General-purpose input	P21_IN.P0	P21_IOCRO. PC0	0XXXX _B
		GTM input	TIN51		
	O	General-purpose output	P21_OUT.P0		1X000 _B
		GTM output	TOUT51		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		ETH output	ETHMDC		1X110 _B
Reserved	–	1X111 _B			
P21.1	I	General-purpose input	P21_IN.P1	P21_IOCRO. PC1	0XXXX _B
		GTM input	TIN52		
		ETH input	ETHMDIOB		
	O	General-purpose output	P21_OUT.P1		1X000 _B
		GTM output	TOUT52		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
ETH output	ETHMDIO ¹⁾	1X110 _B			
Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-26 Port 21 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P21.2	I	General-purpose input	P21_IN.P2	P21_IOCRO. PC2	0XXXX _B
		GTM input	TIN53		
		QSPI2 input	MRST2CN		
		QSPI3 input	MRST3FN		
		SCU input	EMGSTOPB		
		HSCT input	RXDN		
	O	General-purpose output	P21_OUT.P2		1X000 _B
		GTM output	TOUT53		1X001 _B
		ASCLIN3 output	ASLSO3		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		ETH output	ETHMDC		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B
P21.3	I	General-purpose input	P21_IN.P3	P21_IOCRO. PC3	0XXXX _B
		GTM input	TIN54		
		QSPI2 input	MRST2CP		
		QSPI3 input	MRST3FP		
		HSCT input	RXDP		
		ETH input	ETHMDIOD		
	O	General-purpose output	P21_OUT.P3		1X000 _B
		GTM output	TOUT54		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
Reserved	–		1X111 _B		
DIRx	ETH output	ETHMDIO	HWOUT		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-26 Port 21 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P21.4	I	General-purpose input	P21_IN.P4	P21_IOC4. PC4	0XXXX _B
		GTM input	TIN55		
	O	General-purpose output	P21_OUT.P4		1X000 _B
		GTM output	TOUT55		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B
DIRx	HSCT output	TXDN	HWOUT		
P21.5	I	General-purpose input	P21_IN.P5	P21_IOC4. PC5	0XXXX _B
		GTM input	TIN56		
	O	General-purpose output	P21_OUT.P5		1X000 _B
		GTM output	TOUT56		1X001 _B
		ASCLIN3 output	ASCLK3		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B
DIRx	HSCT output	TXDP	HWOUT		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-26 Port 21 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P21.6	I	General-purpose input	P21_IN.P6	P21_IOCR4. PC6	0XXXX _B
		GTM input	TIN57		
		OCDS input	$\overline{\text{TGI2}}$		
		OCDS input	TDI		
		GPT120 input	T5EUDA		
		ASCLIN3 input	ARX3F		
	O	General-purpose output	P21_OUT.P6		1X000 _B
		GTM output	TOUT57		1X001 _B
		ASCLIN3 output	ASLSO3		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		HSCT output	SYSCLK		1X101 _B
		Reserved	–		1X110 _B
		GPT120	T3OUT		1X111 _B
DIRx	OCDS; ENx	$\overline{\text{TGO2}}$	HW_OUT		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-26 Port 21 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P21.7	I	General-purpose input	P21_IN.P7	P21_IOC4. PC7	0XXXX _B
		GTM input	TIN58		
		OCDS input	DAP2		
		OCDS input	$\overline{\text{TGI3}}$		
		OCDS input	TDO		
		GPT120 input	T5INA		
		Ethernet input	ETHRXERB		
	O	General-purpose output	P21_OUT.P7		1X000 _B
		GTM output	TOUT58		1X001 _B
		ASCLIN3 output	ATX3		1X010 _B
		ASCLIN3 output	ASCLK3		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		GPT120 output	T6OUT		1X111 _B
DIRx	OCDS; ENx	$\overline{\text{TGO3}}$	HW_OUT		
	OCDS; ENx	TDO			

1) ETHMDIO on P21.1 cannot be used as a bi-directional signal and in productive devices.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.12.3 Port 21 Registers

The following registers are available on Port 21:

Table 14-27 Port 21 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P21_OUT	Port 21 Output Register	0000 _H	Page 14-177 ²⁾
P21_OMR	Port 21 Output Modification Register	0004 _H	Page 14-177 ²⁾
P21_IOCRO	Port 21 Input/Output Control Register 0	0010 _H	Page 14-14
P21_IOCRR4	Port 21 Input/Output Control Register 4	0014 _H	Page 14-17
P21_IN	Port 21 Input Register	0024 _H	Page 14-177 ²⁾
P21_PDR0	Port 21 Pad Driver Mode 0 Register	0040 _H	Page 14-25
P21_ESR	Port 21 Emergency Stop Register	0050 _H	Page 14-181 ²⁾
P21_OMSR0	Port 21 Output Modification Set Register 0	0070 _H	Page 14-41
P21_OMSR4	Port 21 Output Modification Set Register 4	0074 _H	Page 14-42
P21_OMCR0	Port 21 Output Modification Clear Register 0	0080 _H	Page 14-46
P21_OMCR4	Port 21 Output Modification Clear Register 4	0084 _H	Page 14-47
P21_OMSR	Port 21 Output Modification Set Register	0090 _H	Page 14-177 ²⁾
P21_OMCR	Port 21 Output Modification Clear Register	0094 _H	Page 14-177 ²⁾
P21_LPCR1	Port 21 LVDS Pad Control Register 1	00A4 _H	Page 14-178 ²⁾
P21_LPCR2	Port 21 LVDS Pad Control Register 2	00A8 _H	Page 14-178 ²⁾
P21_ACCEN1	Port 21 Access Enable Register 1	00F8 _H	Page 14-54
P21_ACCEN0	Port 21 Access Enable Register 0	00FC _H	Page 14-53

1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))

2) These registers are listed and noted here in the Port 21 section because they differ from the general port register description given in [Section 14.3](#).

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.12.3.1 Port 21 Output Register

The basic P21_OUT register functionality is described on [Page 14-36](#). Port line P21.[15:8] are not connected. Reading the P21_OUT bit P[15:8] always return 0. These connected bits can be also set/reset by the corresponding bits in P21_OMR, P21_OMSR, P21_OMCR, P21_OMSRx (x=0,4), P21_OMCRx (x=0,4).

14.12.3.2 Port 21 Output Modification Register

The basic P21_OMR register functionality is described on [Page 14-38](#). However, port lines P21.[15:8] are not connected.

14.12.3.3 Port 21 Output Modification Set Register

The basic P21_OMSR register functionality is described on [Page 14-40](#). However, port lines P21.[15:8] is not connected.

14.12.3.4 Port 21 Output Modification Clear Register

The basic P21_OMCR register functionality is described on [Page 14-45](#). However, port lines P21.[15:8] are not connected.

14.12.3.5 Port 21 Input Register

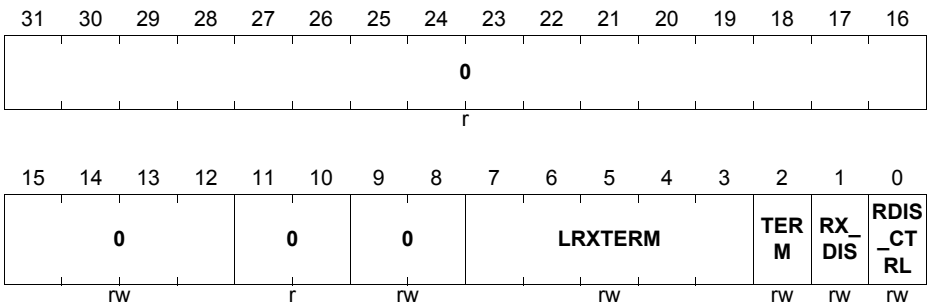
The basic P21_IN register functionality is described on [Page 14-52](#). However, port lines P21.[15:8] are not connected. Therefore, bits P[15:8] in register P21_IN are always read as 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.12.3.6 P21 LVDS Pad Control Register

For configuration of LVDSH mode in LPCR_x registers, the LVDS function need to be selected (CMOS function disabled).

If port-controlled, the receive or transmit enabling of the LVDS path is controlled by the respective RX_{_}DIS and TX_{_}DIS bits in the P21_LPCR_x registers. Otherwise, this is handled by the equivalent control bit via the HSCT module. If LVDS mode is disabled, the respective functions of the control bits of the register are not activated, except TX_{_}PD bit which needs to be additionally configured, if desired.

P21_LPCR1 register controls the functions of the LVDSH RX pads for P21.[3:2].

P21_LPCR1
Port 21 LVDS Pad Control Register 1(00A4_H)
Reset Value: 0000 0046_H


Field	Bits	Type	Description
RDIS_CTRL	0	rw	LVDS RX__DIS controller The LVDS RX __ DIS control function can be selected from the Port (default) or HSCT module. 0 _B Port controlled 1 _B HSCT controlled
RX__DIS	1	rw	Disable Receive LVDS Disable the receive LVDS / enable CMOS path. If this bit is set to 1 - no transfer from the LVDS sender can be received and the receiver LVDS is in low power state. 0 _B enable LVDS / disable CMOS mode 1 _B disable LVDS / enable CMOS mode

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
TERM	2	rw	Select Receiver Termination Mode Selects a suitable internal load capacitance. 0 _B external termination - on the PCB 1 _B 100 Ω Receiver internal termination
LRXTERM	[7:3]	rw	LVDS RX Poly-resistor configuration value Programming bits for the on die poly resistor termination. The value is configured during production test. Each chip configuration on this bit field is unique and configured during production testing. <i>Note: The configuration value shall not be changed by user after start-up for a guaranteed behavior.</i>
0	[15:12], [9:8]	rw	Reserved Read as 0; should be written with 0.
0	[31:16], [11:10]	r	Reserved Read as 0; should be written with 0.

P21_LPCR2 register controls the functions of the LVDS TX pad for P21.[5:4].

P21_LPCR2
Port 21 LVDS Pad Control Register 2(00A8_H)
Reset Value: 0000 6000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX_PWD PD	TX_PD	TX_DIS	TDIS _CT RL	0		LVD SRL	LVD SR					0			
rw	rw	rw	rw	r		rw	rw					rw			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
LVDSR	8	rw	Special reduced LVDS electrical signaling mode LVDSR = 1 and LVDSRL = 0 activates LVDSR electrical. LVDSR = 1 and LVDSRL = 1 Not allowed - Software must prevent this situation! <i>Note: LVDSR = 1 and LVDSRL = 1 the special reduced electrical LVDS signaling is enabled.</i>
LVDSRL	9	rw	LVDS IEEE electrical signaling mode LVDSR = 0 and LVDSRL = 1 activates the IEEE reduced LVDS link electrical signaling specification LVDSR = 0 and LVDSRL = 0 activates the IEEE general purpose link.
TDIS_CTRL	12	rw	LVDS TX_DIS controller The LVDS TX_DIS control function can be selected from the Port (default) or HSCT module. 0 _B Port controlled 1 _B HSCT controlled
TX_DIS	13	rw	Disable Transmit LVDS Disable the transmit LVDS / enable CMOS path. If this bit is set to 1 - no transfer on LVDS data path can be initiated and the LVDS driver is disabled. 0 _B enable LVDS / disable CMOS mode 1 _B disable LVDS / enable CMOS mode
TX_PD	14	rw	LVDS Power Down 0 _B LVDS power on 1 _B LVDS power down (default)
TX_PWDPD	15	rw	Disable TX Power down pull down. This function disables or enables the LVDS pull down resistor. The application code must disable TX power down pull down resistor with a power up. With a LVDS Power Down configuration the pull down function must be enabled, if required. 0 _B disabled TX Power down pull down resistor. 1 _B enabled TX Power down pull down resistor.
0	[7:0]	rw	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
0	[31:16], [11:10]	r	Reserved Read as 0; should be written with 0.

14.12.3.7 Port 21 Emergency Stop Register

The basic P21_ESR register functionality is described on [Page 14-50](#). Port lines P21.[15:8] are not connected. Reading the P15_ESR bits EN[15:8] always returns 0. EN2 is reserved and has no influence on the emergency control function of P21.2, should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.13 Port 22

This section describes the Port 22 functionality.

Port 22 is an 4-bit GPIO port. Pins associated to it be used in two ways:

- as a CMOS Port where each pin outputs one signal, as any other port (only exception - no open drain mode available), and
- as an output LVDS port where a pin pair (two pins) outputs one differential MSC signal.

The switching between the CMOS/LVDS modes is done via the P22_PDR0 registers.

The switching between Input/Ouput and Pull-Up/Pull-Down control is done via the IOCR register.

Attention: In the LVDS mode the IOCR.PCx bit field of each pin of the LVDS pair must be programed as output, that is 1xxxx_b.

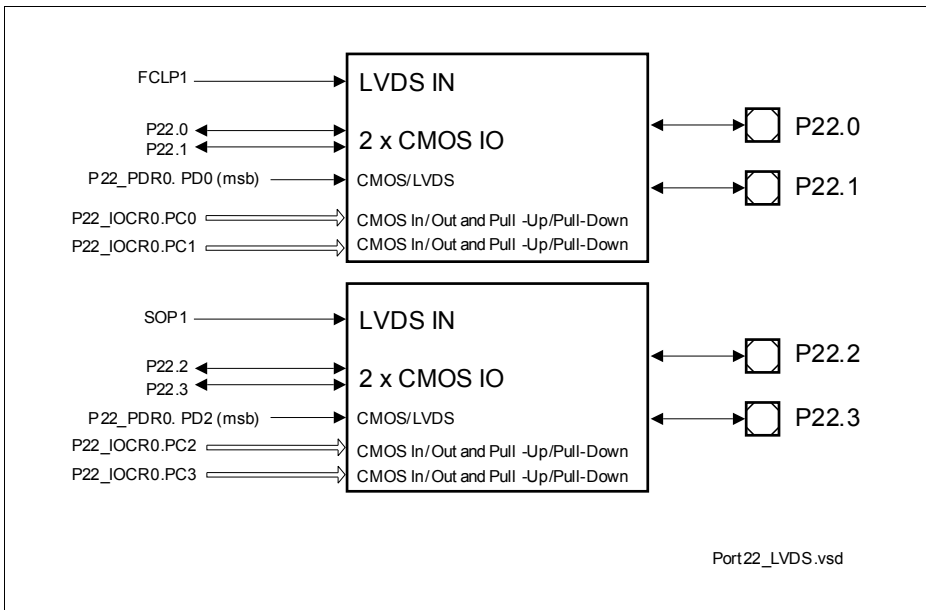


Figure 14-5 Port 22 Pad Connections

14.13.1 Port 22 Configuration

Port 22 is a 4-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, QSPI, CCU6 and MSC functions.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.13.2 Port 22 Function Table

Table 14-28 summarizes the I/O control selection functions of each Port 22 line.

Table 14-28 Port 22 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P22.0	I	General-purpose input	P22_IN.P0	P22_IOCRO. PC0	0XXXX _B
		GTM input	TIN47		
		QSPI3 input	MTSR3E		
	O	General-purpose output	P22_OUT.P0		1X000 _B
		GTM output	TOUT47		1X001 _B
		Reserved	–		1X010 _B
		QSPI3 output	MTSR3		1X011 _B
		QSPI3 output	SCLK3N		1X100 _B
		MSC1 output	FCLN1		1X101 _B
		MSC1 output	FCLND1		1X110 _B
Reserved	–	1X111 _B			
P22.1	I	General-purpose input	P22_IN.P1	P22_IOCRO. PC1	0XXXX _B
		GTM input	TIN48		
		QSPI3 input	MRST3E		
	O	General-purpose output	P22_OUT.P1		1X000 _B
		GTM output	TOUT48		1X001 _B
		Reserved	–		1X010 _B
		QSPI3 output	MRST3		1X011 _B
		QSPI3 output	SCLK3P		1X100 _B
		MSC1 output	FCLP1		1X101 _B
		Reserved	–		1X110 _B
Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-28 Port 22 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.			
				Reg./Bit Field	Value		
P22.2	I	General-purpose input	P22_IN.P2	P22_IOCRO. PC2	0XXXX _B		
		GTM input	TIN49				
		QSPI3 input	SLSI3D				
	O	General-purpose output	P22_OUT.P2		1X000 _B		
		GTM output	TOUT49		1X001 _B		
		Reserved	–		1X010 _B		
		QSPI3 output	SLSO312		1X011 _B		
		QSPI3 output	MTSR3N		1X100 _B		
		MSC1 output	SON1		1X101 _B		
		MSC1 output	SOND1		1X110 _B		
		Reserved	–		1X111 _B		
	P22.3	I	General-purpose input		P22_IN.P3	P22_IOCRO. PC3	0XXXX _B
			GTM input		TIN50		
			QSPI3 input		SCLK3E		
O		General-purpose output	P22_OUT.P3	1X000 _B			
		GTM output	TOUT50	1X001 _B			
		Reserved	–	1X010 _B			
		QSPI3 output	SCLK3	1X011 _B			
		QSPI3 output	MTSR3P	1X100 _B			
		MSC1 output	SOP1	1X101 _B			
		Reserved	–	1X110 _B			
		Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.13.3 Port 22 Registers

The following registers are available on Port 22:

Table 14-29 Port 22 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P22_OUT	Port 22 Output Register	0000 _H	Page 14-186 ²⁾
P22_OMR	Port 22 Output Modification Register	0004 _H	Page 14-186 ²⁾
P22_IOCRO	Port 22 Input/Output Control Register 0	0010 _H	Page 14-14
P22_IN	Port 22 Input Register	0024 _H	Page 14-186 ²⁾
P22_PDR0	Port 22 Pad Driver Mode 0 Register	0040 _H	Page 14-187 ²⁾
P22_ESR	Port 22 Emergency Stop Register	0050 _H	Page 14-190 ²⁾
P22_OMSR0	Port 22 Output Modification Set Register 0	0070 _H	Page 14-41
P22_OMCR0	Port 22 Output Modification Clear Register 0	0080 _H	Page 14-46
P22_OMSR	Port 22 Output Modification Set Register	0090 _H	Page 14-186 ²⁾
P22_OMCR	Port 22 Output Modification Clear Register	0094 _H	Page 14-186 ²⁾
P22_LPCR0	Port 22 LVDS Pad Control Register 0	00A0 _H	Page 14-189 ²⁾
P22_LPCR1	Port 22 LVDS Pad Control Register 1	00A4 _H	Page 14-189 ²⁾
P22_ACCEN1	Port 22 Access Enable Register 1	00F8 _H	Page 14-54
P22_ACCEN0	Port 22 Access Enable Register 0	00FC _H	Page 14-53

1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))

2) These registers are listed and noted here in the Port 22 section because they differ from the general port register description given in [Section 14.3](#).

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.13.3.1 Port 22 Output Register

The basic P22_OUT register functionality is described on [Page 14-36](#). Port line P22.[15:4] are not connected. Reading P22_OUT bit P[15:4] always return 0. These connected bits can be also set/reset by the corresponding bits in P22_OMR, P22_OMSR, P22_OMCR, P22_OMSR0, P22_OMCR0.

14.13.3.2 Port 22 Output Modification Register

The basic P22_OMR register functionality is described on [Page 14-38](#). However, port lines P22.[15:4] are not connected.

14.13.3.3 Port 22 Output Modification Set Register

The basic P22_OMSR register functionality is described on [Page 14-40](#). However, port lines P22.[15:4] are not connected.

14.13.3.4 Port 22 Output Modification Clear Register

The basic P22_OMCR register functionality is described on [Page 14-45](#). However, port lines P22.[15:12] are not connected.

14.13.3.5 Port 22 Input Register

The basic P22_IN register functionality is described on [Page 14-52](#). However, port lines P22.[15:12] are not connected. Therefore, bits P[15:12] in register P22_IN are always read as 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.13.3.6 Port 22 Pad Driver Mode 0 Register

The basic P22_PDR0 register functionality is described on [Page 14-25](#).

P22_PDR0
Port 22 Pad Driver Mode 0 Register (40_H)
Reset Value: 3333 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL7	PD7		PL6	PD6		PL5	PD5		PL4	PD4					
rw	rw		rw	rw		rw	rw		rw	rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL3	PD3		PL2	PD2		PL1	PD1		PL0	PD0					
rw	rw		rw	rw		rw	rw		rw	rw					

Field	Bits	Type	Description
PD0	[2:0]	rw	Pad Driver Mode for Port 22 Pin [1:0] The MSB of PDx switches between CMOS (default) and LVDS modes. 0XX _B CMOS input or output (depending on IOCRx setting). 1XX _B LVDS output (pull-up or pull-down must be switched off via IOCRx)
PD2	[10:8]	rw	Pad Driver Mode for Port 22 Pin [3:2] The MSB of PDx switches between CMOS (default) and LVDS modes. 0XX _B CMOS input or output (depending on IOCRx setting). 1XX _B LVDS output (pull-up or pull-down must be switched off via IOCRx)
PD1	[6:4]	rw	Pad Driver Mode for Port 22 Pin 1
PD3	[14:12]	rw	Pad Driver Mode for Port 22 Pin 3
PD4	[18:16]	rw	Pad Driver Mode for Port 22 Pin 4
PD5	[22:20]	rw	Pad Driver Mode for Port 22 Pin 5
PL0	3	rw	Pad Level Selection for Port 22 Pin 0
PL1	7	rw	Pad Level Selection for Port 22 Pin 1
PL2	11	rw	Pad Level Selection for Port 22 Pin 2
PL3	15	rw	Pad Level Selection for Port 22 Pin 3

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PL4	19	rw	Pad Level Selection for Port 22 Pin 4
PL5	23	rw	Pad Level Selection for Port 22 Pin 5
PD6, PD7	[26:24], [30:28]	rw	Reserved Read as 000 _B after reset; returns value that was written.
PL6, PL7	27, 31	rw	Reserved Read as 0 _B after reset; returns value that was written.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

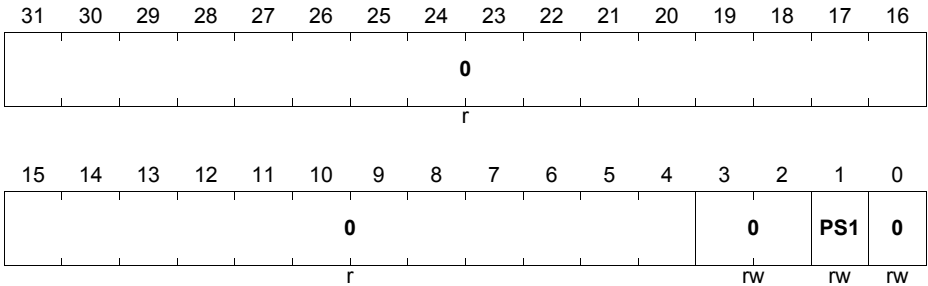
14.13.3.7 Port 22 LVDS Pad Control Register

For P22_LPCR_x($x=0-1$), PS1 needs to be configured to the intended pad supply level before the LVDSM function is enabled.

P22_LPCR0

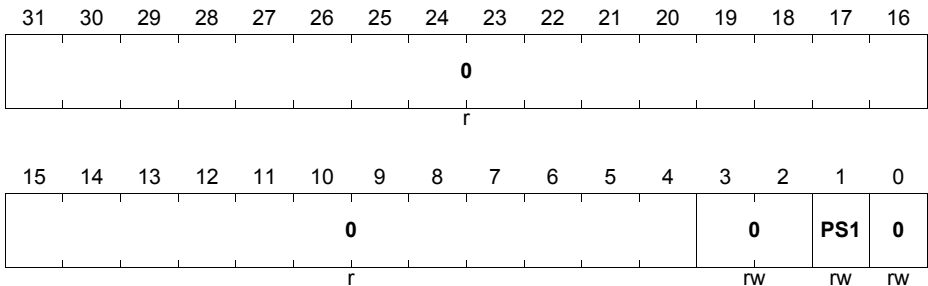
Port 22 LVDS Pad Control Register 0 (A0_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
PS1	1	rw	Pad Supply for pins [1:0] Selects between 5V or 3.3V supply on V_{EXT} for the LVDSM pad-pair. 0 _B 5V supply 1 _B 3.3V supply
0	[3:2], 0	rw	Reserved Read as 0; must be written with 0.
0	[31:4]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P22_LPCR1
Port 22 LVDS Pad Control Register 1 (A4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
PS1	1	rw	Pad Supply for pins [3:2] Selects between 5V or 3.3V supply on V_{EXT} for the LVDSM pad-pair. 0 _B 5V supply 1 _B 3.3V supply
0	[3:2], 0	rw	Reserved Read as 0; must be written with 0.
0	[31:4]	r	Reserved Read as 0; should be written with 0.

14.13.3.8 Port 22 Emergency Stop Register

The basic P22_ESR register functionality is described on [Page 14-50](#). Port lines P22.[15:4] are not connected. Reading EN[15:4] always returns 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.14 Port 23

This section describes the Port 23 functionality.

14.14.1 Port 23 Configuration

Port 23 is a 6-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, MSC, QSPI, ASCLIN, SCU and CCU6 functions.

14.14.2 Port 23 Function Table

Table 14-30 summarizes the I/O control selection functions of each Port 23 line.

Table 14-30 Port 23 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P23.0	I	General-purpose input	P23_IN.P0	P23_IOCRO. PC0	0XXXX _B
		GTM input	TIN41		
	O	General-purpose output	P23_OUT.P0		1X000 _B
		GTM output	TOUT41		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-30 Port 23 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P23.1	I	General-purpose input	P23_IN.P1	P23_IOCRO. PC1	0XXXX _B
		GTM input	TIN42		
		MSC1 input	SDI10		
	O	General-purpose output	P23_OUT.P1		1X000 _B
		GTM output	TOUT42		1X001 _B
		ASCLIN1 output	ARTS1		1X010 _B
		QSPI3 output	SLSO313		1X011 _B
		GTM output	GTMCLK0		1X100 _B
		Reserved	–		1X101 _B
		SCU output	EXTCLK0		1X110 _B
Reserved	–	1X111 _B			
P23.2	I	General-purpose input	P23_IN.P2	P23_IOCRO. PC2	0XXXX _B
		GTM input	TIN43		
	O	General-purpose output	P23_OUT.P2		1X000 _B
		GTM output	TOUT43		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-30 Port 23 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P23.3	I	General-purpose input	P23_IN.P3	P23_IOCRO. PC3	0XXXX _B
		GTM input	TIN44		
		MSC1 input	INJ10		
	O	General-purpose output	P23_OUT.P3		1X000 _B
		GTM output	TOUT44		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
Reserved		–	1X110 _B		
Reserved		–	1X111 _B		
P23.4	I	General-purpose input	P23_IN.P4	P23_IOCRO4. PC4	0XXXX _B
		GTM input	TIN45		
	O	General-purpose output	P23_OUT.P4		1X000 _B
		GTM output	TOUT45		1X001 _B
		Reserved	–		1X010 _B
		QSPI3 output	SLSO35		1X011 _B
		MSC1 output	END12		1X100 _B
		MSC1 output	EN10		1X101 _B
		Reserved	–		1X110 _B
Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Table 14-30 Port 23 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P23.5	I	General-purpose input	P23_IN.P5	P23_IOC4. PC5	0XXXX _B
		GTM input	TIN46		
	O	General-purpose output	P23_OUT.P5		1X000 _B
		GTM output	TOUT46		1X001 _B
		Reserved	–		1X010 _B
		QSPI3 output	SLSO34		1X011 _B
		MSC1 output	END13		1X100 _B
		MSC1 output	EN11		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.14.3 Port 23 Registers

The following registers are available on Port 23:

Table 14-31 Port 23 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P23_OUT	Port 23 Output Register	0000 _H	Page 14-196 ²⁾
P23_OMR	Port 23 Output Modification Register	0004 _H	Page 14-200 ²⁾
P23_IOCRO	Port 23 Input/Output Control Register 0	0010 _H	Page 14-14
P23_IOCRR4	Port 23 Input/Output Control Register 4	0014 _H	Page 14-197 ²⁾
P23_IN	Port 23 Input Register	0024 _H	Page 14-200 ²⁾
P23_PDR0	Port 23 Pad Driver Mode 0 Register	0040 _H	Page 14-199 ²⁾
P23_ESR	Port 23 Emergency Stop Register	0050 _H	Page 14-200 ²⁾
P23_OMSR0	Port 23 Output Modification Set Register 0	0070 _H	Page 14-41
P23_OMSR4	Port 23 Output Modification Set Register 4	0074 _H	Page 14-200 ²⁾
P23_OMCR0	Port 23 Output Modification Clear Register 0	0080 _H	Page 14-46
P23_OMCR4	Port 23 Output Modification Clear Register 4	0084 _H	Page 14-200 ²⁾
P23_OMSR	Port 23 Output Modification Set Register	0090 _H	Page 14-200 ²⁾
P23_OMCR	Port 23 Output Modification Clear Register	0094 _H	Page 14-200 ²⁾
P23_ACCEN1	Port 23 Access Enable Register 1	00F8 _H	Page 14-54
P23_ACCEN0	Port 23 Access Enable Register 0	00FC _H	Page 14-53

1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))

2) These registers are listed and noted here in the Port 23 section because they differ from the general port register description given in [Section 14.3](#).

14.14.3.1 Port 23 Output Register

The basic P23_OUT register functionality is described on [Page 14-36](#). Port line P23.[15:6] are not connected. Reading the P23_OUT bits P[15:8] always return 0, reading P[7:6] returns the value that was last written. These connected bits can be also set/reset by the corresponding bits in P23_OMR, P23_OMSR, P23_OMCR, P23_OMSRx (x=0,4), P23_OMCRx (x=0,4).

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.14.3.2 Port 23 Input/Output Control Register 4

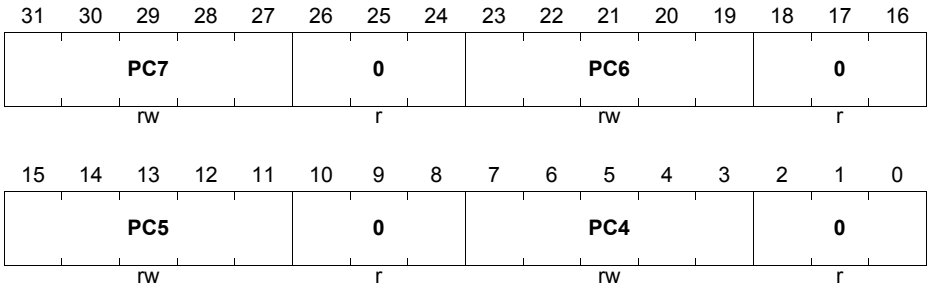
The PC[7:6] bit fields in register P23_IOCRA are not connected.

P23_IOCRA

Port 23 Input/Output Control Register 4

(14_H)

Reset Value: 1010 1010_H



Field	Bits	Type	Description
PC4	[7:3]	rw	Port Control for Port 23 Pin 4
PC5	[15:11]	rw	Port Control for Port 23 Pin 5
PC6, PC7	[23:19], [31:27]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P23_IOCRA
Port 23 Input/Output Control Register 4

 (14_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC7				0				PC6				0			
rw				r				rw				r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC5				0				PC4				0			
rw				r				rw				r			

Field	Bits	Type	Description
PC4	[7:3]	rw	Port Control for Port 23 Pin 4
PC5	[15:11]	rw	Port Control for Port 23 Pin 5
PC6, PC7	[23:19], [31:27]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.14.3.3 Port 23 Pad Driver Mode 0 Register

The basic P23_PDR0 register functionality is described on [Page 14-25](#).

P23_PDR0
Port 23 Pad Driver Mode 0 Register (40_H)
Reset Value: 3333 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL7	PD7		PL6	PD6		PL5	PD5		PL4	PD4					
rw	rw		rw	rw		rw	rw		rw	rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL3	PD3		PL2	PD2		PL1	PD1		PL0	PD0					
rw	rw		rw	rw		rw	rw		rw	rw					

Field	Bits	Type	Description
PD0	[2:0]	rw	Pad Driver Mode for Port 23 Pin 0
PD1	[6:4]	rw	Pad Driver Mode for Port 23 Pin 1
PD2	[10:8]	rw	Pad Driver Mode for Port 23 Pin 2
PD3	[14:12]	rw	Pad Driver Mode for Port 23 Pin 3
PD4	[18:16]	rw	Pad Driver Mode for Port 23 Pin 4
PD5	[22:20]	rw	Pad Driver Mode for Port 23 Pin 5
PL0	3	rw	Pad Level Selection for Port 23 Pin 0
PL1	7	rw	Pad Level Selection for Port 23 Pin 1
PL2	11	rw	Pad Level Selection for Port 23 Pin 2
PL3	15	rw	Pad Level Selection for Port 23 Pin 3
PL4	19	rw	Pad Level Selection for Port 23 Pin 4
PL5	23	rw	Pad Level Selection for Port 23 Pin 5
PD6, PD7	[26:24], [30:28]	rw	Reserved Read as 000 _B after reset; returns value that was written.
PL6, PL7	27, 31	rw	Reserved Read as 0 _B after reset; returns value that was written.

14.14.3.4 Port 23 Output Modification Register

The basic P23_OMR register functionality is described on [Page 14-38](#). However, port lines P23.[15:6] are not connected. The P23_OMR bits PS[7:6] and PCL[7:6] have no direct effect on port lines but only on register bits P23_OUT.P[7:6].

14.14.3.5 Port 23 Output Modification Set Register

The basic P23_OMSR register functionality is described on [Page 14-40](#). However, port lines P23.[15:6] is not connected. The P23_OMSR bits PS[7:6] have no direct effect on port lines but only on register bits P23_OUT.P[7:6].

14.14.3.6 Port 23 Output Modification Set Register 4

The basic P23_OMSR4 register functionality is described on [Page 14-42](#). However, port lines P23.[7:6] are not connected. The P23_OMSR4 bits PS[7:6] have no direct effect on port lines but only on register bits P23_OUT.P[7:6].

14.14.3.7 Port 23 Output Modification Clear Register

The basic P23_OMCR register functionality is described on [Page 14-45](#). However, port lines P23.[15:6] are not connected. The P23_OMCR bits PCL[7:6] have no direct effect on port lines but only on register bits P23_OUT.P[7:6].

14.14.3.8 Port 23 Output Modification Clear Register 4

The basic P23_OMCR4 register functionality is described on [Page 14-47](#). However, port lines P23.[7:6] are not connected. The P23_OMCR4 bits PCL[7:6] have no direct effect on port lines but only on register bits P23_OUT.P[7:6].

14.14.3.9 Port 23 Input Register

The basic P23_IN register functionality is described on [Page 14-52](#). However, port lines P23.[15:6] are not connected. Therefore, bits P[15:6] in register P23_IN are always read as 0.

14.14.3.10 Port 23 Emergency Stop Register

The basic P23_ESR register functionality is described on [Page 14-50](#). Port lines P23.[15:6] are not connected. Reading the EN[15:8] always returns 0. Reading the P15_ESR bits EN[7:6] returns value that was last written.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.15 Port 32

This section describes the Port 32 functionality.

14.15.1 Port 32 Configuration

Port 32 is a 6-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, ASCLIN, MultiCAN+, SCU, CCU6, PMU, OCDS and MSC functions.

14.15.2 Port 32 Function Table

Table 14-32 summarizes the I/O control selection functions of each Port 32 line.

Table 14-32 Port 32 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P32.0	I	General-purpose input	P32_IN.P0	P32_IOCRO. PC0	0XXXX _B
		GTM input	TIN36		
		PMU input	FDEST		
			VGATE1N		
	O	General-purpose output	P32_OUT.P0		1X000 _B
		GTM output	TOUT36		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–	1X110 _B	
		Reserved	–	1X111 _B	

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-32 Port 32 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.			
				Reg./Bit Field	Value		
P32.2	I	General-purpose input	P32_IN.P2	P32_IOCRO. PC2	0XXXX _B		
		GTM input	TIN38				
		ASCLIN3 input	ARX3D				
		CAN node 3 input	RXDCAN3B				
	O	General-purpose output	P32_OUT.P2		1X000 _B		
		GTM output	TOUT38		1X001 _B		
		ASCLIN3 output	ATX3		1X010 _B		
		Reserved	–		1X011 _B		
		Reserved	–		1X100 _B		
		Reserved	–		1X101 _B		
		Reserved	–		1X110 _B		
		Reserved	–		1X111 _B		
	P32.3	I	General-purpose input		P32_IN.P3	P32_IOCRO. PC3	0XXXX _B
			GTM input		TIN39		
O		General-purpose output	P32_OUT.P3	1X000 _B			
		GTM output	TOUT39	1X001 _B			
		ASCLIN3 output	ATX3	1X010 _B			
		Reserved	–	1X011 _B			
		ASCLIN3 output	ASCLK3	1X100 _B			
		CAN node 3 output	TXDCAN3	1X101 _B			
		Reserved	–	1X110 _B			
		Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-32 Port 32 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.			
				Reg./Bit Field	Value		
P32.4	I	General-purpose input	P32_IN.P4	P32_IOC4. PC4	0XXXX _B		
		GTM input	TIN40				
		MSC1 input	SDI12				
		ASCLIN1 input	ACTS1B				
	O	General-purpose output	P32_OUT.P4		1X000 _B		
		GTM output	TOUT40		1X001 _B		
		Reserved	–		1X010 _B		
		MSC1 output	END12		1X011 _B		
		GTM output	GTMCLK1		1X100 _B		
		MSC1 output	EN10		1X101 _B		
		SCU output	EXTCLK1		1X110 _B		
		CCU60 output	COUT63		1X111 _B		
	P32.6 1)	I	General-purpose input		P32_IN.P6	P32_IOC4. PC6	0XXXX _B
			OCDS input		$\overline{\text{TGI4}}$		
O		General-purpose output	P32_OUT.P6	1X000 _B			
		Reserved	–	1X001 _B			
		Reserved	–	1X010 _B			
		Reserved	–	1X011 _B			
		Reserved	–	1X100 _B			
		Reserved	–	1X101 _B			
		Reserved	–	1X110 _B			
		Reserved	–	1X111 _B			
		Reserved	–	1X111 _B			
DIRx	OCDS; ENx	$\overline{\text{TGO4}}$	HW_OUT				

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Table 14-32 Port 32 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P32.7 ¹)	I	General-purpose input	P32_IN.P7	P32_IOC4. PC7	0XXXX _B
		OCDS input	$\overline{TGI5}$		
	O	General-purpose output	P32_OUT.P7		1X000 _B
		Reserved	–		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B
DIRx	OCDS; ENx	$\overline{TGO5}$	HW_OUT		

1) Only available in Emulation Device in the PG-TQFP-216-1 package

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.15.3 Port 32 Registers

The following registers are available on Port 32:

Table 14-33 Port 32 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P32_OUT	Port 10 Output Register	0000 _H	Page 14-206 ²⁾
P32_OMR	Port 10 Output Modification Register	0004 _H	Page 14-206 ²⁾
P32_IOCRO	Port 10 Input/Output Control Register 0	0010 _H	Page 14-207 ²⁾
P32_IOCRR4	Port 10 Input/Output Control Register 4	0014 _H	Page 14-209 ²⁾
P32_IN	Port 10 Input Register	0024 _H	Page 14-212 ²⁾
P32_PDR0	Port 10 Pad Driver Mode 0 Register	0040 _H	Page 14-211 ²⁾
P32_ESR	Port 10 Emergency Stop Register	0050 _H	Page 14-213 ²⁾
P32_OMSR0	Port 10 Output Modification Set Register 0	0070 _H	Page 14-212 ²⁾
P32_OMSR4	Port 10 Output Modification Set Register 4	0074 _H	Page 14-212 ²⁾
P32_OMCR0	Port 10 Output Modification Clear Register 0	0080 _H	Page 14-212 ²⁾
P32_OMCR4	Port 10 Output Modification Clear Register 4	0084 _H	Page 14-212 ²⁾
P32_OMSR	Port 10 Output Modification Set Register	0090 _H	Page 14-212 ²⁾
P32_OMCR	Port 10 Output Modification Clear Register	0094 _H	Page 14-212 ²⁾
P32_ACCEN1	Port 10 Access Enable Register 1	00F8 _H	Page 14-54
P32_ACCEN0	Port 10 Access Enable Register 0	00FC _H	Page 14-53

1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

- 2) These registers are listed and noted here in the Port 32 section because they differ from the general port register description given in [Section 14.3](#).

14.15.3.1 Port 32 Output Register

The basic P32_OUT register functionality is described on [Page 14-36](#). Port lines P32.1, P32.5, P32.[15:8] are not connected. Reading the P32_OUT bits P1, P5 returns the value that was last written. Reading P[15:8] always return 0. These connected bits can be also set/reset by the corresponding bits in P32_OMR, P32_OMSR, P32_OMCR, P32_OMSRx (x=0,4), P32_OMCRx (x=0,4).

14.15.3.2 Port 32 Output Modification Register

The basic P32_OMR register functionality is described on [Page 14-38](#). However, port lines P32.1, P32.5, P32.[15:8] are not connected. The P32_OMR bits PS1, PS5, and PCL1, PCL5 have no direct effect on port lines but only on register bits P32_OUT.P1, P5.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.15.3.3 Port 32 Input/Output Control Register 0

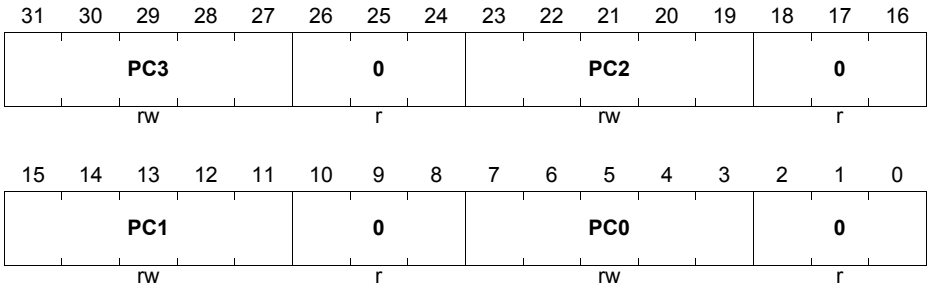
The PC1 bit field in register P32_IOCRO is not connected.

P32_IOCRO

Port 32 Input/Output Control Register 0

(10_H)

Reset Value: 1010 1010_H



Field	Bits	Type	Description
PC0	[7:3]	rw	Port Control for Port n Pin 0
PC2	[23:19]	rw	Port Control for Port n Pin 2
PC3	[31:27]	rw	Port Control for Port n Pin 3
PC1	[15:11]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P32_IOCR0
Port 32 Input/Output Control Register 0

 (10_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC3				0				PC2				0			
rw				r				rw				r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC1					0					PC0					0
rw					r					rw					r

Field	Bits	Type	Description
PC0	[7:3]	rw	Port Control for Port n Pin 0
PC2	[23:19]	rw	Port Control for Port n Pin 2
PC3	[31:27]	rw	Port Control for Port n Pin 3
PC1	[15:11]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

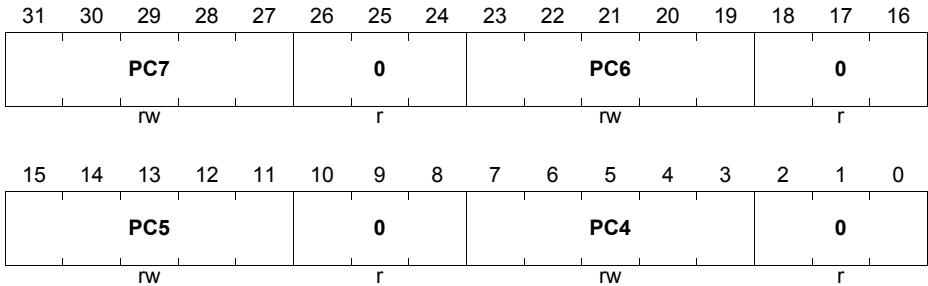
General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.15.3.4 Port 32 Input/Output Control Register 4

The PC[7:5] bit fields in register P32_IOC4 are not connected.

P32_IOC4
Port 32 Input/Output Control Register 4

 (14_H)

 Reset Value: 1010 1010_H


Field	Bits	Type	Description
PC4, PC6, PC7	[7:3], [23:19], [31:27]	rw	Port Control for Port 32 Pin 4, 6 to 7
PC5	[15:11]	rw	Reserved Read as 00010 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

P32_IOCR4
Port 32 Input/Output Control Register 4

 (14_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC7				0				PC6				0			
rw				r				rw				r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC5					0			PC4					0		
rw					r			rw					r		

Field	Bits	Type	Description
PC4, PC6, PC7	[7:3], [23:19], [31:27]	rw	Port Control for Port 32 Pin 4, 6 to 7
PC5	[15:11]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.15.3.5 Port 32 Pad Driver Mode 0 Register

 The basic P32_PDR0 register functionality is described on [Page 14-25](#).

P32_PDR0
Port 32 Pad Driver Mode 0 Register (40_H)
Reset Value: 3333 3333_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PL7	PD7			PL6	PD6			PL5	PD5			PL4	PD4		
rw		rw			rw		rw			rw		rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PL3	PD3			PL2	PD2			PL1	PD1			PL0	PD0		
rw		rw			rw		rw			rw		rw			

Field	Bits	Type	Description
PD0	[2:0]	rw	Pad Driver Mode for Port 32 Pin 0
PD2	[10:8]	rw	Pad Driver Mode for Port 32 Pin 2
PD3	[14:12]	rw	Pad Driver Mode for Port 32 Pin 3
PD4	[18:16]	rw	Pad Driver Mode for Port 32 Pin 4
PD6	[26:24]	rw	Pad Driver Mode for Port 32 Pin 6
PD7	[30:28]	rw	Pad Driver Mode for Port 32 Pin 7
PL0	3	rw	Pad Level Selection for Port 32 Pin 0
PL2	11	rw	Pad Level Selection for Port 32 Pin 2
PL3	15	rw	Pad Level Selection for Port 32 Pin 3
PL4	19	rw	Pad Level Selection for Port 32 Pin 4
PL6	27	rw	Pad Level Selection for Port 32 Pin 6
PL7	31	rw	Pad Level Selection for Port 32 Pin 7
PD1, PD5	[6:4], [22:20]	rw	Reserved Read as 000 _B after reset; returns value that was written.
PL1, PL5	7, 23	rw	Reserved Read as 0 _B after reset; returns value that was written.

14.15.3.6 Port 32 Output Modification Set Register

The basic P32_OMSR register functionality is described on [Page 14-40](#). However, port lines P32.1, P32.5, P32.[15:8] are not connected. The P32_OMSR bits PS1, PS5 have no direct effect on port lines but only on register bits P32_OUT.P1, P5.

14.15.3.7 Port 32 Output Modification Set Register 0

The basic P32_OMSR0 register functionality is described on [Page 14-41](#). However, port lines P32.1 is not connected. The P32_OMSR0 bits PS1 has no direct effect on port lines but only on register bits P32_OUT.P1.

14.15.3.8 Port 32 Output Modification Set Register 4

The basic P32_OMSR4 register functionality is described on [Page 14-42](#). However, port lines P32.5 is not connected. The P32_OMSR4 bits PS5 has no direct effect on port lines but only on register bits P32_OUT.P5.

14.15.3.9 Port 32 Output Modification Clear Register

The basic P32_OMCR register functionality is described on [Page 14-45](#). However, port lines P32.1, P32.5, P32.[15:8] are not connected. The P32_OMCR bits PCL1, PCL5 have no direct effect on port lines but only on register bits P32_OUT.P32_OUT.P1, P5.

14.15.3.10 Port 32 Output Modification Clear Register 0

The basic P32_OMCR0 register functionality is described on [Page 14-46](#). However, port lines P32.1 are not connected. The P32_OMCR0 bits PCL1 has no direct effect on port lines but only on register bits P32_OUT.P1.

14.15.3.11 Port 32 Output Modification Clear Register 4

The basic P32_OMCR4 register functionality is described on [Page 14-47](#). However, port lines P32.5 are not connected. The P32_OMCR4 bits PCL5 has no direct effect on port lines but only on register bits P32_OUT.P5.

14.15.3.12 Port 32 Input Register

The basic P32_IN register functionality is described on [Page 14-52](#). However, port lines P32.1, P32.5, P32.[15:8] are not connected. Therefore, bits P1, P5, P[15:8] in register P32_IN are always read as 0.

14.15.3.13 Port 32 Emergency Stop Register

The basic P32_ESR register functionality is described on [Page 14-50](#). Port lines P32.1, P32.5, P32.[15:8] are not connected. Reading EN1, EN5 returns the value that was last written, reading EN[15:8] always returns 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.16 Port 33

This section describes the Port 33 functionality.

14.16.1 Port 33 Configuration

Port 33 is a 16-bit bi-directional general-purpose I/O port that can be alternatively used for GTM, CCU6, PSI5, PSI5-S, SENT, DSADC, ASCLIN, VADC, GPT12, QSPI, MultiCAN+, SCU, SMU, OCDS and MSC functions.

14.16.2 Port 33 Function Table

Table 14-34 summarizes the I/O control selection functions of each Port 33 line.

Table 14-34 Port 33 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P33.0	I	General-purpose input	P33_IN.P0	P33_IOCRO. PC0	0XXXX _B
		GTM input	TIN22		
		DSADC input	DSITR0E		
	O	General-purpose output	P33_OUT.P0		1X000 _B
		GTM output	TOUT22		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		VADC output	VADCG1BFL0		1X110 _B
		Reserved	–		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-34 Port 33 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.			
				Reg./Bit Field	Value		
P33.1	I	General-purpose input	P33_IN.P1	P33_IOCR0. PC1	0XXXX _B		
		GTM input	TIN23				
		PSI5 input	PSIRX0C				
		DSADC input	DSCIN2B				
	O	General-purpose output	P33_OUT.P1		1X000 _B		
		GTM output	TOUT23		1X001 _B		
		ASCLIN3 output	ASLSO3		1X010 _B		
		Reserved	–		1X011 _B		
		DSADC output	DSCOUT2		1X100 _B		
		VADC output	VADCEMUX02		1X101 _B		
		VADC output	VADCG1BFL1		1X110 _B		
		Reserved	–		1X111 _B		
	P33.2	I	General-purpose input		P33_IN.P2	P33_IOCR0. PC2	0XXXX _B
			GTM input		TIN24		
DSADC input			DSDIN2B				
DSADC input			DSITR2E				
O		General-purpose output	P33_OUT.P2	1X000 _B			
		GTM output	TOUT24	1X001 _B			
		ASCLIN3 output	ASCLK3	1X010 _B			
		Reserved	–	1X011 _B			
		PSI5 output	PSITX0	1X100 _B			
		VADC output	VADCEMUX01	1X101 _B			
		VADC output	VADCG1BFL2	1X110 _B			
		Reserved	–	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-34 Port 33 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P33.3	I	General-purpose input	P33_IN.P3	P33_IOCR0. PC3	0XXXX _B
		GTM input	TIN25		
		PSI5 input	PSIRX1C		
	O	General-purpose output	P33_OUT.P3		1X000 _B
		GTM output	TOUT25		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		VADC output	VADCEMUX00		1X101 _B
		VADC output	VADCG1BFL3		1X110 _B
Reserved	–	1X111 _B			
P33.4	I	General-purpose input	P33_IN.P4	P33_IOCR4. PC4	0XXXX _B
		GTM input	TIN26		
		CCU61 input	CTRAPC		
		DSADC input	DSITR0F		
	O	General-purpose output	P33_OUT.P4		1X000 _B
		GTM output	TOUT26		1X001 _B
		ASCLIN2 output	ARTS2		1X010 _B
		Reserved	–		1X011 _B
		PSI5 output	PSITX1		1X100 _B
		VADC output	VADCEMUX12		1X101 _B
		VADC output	VADCG0BFL0		1X110 _B
		Reserved	–		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-34 Port 33 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P33.5	I	General-purpose input	P33_IN.P5	P33_IOCRA. PC5	0XXXX _B
		GTM input	TIN27		
		PSI5 input	PSIRX2C		
		SENT input	SENT5C		
		CCU61 input	CCPOS2C		
		GPT120 input	T4EUDB		
		DSADC input	DSCIN0B		
		ASCLIN2 input	ACTS2B		
	PSI5-S input	PSISRXC			
	O	General-purpose output	P33_OUT.P5		1X000 _B
		GTM output	TOUT27		1X001 _B
		QSPI0 output	SLSO07		1X010 _B
		QSPI1 output	SLSO17		1X011 _B
		DSADC output	DSCOUT0		1X100 _B
		VADC output	VADCEMUX11		1X101 _B
VADC output		VADCG0BFL1		1X110 _B	
Reserved	–		1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-34 Port 33 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P33.6	I	General-purpose input	P33_IN.P6	P33_IOCRA. PC6	0XXXX _B
		GTM input	TIN28		
		SENT input	SENT4C		
		CCU61 input	CCPOS1C		
		GPT120 input	T2EUDB		
		DSADC input	DSDIN0B		
		DSADC input	DSITR2F		
	O	General-purpose output	P33_OUT.P6		1X000 _B
		GTM output	TOUT28		1X001 _B
		ASCLIN2 output	ASLSO2		1X010 _B
		Reserved	–		1X011 _B
		PSI5 output	PSITX2		1X100 _B
		VADC output	VADCEMUX10		1X101 _B
		VADC output	VADCG0BFL2		1X110 _B
		PSI5-S output	PSISTX		1X111 _B
P33.7	I	General-purpose input	P33_IN.P7	P33_IOCRA. PC7	0XXXX _B
		GTM input	TIN29		
		CAN node 0 input	RXDCAN0E		
		SCU input	REQ8		
		CCU61 input	CCPOS0C		
		GPT120 input	T2INB		
	O	General-purpose output	P33_OUT.P7		1X000 _B
		GTM output	TOUT29		1X001 _B
		ASCLIN2 output	ASCLK2		1X010 _B
		QSPI3 output	SLSO37		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		VADC output	VADCG0BFL3		1X110 _B
		Reserved	–		1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-34 Port 33 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
P33.8	I	General-purpose input	P33_IN.P8	P33_IOCR8. PC8	0XXXX _B	
		GTM input	TIN30			
		SCU input	EMGSTOPA			
		ASCLIN2 input	ARX2E			
	O	General-purpose output	P33_OUT.P8		1X000 _B	
		GTM output	TOUT30		1X001 _B	
		ASCLIN2 output	ATX2		1X010 _B	
		QSPI3 output	SLSO32		1X011 _B	
		Reserved	–		1X100 _B	
		CAN node 0 output	TXDCAN0		1X101 _B	
		Reserved	–		1X110 _B	
		CCU61 output	COU62		1X111 _B	
	DIRx	SMU	SMUFSP		HW_OUT	
	P33.9	I	General-purpose input		P33_IN.P9	P33_IOCR8. PC9
GTM input			TIN31			
QSPI3 input			HSIC3INA			
O		General-purpose output	P33_OUT.P9	1X000 _B		
		GTM output	TOUT31	1X001 _B		
		ASCLIN2 output	ATX2	1X010 _B		
		QSPI3 output	SLSO31	1X011 _B		
		ASCLIN2 output	ASCLK2	1X100 _B		
		Reserved	–	1X101 _B		
		Reserved	–	1X110 _B		
		CCU61 output	CC62	1X111 _B		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-34 Port 33 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.			
				Reg./Bit Field	Value		
P33.10	I	General-purpose input	P33_IN.P10	P33_IOCR8. PC10	0XXXX _B		
		GTM input	TIN32				
		QSPI3 input	SLSI3C				
		QSPI3 input	HSIC3INB				
	O	General-purpose output	P33_OUT.P10		1X000 _B		
		GTM output	TOUT32		1X001 _B		
		QSPI1 output	SLSO16		1X010 _B		
		QSPI3 output	SLSO311		1X011 _B		
		ASCLIN1 output	ASLSO1		1X100 _B		
		PSI5-S output	PSISCLK		1X101 _B		
		Reserved	–		1X110 _B		
		CCU61 output	COU61		1X111 _B		
	P33.11	I	General-purpose input		P33_IN.P11	P33_IOCR8. PC11	0XXXX _B
			GTM input		TIN33		
QSPI3 input			SCLK3D				
O		General-purpose output	P33_OUT.P11	1X000 _B			
		GTM output	TOUT33	1X001 _B			
		ASCLIN1 output	ASCLK1	1X010 _B			
		QSPI3 output	SCLK3	1X011 _B			
		Reserved	–	1X100 _B			
		Reserved	–	1X101 _B			
		DSADC output	DSCGPWMN	1X110 _B			
		CCU61 output	CC61	1X111 _B			

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-34 Port 33 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.		
				Reg./Bit Field	Value	
P33.12	I	General-purpose input	P33_IN.P12	P33_IOC12. PC12	0XXXX _B	
		GTM input	TIN34			
		QSPI3 input	MTSR3D			
	O	General-purpose output	P33_OUT.P12			1X000 _B
		GTM output	TOUT34			1X001 _B
		ASCLIN1 output	ATX1			1X010 _B
		QSPI3 output	MTSR3			1X011 _B
		ASCLIN1 output	ASCLK1			1X100 _B
		Reserved	–			1X101 _B
		DSADC output	DSCGPWMP			1X110 _B
CCU61 output	COU60	1X111 _B				
P33.13	I	General-purpose input	P33_IN.P13	P33_IOC12. PC13	0XXXX _B	
		GTM input	TIN35			
		QSPI3 input	MRST3D			
		DSADC input	DSSGNB			
		MSC1 input	INJ11			
		ASCLIN1 input	ARX1F			
	O	General-purpose output	P33_OUT.P13			1X000 _B
		GTM output	TOUT35			1X001 _B
		ASCLIN1 output	ATX1			1X010 _B
		QSPI3 output	MRST3			1X011 _B
		QSPI2 output	SLSO26			1X100 _B
		Reserved	–			1X101 _B
		SCU output	DCDCSYNC			1X110 _B
		CCU61 output	CC60			1X111 _B

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
Table 14-34 Port 33 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P33.14 1)	I	General-purpose input	P33_IN.P14	P33_IOC12. PC14	0XXXX _B
		OCDS input	$\overline{TGI6}$		
	O	General-purpose output	P33_OUT.P14		1X000 _B
		Reserved	–		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B
DIRx	OCDS; ENx	$\overline{TGO6}$	HW_OUT		
P33.15 ¹⁾	I	General-purpose input	P33_IN.P15	P33_IOC12. PC15	0XXXX _B
		OCDS input	$\overline{TGI7}$		
	O	General-purpose output	P33_OUT.P15		1X000 _B
		Reserved	–		1X001 _B
		Reserved	–		1X010 _B
		Reserved	–		1X011 _B
		Reserved	–		1X100 _B
		Reserved	–		1X101 _B
		Reserved	–		1X110 _B
		Reserved	–		1X111 _B
DIRx	OCDS; ENx	$\overline{TGO7}$	HW_OUT		

1) Only available in Emulation Device in the PG-TQFP-216-1 and PG-TQFP-172-1 packages

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.16.3 Port 33 Registers

The following registers are available on Port 33:

Table 14-35 Port 33 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P33_OUT	Port 33 Output Register	0000 _H	Page 14-36
P33_OMR	Port 33 Output Modification Register	0004 _H	Page 14-38
P33_IOCRO	Port 33 Input/Output Control Register 0	0010 _H	Page 14-14
P33_IOCRA	Port 33 Input/Output Control Register 4	0014 _H	Page 14-16
P33_IOCRA8	Port 33 Input/Output Control Register 8	0018 _H	Page 14-18
P33_IOCRA12	Port 33 Input/Output Control Register 12	001C _H	Page 14-20
P33_IN	Port 33 Input Register	0024 _H	Page 14-52
P33_PDR0	Port 33 Pad Driver Mode 0 Register	0040 _H	Page 14-25
P33_PDR1	Port 33 Pad Driver Mode 1 Register	0044 _H	Page 14-26
P33_ESR	Port 33 Emergency Stop Register	0050 _H	Page 14-50
P33_OMSR0	Port 33 Output Modification Set Register 0	0070 _H	Page 14-41
P33_OMSR4	Port 33 Output Modification Set Register 4	0074 _H	Page 14-42
P33_OMSR8	Port 33 Output Modification Set Register 8	0078 _H	Page 14-43
P33_OMSR12	Port 33 Output Modification Set Register 12	007C _H	Page 14-44
P33_OMCR0	Port 33 Output Modification Clear Register 0	0080 _H	Page 14-46
P33_OMCR4	Port 33 Output Modification Clear Register 4	0084 _H	Page 14-47
P33_OMCR8	Port 33 Output Modification Clear Register 8	0088 _H	Page 14-48
P33_OMCR12	Port 33 Output Modification Clear Register 12	008C _H	Page 14-49
P33_OMSR	Port 33 Output Modification Set Register	0090 _H	Page 14-40
P33_OMCR	Port 33 Output Modification Clear Register	0094 _H	Page 14-45
P33_ACCEN1	Port 33 Access Enable Register 1	00F8 _H	Page 14-54
P33_ACCEN0	Port 33 Access Enable Register 0	00FC _H	Page 14-53

1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))

14.16.3.1 Port 33 Emergency Stop Register

The basic P33_ESR register functionality is described on [Page 14-50](#). EN8 is reserved and has no influence on the emergency control function of P33.8, should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)**14.17 Port 40**

This section describes the Port 40 functionality in detail.

14.17.1 Port 40 Configuration

Port 40 is a 8-bit input port.

Table 14-36 summarizes the input control selection functions of each Port 40 line.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.17.2 Port 40 Function Table

Table 14-36 summarizes the input control selection functions of each Port 40 line. If digital input functionality is selected, the digital input needs to meet the criteria defined for the Class S pads.

Table 14-36 Port 40 Functions

Port Pin	I/O	Pin Functionality	Associated Reg./ Input Line	Port Functionality Control Select	
				Reg./Bit Field	Value
P40.0	I	VADC input	VADCG1.8	P40_PDISC.PDIS0	1 _B
		CCU60 input	CCPOS0D		0 _B
		SENT input	SENT0A		
P40.1	I	VADC input	VADCG1.9	P40_PDISC.PDIS1	1 _B
		CCU60 input	CCPOS1B		0 _B
		SENT input	SENT1A		
P40.2	I	VADC input	VADCG1.10	P40_PDISC.PDIS2	1 _B
		CCU60 input	CCPOS1D		0 _B
		SENT input	SENT2A		
P40.3	I	VADC input	VADCG1.11	P40_PDISC.PDIS3	1 _B
		CCU60 input	CCPOS2B		0 _B
		SENT input	SENT3A		
P40.6	I	VADC/DSADC input	VADCG2.4 / DS3PA	P40_PDISC.PDIS6	1 _B
		CCU61 input	CCPOS1B		0 _B
		SENT input	SENT2D		
P40.7	I	VADC/DSADC input	VADCG2.5 / DS3NA	P40_PDISC.PDIS7	1 _B
		CCU61 input	CCPOS1D		0 _B
		SENT input	SENT3D		
P40.8	I	VADC/DSADC input	VADCG2.6 / DS3PB	P40_PDISC.PDIS8	1 _B
		CCU61 input	CCPOS2B		0 _B
		SENT input	SENT4A		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Table 14-36 Port 40 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ Input Line	Port Functionality Control Select	
				Reg./Bit Field	Value
P40.9	I	VADC/DSADC input	VADCG2.7 / DS3NB	P40_PDISC.PDIS9	1 _B
		CCU61 input	CCPOS2D		0 _B
		SENT input	SENT5A		

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.17.3 Port 40 Registers

The following registers are available on Port 40:

Table 14-37 Port 40 Registers

Register Short Name	Register Long Name	Offset¹⁾ Address	Description see
P40_IOCRO	Port 40 Input/Output Control Register 0	0010 _H	Page 14-229²⁾
P40_IOCRR4	Port 40 Input/Output Control Register 4	0014 _H	Page 14-229²⁾
P40_IOCRR8	Port 40 Input/Output Control Register 8	0018 _H	Page 14-229²⁾
P40_IN	Port 40 Input Register	0024 _H	Page 14-52
P40_PDISC	Port 40 Pin Function Decision Control Register	0060 _H	Page 14-231²⁾
P40_PCSR	Port 40 Pin Controller Select Register	0064 _H	Page 14-233²⁾
P40_ACCEN1	Port 40 Access Enable Register 1	00F8 _H	Page 14-54²⁾
P40_ACCEN0	Port 40 Access Enable Register 0	00FC _H	Page 14-53²⁾

1) The absolute addresses are calculated by adding the offset address to the module base address (see [Table 14-3](#))

2) These registers are listed here in the Port 40 section because they differ from the general port register description given in [Section 14.3](#).

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

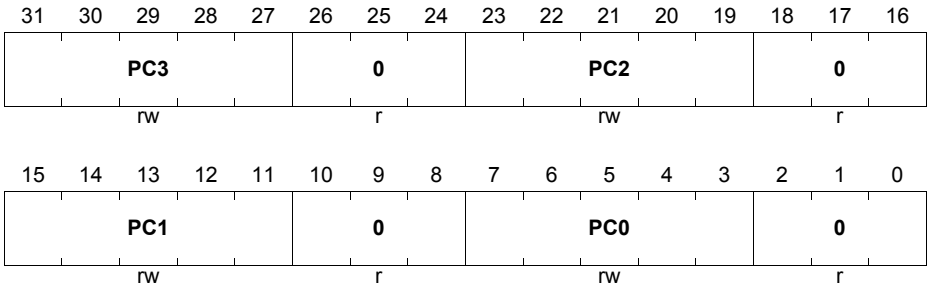
14.17.4 Port 40 Input/Output Control Registers

P40_IOCRO

Port 40 Input/Output Control Register 0

(10_H)

Reset Value: 0000 0000_H



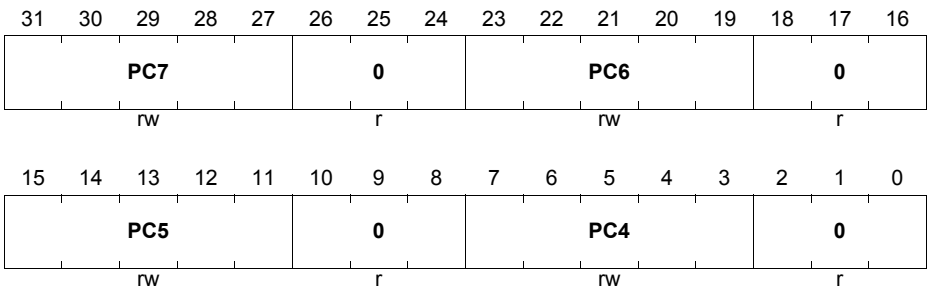
Field	Bits	Type	Description
PC0, PC1, PC2, PC3	[7:3], [15:11], [23:19], [31:27]	rw	Port Control for Port n Pin 0 to 3 This bit field defines the Port n line x functionality according to Table 14-5 , only input selection apply.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

P40_IOCRO4

Port 40 Input/Output Control Register 4

(14_H)

Reset Value: 0000 0000_H



General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PC6, PC7	[23:19], [31:27]	rw	Port Control for Port n Pin 6 to 7 This bit field defines the Port n line x functionality according to Table 14-5 , only input selection apply.
PC4, PC5	[7:3], [15:11]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

P40_IOCR8
Port 40 Input/Output Control Register 8
(18_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC11					0			PC10					0		
rw					r			rw					r		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC9					0			PC8					0		
rw					r			rw					r		

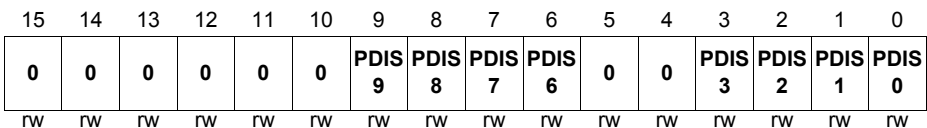
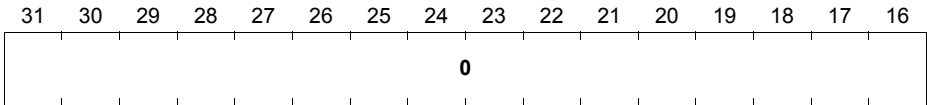
Field	Bits	Type	Description
PC10, PC11	[23:19], [31:27]	rw	Reserved Read as 00000 _B after reset; returns value that was written.
PC8, PC9	[7:3], [15:11]	rw	Port Control for Port 40 Pin 8 to 9 (coding see Table 14-5 , only input selection apply.)
0	[2:0], [10:8], [18:16], [26:24]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

14.17.5 Port 40 Pin Function Decision Control Register

P40_PDISC

Port 40 Pin Function Decision Control Register(60_H) Reset Value: 0000 03CF_H



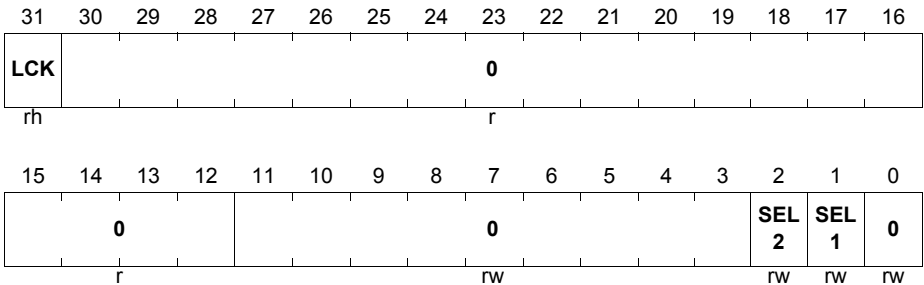
Field	Bits	Type	Description
PDIS0	0	rw	Pin Function Decision Control for Pin 0 The bit selects the function of P40.0 The default state of the pad selects the ADC analog input. 0 _B Pad is used for digital input 0. 1 _B Pad is used for ADC analog input x.
PDIS1	1	rw	Pin Function Decision Control for Pin 1 The bit selects the function of P40.1 The default state of the pad selects the ADC analog input. 0 _B Pad is used for digital input 1. 1 _B Pad is used for ADC analog input x.
PDIS2	2	rw	Pin Function Decision Control for Pin 2 The bit selects the function of P40.2. The default state of the pad selects the ADC analog input. 0 _B Pad is used for digital input 2 1 _B Pad is used for ADC analog input x.
PDIS3	3	rw	Pin Function Decision Control for Pin 3 The bit selects the function of P40.3. The default state of the pad selects the ADC analog input. 0 _B Pad is used for digital input 3. 1 _B Pad is used for ADC analog input x.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

Field	Bits	Type	Description
PDIS6	6	rw	Pin Function Decision Control for Pin 6 The bit selects the function of P40.6. The default state of the pad selects the ADC analog input. 0 _B Pad is used for digital input 6. 1 _B Pad is used for ADC analog input x.
PDIS7	7	rw	Pin Function Decision Control for Pin 7 The bit selects the function of P40.7. The default state of the pad selects the ADC analog input. 0 _B Pad is used for digital input 7. 1 _B Pad is used for ADC analog input x.
PDIS8	8	rw	Pin Function Decision Control for Pin 8 The bit selects the function of P40.8. The default state of the pad selects the ADC analog input. 0 _B Pad is used for digital input 8. 1 _B Pad is used for ADC analog input x.
PDIS9	9	rw	Pin Function Decision Control for Pin 9 The bit selects the function of P40.9. The default state of the pad selects the ADC analog input. 0 _B Pad is used for digital input 9. 1 _B Pad is used for ADC analog input x.
0	[15:10]	rw	Reserved Read as 0; should be written with 0.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)
14.17.6 Port 40 Pin Controller Select Register

P40_PCSR enables or disables the VADC Multiplexer Diagnostics (MD) feature.

P40_PCSR
Port 40 Pin Controller Select Register (64_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SELx (x = 2-1)	0 + x	rw	Pin Controller Select for Pin x This bit enables or disables the VADC Multiplexer Diagnostics (MD) feature. For the respective analog signal connection, see analog connections table in VADC chapter. 0 _B Disables VADC MD feature. 1 _B Enables VADC MD feature.
LCK	31	rh	Lock Status This bit indicates if the register can be updated with a new value or if the register is locked and a write action from the bus has no effect. 0 _B The register is unlocked and can be updated. 1 _B The register is locked and can not be updated.
0	0, [11:3]	rw	Reserved Read as 0; returns value last written, must be written with 0.
0	[30:12]	r	Reserved Read as 0; should be written with 0.

General Purpose I/O Ports and Peripheral I/O Lines (Ports)

15 Direct Memory Access (DMA)

This chapter describes the DMA Controller. It contains the following sections:

- Features (see [Section 15.2](#))
- Block Diagram (see [Section 15.3](#))
- Functional Description (see [Section 15.4](#))
- Power Modes (see [Section 15.5](#))
- Functional Safety Features (see [Section 15.6](#))
- Debug Features (see [Section 15.7](#))
- Register Description (see [Section 15.8](#))
- Use Cases (see [Section 15.9](#))

Note: The DMA kernel register names described in [Section 15.8](#) are referenced in the TC26x B-Step User's Manual by the module name prefix "DMA_".

15.1 What is new

Major differences of the RAM based AURIX DMA compared to the register based AURIX DMA are

- Any DMA move engine can service a DMA request from any DMA channel. There is no fixed mapping of DMA channels to move engines. DMA requests from the highest number DMA channel are serviced first by the DMA controller. The DMA move engine is the module that executes DMA channel requests.
- The DMA channel transaction control set is stored in the DMARAM. The move engines use a copy of the DMARAM channel transaction control set to service DMA requests.
- DMA channels can be assigned to one of four hardware resource partitions that determine access protection of the DMARAM channel transaction control set.
- DMA channels can be daisy chained. On completing a DMA transaction a DMA channel can initiate a DMA transaction in the immediate next lower priority channel.
- DMA channel source and address pointers are 32-bit wide address counters. The source and destination wrap buffers are optionally selectable.
- Support for DMA Linked Lists. The current DMA transaction can load the next DMA transaction control set into the DMARAM by overwriting the existing channel transaction control set. It can optionally auto start the next DMA transaction.
- Support for DMA Double Buffering. The DMA transaction can execute read or fill DMA transfer from one of two source or destination buffers. A control bit allows the re-direction of DMA transfers from the one buffer to the other buffer.
- All DMA channels support safe DMA operation by the addition of logic to calculate checksums during DMA Moves.
- The move engine supports 64-bit data transfers to SRI addresses. Each move engine has its own 256-bit read buffer to access resources in the cached address ranges.
- The DMA kernel is clocked at the SRI clock frequency.

15.2 Features

The DMA controller is a fast and flexible DMA controller that has the following features:

- The DMA controller supports 48 DMA channels:
 - DMA channel 047 has the highest priority.
 - DMA channel 000 has the lowest priority.
- DMA channel hardware requests are sourced from the Interrupt Router (IR) Interrupt Control Unit (ICU). The system architecture defines a dedicated ICU to arbitrate between peripheral interrupt triggers and generate a DMA hardware request.
 - Any peripheral that can trigger an interrupt can initiate a DMA transfer.
- DMA channel software requests.
- The DMA controller supports 2 move engines for the parallel execution of DMA requests:
 - 1 X DMA sub-block (move engine) services 1 X active DMA channel.
 - DMA sub-blocks (move engines) work in parallel.
- The DMA can be hardware configured to support three programmable levels of System Peripheral Bus (SPB) bus priority through the single on chip bus data path to support the optional addition of a Debug Interface.
 - Setting high priority supports enhanced debug access. The debugger wins arbitration at the internal DMA bus switch and over all other bus master requests in order to gain immediate access to the SPB. There is a potential impact on performance.
 - Setting low priority supports non intrusive debugging. The debugger only wins arbitration when no other DMA access requesters or bus masters are requesting access to the bus. There is minimal performance impact.
- Individually programmable operation modes for each DMA channel
- Full 32-bit addressing capability of each DMA channel
 - 4 Gbyte address range.
 - Circular buffer addressing mode with flexible circular buffer sizes
- Data block move throughput
 - DMA transaction moving data from an SRI-source to an SRI-destination supports >8 Mbyte moves per DMA transaction.
 - DMA transaction moving data either from an FPI-source or to an FPI-destination supports >1 Mbyte moves per DMA transaction.
- The DMA transaction control set is stored in the DMARAM.
- DMA read move and DMA write moves are directed by the DMA switch to different sources and destinations depending on the source or destination address.
- Buffer capability for move actions on the buses (at least 1 move per bus is buffered)
- Programmable data width of DMA moves:
 - SPB master interface: 8-bit, 16-bit, or 32-bit.
 - SRI master interface: 8-bit, 16-bit, 32-bit, 64-bit, 128-bit or 256-bit.
- Interrupt Triggers:

Direct Memory Access (DMA)

- Each DMA channel generates one traffic management interrupt trigger with an unique interrupt vector and priority level.
- The DMA generates one error interrupt trigger.
- Operating frequencies:
 - The DMA controller kernel (active channel, move engine and SRI master interface) work at the SRI clock frequency in order to maximise the data throughput for DMA moves from SRI source addresses to SRI destination addresses.
 - The DMA controller configuration sector (slave interface, DMARAM, ICU interface and SPB Master interface) work at the SPB clock frequency.
- Slave based access protection:
 - DMA channel transaction control sets are access protected.
 - Each DMA channel is assigned to a hardware resource partition.
 - Access control to a hardware partition is via enabling of individual Master TAG IDs to have write access to the hardware partition.
 - Each DMA hardware resource partition has an unique Master TAG IDs and is configured to make accesses in supervisor or user mode.

Direct Memory Access (DMA)

15.3 Block Diagram

The DMA Controller transfers data from data source locations to data destination locations without intervention of the CPU or other on chip devices. One data move operation is controlled by an active DMA channel. A DMA sub-block can service DMA requests from any of the DMA channels. The Bus Switch provides the connection between a DMA sub-block and on chip bus interfaces.

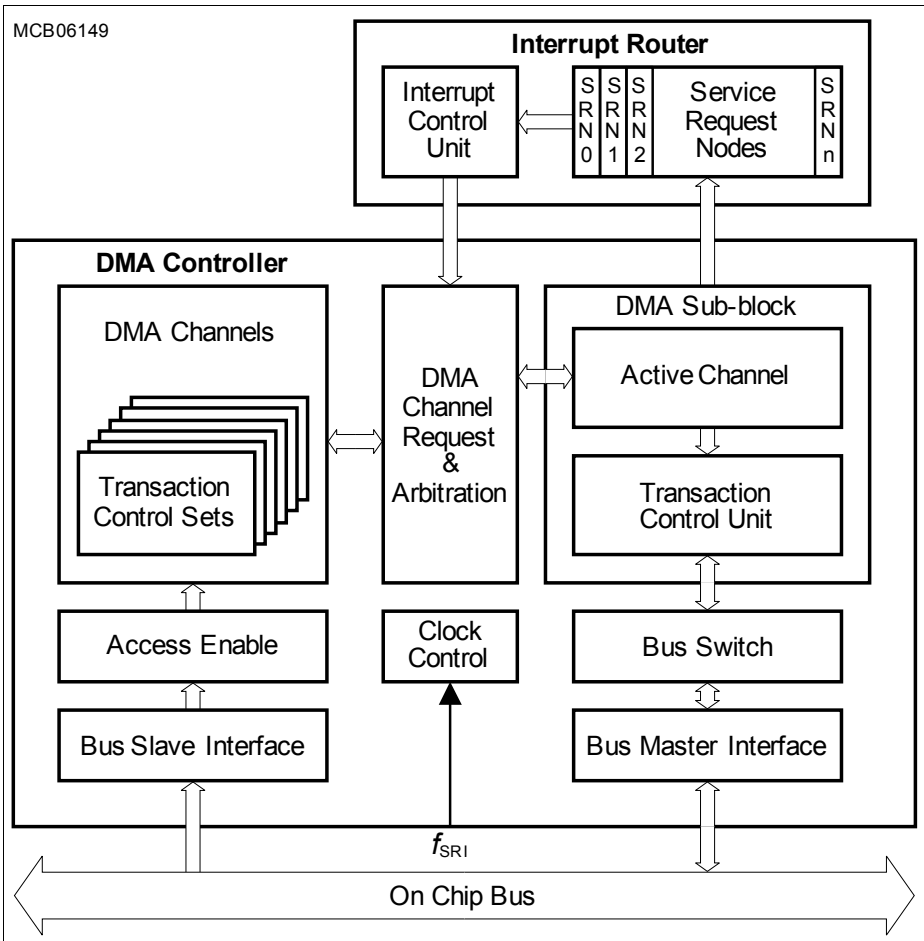


Figure 15-1 DMA Block Diagram

Programming of the DMA transaction control sets is via a bus slave interface. The slave interface supports clock control and access protection to the DMA controller. The Access

Direct Memory Access (DMA)

Enable registers are safe endinit protected and support register monitoring safety measures.

Address decoding and DMA request wiring are managed outside the DMA controller.

15.4 Functional Description

15.4.1 Definition of Terms

Some basic terms must be defined for the functional description of the DMA controller.

DMA Move

A DMA move is an operation that always consists of two parts:

1. A read move that loads data from a data source into the DMA controller
2. A write move that puts data from the DMA controller to a data destination

Within a DMA move, data is always moved from the data source via the DMA controller to the data destination. Data is temporarily stored in the DMA controller. The data widths of read move and write move are typically identical (8-bit, 16-bit, 32-bit, 64-bit, 128-bit or 256-bit).

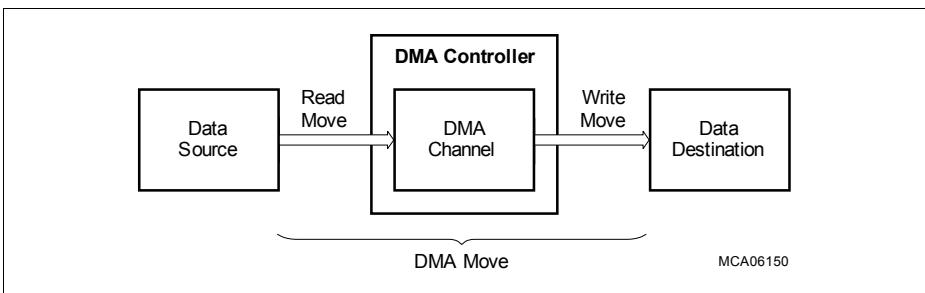


Figure 15-2 DMA Definition of Terms

DMA Transfer

A DMA transfer can be composed of 1, 2, 3, 4, 5, 8, 9 or 16 DMA moves.

DMA Transaction

A DMA transaction is composed of several (at least one) DMA transfers. The Transfer Count determines the number of DMA transfers within one DMA transaction.

Example:

1024 word (32-bit wide) transactions can be composed of 256 transfers of four DMA word moves, or 128 transfers of eight DMA word moves.

Linked List

A linked list is a series of DMA transactions executed in the same DMA channel.

15.4.2 DMA Principles

The DMA controller supports DMA moves from one address location to another location.

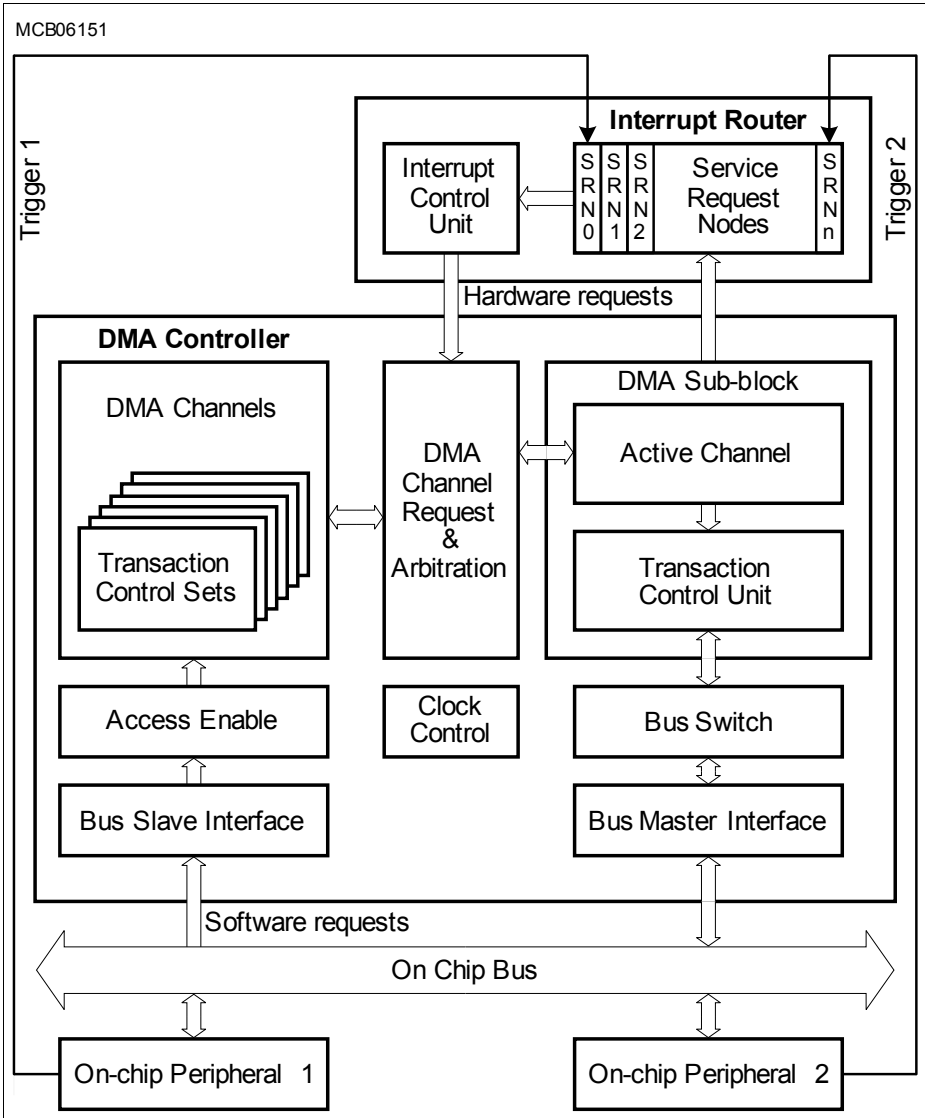


Figure 15-3 DMA System Overview

Direct Memory Access (DMA)

DMA moves can be requested either by hardware or by software. DMA hardware requests are triggered by specific request lines from the Interrupt Control Unit (ICU) or from other DMA channels (see [Figure 15-3](#)). Typically, the parallel occurrence of DMA requests and interrupts requests for DMA channels is possible. Therefore, the ICU and the DMA controller can react independently to interrupt DMA requests that have been generated by one source.

The DMA controller primarily consists of DMA channels, sub-blocks (move engines) and a bus switch. Once configured, the DMA sub-blocks are able to act as a master on the SPB Bus and on the SRI Bus.

15.4.3 DMA Channel Functionality

The transaction control set of each DMA channel is stored in the DMARAM in an array of 8 X 32-bit words. When a DMA request is serviced the associated transaction control set is copied from the DMARAM to the active DMA channel register set within a DMA sub-block. The transaction control set of a DMA channel includes functionality to support the calculation of Cyclic Redundancy Checker (CRC) checksums for source and destination addresses and read data to support enhanced data integrity checking. The index “z” refers to the channel number (z = 000 to a maximum of z = 047) within the DMA controller.

DMA channels are associated with a hardware resource partition. The index “y” refers to the hardware resource number (y = 0-3). The move engine and error registers located in the DMA sub-block use the index “x” to refer to the DMA sub-block number (x = 0-1).

When a DMA sub-block x services a DMA request from DMA channel z then the transaction control set (8 X 32-bit words) is copied from the DMARAM to the following active channel register set in DMA sub-block x:

- Channel x Status Register MExCHSR (for details, see [Page 15-108](#))
- Channel x Control Register MExCHCR (for details, see [Page 15-112](#))
- Channel x Address and Interrupt Control Register MExADICR (for details, see [Page 15-116](#))
- Channel x Shadow Address Register MExSHADR (for details, see [Page 15-111](#))
- Channel x Destination Address Register MExDADR (for details, see [Page 15-126](#))
- Channel x Source Address Register MExSADR (for details, see [Page 15-127](#))
- Channel x Source and Destination Address CRC Register MExSDCRC (for details, see [Page 15-128](#))
- Channel x Read Data CRC Register MExRDCRC (for details, see [Page 15-129](#))

15.4.3.1 Shadowed Source or Destination Address

As a typical application, an ASC module that receives data (fixed source address) has to deliver it to a memory buffer using a DMA transaction (variable destination address). After a certain amount of data has been transferred, a new DMA transaction should be

Direct Memory Access (DMA)

initiated to deliver further ASC data into another memory buffer. While the destination address register is updated during a running DMA transaction with the actual destination address, a shadow mechanism allows programming of a new destination address without disturbing the content of the destination address register. In this case, the new destination address is written into a buffer register, i.e. the shadow address register. At the start of the next DMA transaction, the new address is transferred from this shadow address register to the destination address register without CPU intervention. This shadow mechanism avoids the CPU having to check for the end of a DMA transaction before reprogramming address registers.

The shadow address register can be used also to store a source address. However, it cannot store source and destination address at the same time. This means that the shadow mechanism makes it possible to automatically update either a new source address, or a new destination address at the start of a DMA transaction. If both address registers (for source and destination address) have to be updated for the next DMA transaction, a running DMA transaction for this channel must be finished. After that, source and destination address registers can be written before the next DMA transaction is started.

Only one address register can be shadowed while a transaction is running, because the shadow register can only be assigned either to the source or to the destination address. Note that the shadow address transfer mechanism has the same behavior in Single and Continuous Mode.

When the shadow mechanism is disabled the DMA sub-block active channel shadow address register MExSHADR is always read as 0000 0000_H.

DMA Channel Idle

If no DMA transaction is running and a new source or destination address is written to DMA channel z then the new address value is directly written into the DMARAM source address word (SADRz) or destination address word (DADRz). Writes to the shadow address word (SHADRz) are as follows:

- Read Only Mode: the shadow address is not writable (ADICRz.SHCT = 0001_B or 0010_B) and a write to the shadow address word SHADRz will result in a bus error.
- Direct Write Mode: the shadow address is directly writable (ADICRz.SHCT = 0101_B or 0110_B) and the shadow address word SHADRz is updated.

DMA Channel Active

In the case when move engine x is servicing a DMA request from DMA channel z and a DMA transfer is in progress then address updates are dependent on the shadow control settings:

- Read Only Mode: [Figure 15-4](#) shows the actions that take place when a source address register is updated (destination register update happens in an equivalent

Direct Memory Access (DMA)

manner). The shadow address is not directly writable ($ADICRz.SHCT = 0001_B$ or 0010_B).

- For source shadow addressing ($ADICRz.SHCT = 0001_B$) the new source address is written to the shadow address register $MExSHADR$ in DMA sub-block x .
- For destination shadow addressing ($ADICRz.SHCT = 0010_B$) the new destination address is written to the shadow address register $MExSHADR$ in DMA sub-block x .
- Direct writes to the shadow address register will result in a bus error.
- When the current DMA transaction completes and the final write back to DMARAM occurs the $MExSHADR$ value is written to either $SADRz$ or $DADRz$ and the $SHADRz$ is set to 00000000_H .
- Direct Write Mode: the shadow address is directly writable ($ADICRz.SHCT = 0101_B$ or 0110_B) and the shadow address register $MExSHADR$ in DMA sub-block x is directly written.
 - The $SHADRz$ value stored in the DMARAM is updated to the $MExSHADR$ value on the write back.
 - At the start of a new DMA transaction the source address register ($MExSADR$) or the destination address register ($MExDADR$) is updated with the $SHADRz$ value. $MExADICR.SHCT$ must be set accordingly.
 - The shadow transfer will be repeated until DMA channel z is reset or until the value in $MExSHADR$ is $0000\ 0000_H$.

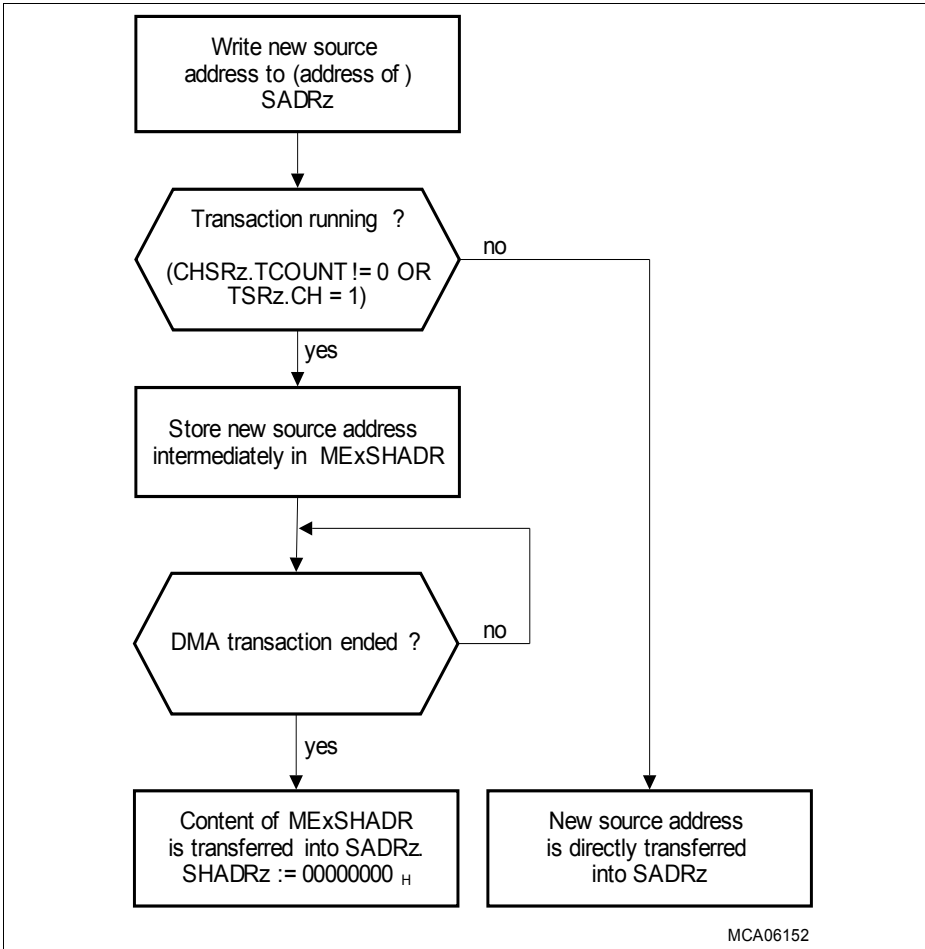


Figure 15-4 Source Address Update (ADICRz.SHCT = 0001_B)

Transfer Count Update

The transfer count of a DMA transaction, stored in bit field CHCFGRz.TREL, can also be programmed if the DMA transaction is running. At the start of a DMA transaction, CHCFGRz.TREL is transferred to bit field MExCHSR.TCOUNT. TCOUNT is updated during the DMA transaction.

No reload of address or counter will be done if MExCHSR.TCOUNT is not equal to 0.

Direct Memory Access (DMA)

The reprogramming of channel specific values (except for the selected shadow address register) must be avoided while a DMA channel is active as the data transfer maybe corrupted.

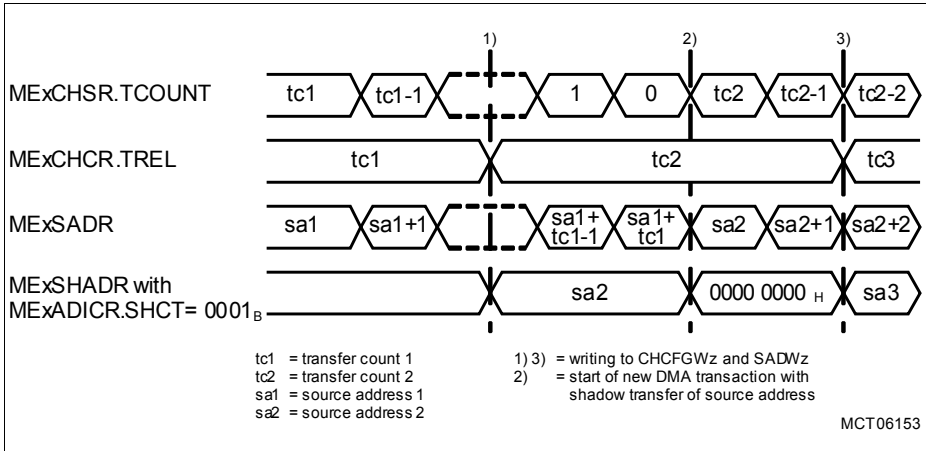


Figure 15-5 Shadow Source Address and Transfer Count Update with MExADICR.SHCT = 0001_B (Shadow Read Only Mode)

Figure 15-5 shows how the contents of the source address register MExSADR and the transfer count MExCHSR.TCOUNT are updated during two DMA transactions with a shadowed source address and transfer count update.

At reference point 2) the DMA transaction 1 is finished and DMA transaction 2 is started. At 1) the DMA channel is reprogrammed with two new parameters for the next DMA transaction: Transfer count $tc2$ and source address $sa2$. Source address $sa2$ is buffered in MExSHADR and transferred to MExSADR when the new DMA transaction is started at 2). At this time, transfer count $tc2$ is also transferred to MExCHSR.TCOUNT. Note that the shadow address register is only reset by hardware to $0000\ 0000_H$ as shown in this example, if the write enable bit is set to 0 (ADICRz.SHCT = 0001_B or 0010_B).

In the event that CHCFGz.TREL is written while DMA channel z is active:

- A write to CHCFGz.TREL will update the MExCHCR.TREL value in the DMA sub-block x.
- On write back CHCFGz.TREL is updated with the latest MExCHCR.TREL value.
- MExCHSR.TCOUNT is updated to the new CHCFGz.TREL value at the start of the next DMA transaction.

15.4.3.2 DMA Channel Request Control

Figure 15-6 shows the control logic for DMA requests that is implemented for each DMA channel.

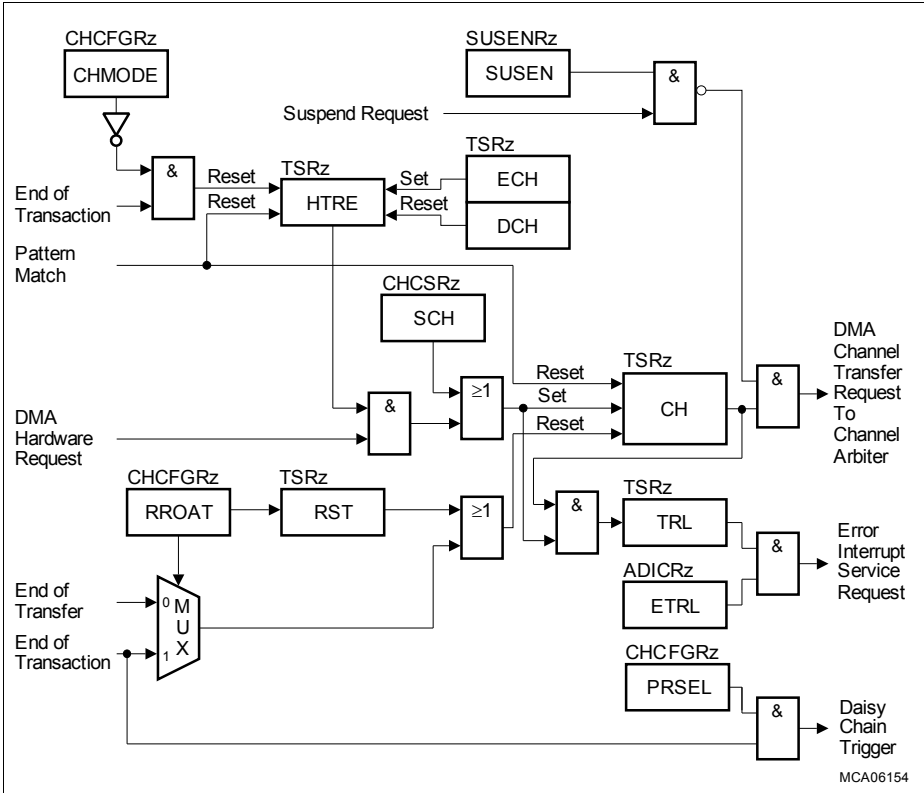


Figure 15-6 Channel Request Control

Two different types of DMA requests are possible:

- Hardware DMA requests
- Software DMA requests

DMA hardware requests (CHz_REQ) are triggered by the Interrupt Control Unit (ICU).

If CHCFGRz.PRSEL = 1 in the current DMA channel z can bypass the ICU and trigger a DMA hardware request in the next lower DMA channel z-1. The latency to service a DMA channel z-1 request is reduced. DMA channel z interrupt service requests are disabled.

Direct Memory Access (DMA)

Hardware requests are enabled/disabled by status bit TSRz.HTRE. HTRE can be set/reset by software or by hardware in Single Mode at the end of a DMA transaction. A software request can be generated by setting bit CHCSRz.SCH.

Status flag TSRz.CH indicates whether or not a software or hardware generated DMA request for DMA channel z is pending. TSRz.CH is cleared when the DMA transfer starts (RROAT = 0) or at the end of a DMA transaction (RROAT = 1).

If a software or a hardware DMA request is detected for channel z while TSRz.CH is set, a request lost event occurs. This error event indicates that the DMA is already processing a transfer and that another transfer has been requested before the end of the previous one. In this case, bit TSRz.TRL will be set. If ADICRz.ETRL is set then a transfer lost interrupt trigger will be generated when DMA channel z becomes active.

15.4.3.3 DMA Channel Operation Modes

The operation mode of a DMA channel is individually programmable for each DMA channel z. In basic terms, a DMA channel can operate in the following modes:

- Software controlled mode
- Hardware controlled mode, in Single Mode, Continuous Mode or Linked List.

In software-controlled mode, a DMA channel request is generated by setting a control bit. In hardware-controlled mode, a DMA channel request is generated by the ICU servicing an interrupt trigger generated by an on chip peripheral unit.

In hardware-controlled Single Mode, a DMA channel z becomes disabled by hardware after the last DMA transfer of its DMA transaction. In hardware-controlled Continuous Mode, a DMA channel z remains enabled after the last DMA transfer of its DMA transaction.

In linked list mode the DMA channel is loaded with the new transaction control set on completion of the current DMA transaction. The next DMA transaction can optionally be auto started by a software request.

In hardware- and software-controlled mode, a DMA request signal can be configured to trigger a complete DMA transaction or one single transfer.

Software-controlled Modes

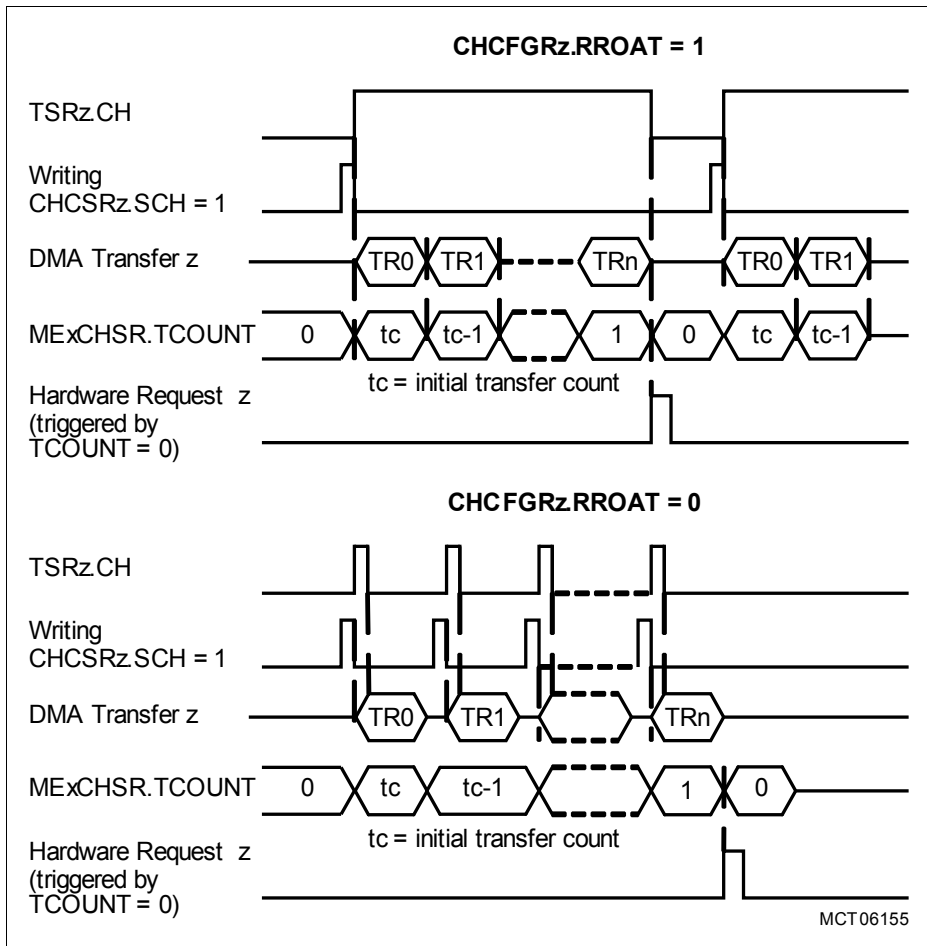
In software-controlled mode, one software request starts one complete DMA transaction or one single DMA transfer. Software-controlled modes are selected by writing TSRz.DCH = 1. This forces status flag TSRz.HTRE = 0 (hardware request of DMA channel z is disabled).

The software-controlled mode that initiates one complete DMA transaction to be executed is selected for DMA channel z by the following write operations:

- CHCFGRz.RROAT = 1
- CHCSRz.SCH = 1

Direct Memory Access (DMA)

Setting CHCSRz.SCH to 1 (this is the software request) causes the DMA transaction of DMA channel z to be started and TSRz.CH to be set. At the start of the DMA transaction, the value of CHCFGRz.TREL is loaded into MExCHSR.TCOUNT (transfer count or tc) and the DMA transfers are executed. After each DMA transfer, TCOUNT becomes decremented and next source and destination addresses are calculated. When TCOUNT reaches the 0, DMA channel z becomes disabled and status flag TSRz.CH is reset. Setting CHCSRz.SCH again starts a new DMA transaction of DMA channel z with the parameters as currently defined in the channel register set.


Figure 15-7 Software Controlled Mode Operation

Direct Memory Access (DMA)

The software-controlled mode that initiates a single DMA transfer to be executed is selected for DMA channel z by the following write operations:

- CHCFGRz.RROAT = 0
- CHCSRz.SCH = 1, repeated for each DMA transfer

When CHCFGRz.RROAT = 0, TSRz.CH is cleared when a DMA transfer starts and a new software request (writing CHCSRz.SCH = 1) must be generated for starting the next DMA transfer.

Hardware-controlled Modes

In hardware-controlled modes, a hardware request signal starts a DMA transaction or a single DMA transfer. There are two hardware-controlled modes available:

- **Single Mode:**
Hardware requests are disabled by hardware after a DMA transaction
- **Continuous Mode:**
Hardware requests are not disabled by hardware after a DMA transaction

Hardware-controlled Single Mode

In hardware-controlled Single Modes, one hardware request starts one complete DMA transaction or one single DMA transfer. The hardware-controlled Single Mode that initiates one complete DMA transaction to be executed for DMA channel z is selected by the following operations:

- CHCFGRz.CHMODE = 0
- CHCFGRz.RROAT = 1
- CHCFGRz.PRSEL = 0
- TSRz.ECH = 1

Setting TSRz.ECH to 1 enables hardware transfer requests (TSRz.HTRE = 1) for DMA channel z. When the ICU generates a DMA hardware request CHz_REQ then TSRz.CH is set high. If DMA channel z wins channel arbitration then DMA channel z becomes a move engine active channel. The value of CHCFGRz.TREL is loaded into MEXCHSR.TCOUNT and the DMA transaction is started by executing its first DMA transfer. After each DMA transfer, TCOUNT is decremented and next source and destination addresses are calculated. When TCOUNT reaches the 0, DMA channel z becomes disabled and status flags TSRz.CH and TSRz.HTRE are reset. In order to start a new hardware-controlled DMA transaction, hardware requests must be enabled again by setting TSRz.HTRE through TSRz.ECH = 1. The hardware request disable function in Single Mode is typically needed when a reprogramming of the DMA channel register set (addresses, transfer count) is required before the next hardware triggered DMA transaction is started.

Direct Memory Access (DMA)

The hardware-controlled Single Mode in which each single DMA transfer has to be requested by a hardware request signal is selected as described above, with one difference:

- $CHCFGRz.RROAT = 0$

In this operation mode, $TSRz.CH$ is cleared when a new DMA transfer starts, and a new hardware request at CHz_REQ must be generated for starting the next DMA transfer.

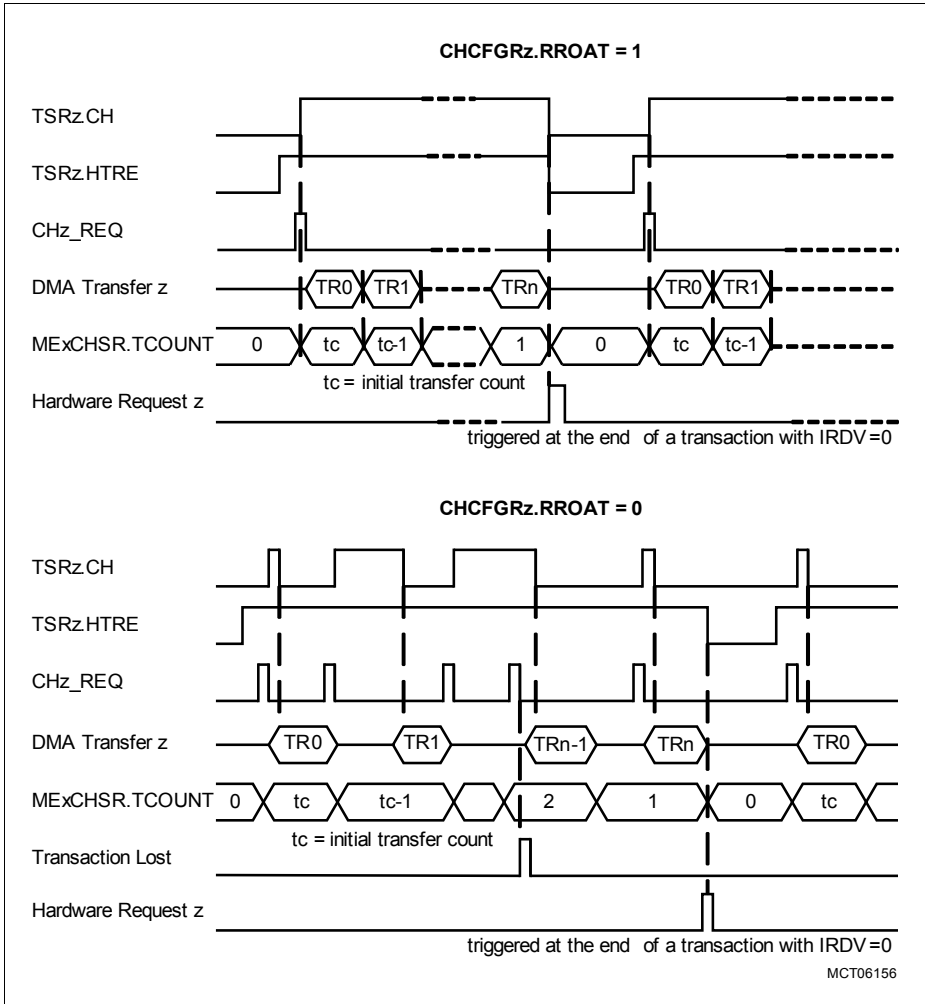


Figure 15-8 Hardware-controlled Single Mode Operation

Direct Memory Access (DMA)

Hardware-controlled Continuous Mode

In hardware-controlled Continuous Mode (CHCFGRz.CHMODE = 1), the hardware transaction request enable bit TSRz.HTRE is not reset at the end of a DMA transaction. A new transaction of DMA channel z with the parameters actually stored in the transaction control set of DMA channel z is started each time when CHCSRz.TCOUNT = 0 at the end of the DMA transaction. No software re-enable for a hardware request at CHz_REQ is required.

Combined Software/Hardware-controlled Mode

Figure 15-9 shows how software- and hardware-controlled modes can be combined. In the example, the first DMA transfer is triggered by software when setting CHCSRz.SCH. Hardware requests are still disabled. After hardware requests have been enabled by setting TSRz.ECH, subsequent DMA transfers are triggered now by hardware request coming from the CHz_REQ line.

In the example, DMA channel z operates in Single Mode (CHCFGRz.CHMODE = 0). In this mode, TSRz.HTRE becomes reset by hardware when CHCSRz.TCOUNT = 0 at the end of the DMA transaction.

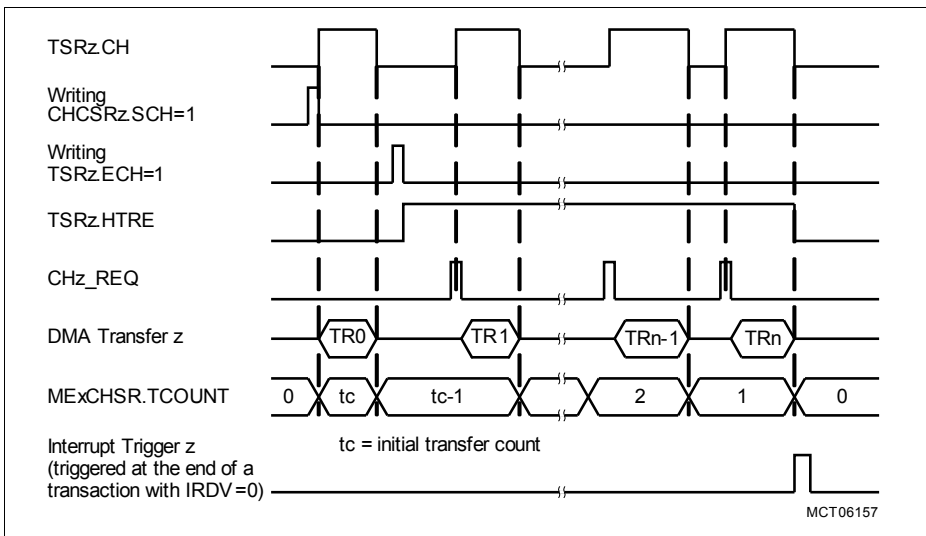


Figure 15-9 Transaction Start by Software, Continuation by Hardware

If a DMA channel is configured to be triggered by parallel hardware and software requests then if the requests collide in the same clock cycle the DMA move will be executed and a Transaction/Transfer Request Lost event will be flagged in the TSRz.TRL bit.

15.4.3.4 DMA Service Requests

Interrupt Requests are prioritized by the Interrupt Router and processed by one of the Service Providers (CPU or DMA). Each DMA peripheral interfaces to a dedicated Interrupt Control Unit (ICU) instantiated in the Interrupt Router (IR).

DMA channels are associated with the Service Request Priority Number bit field programmed in the Service Request Control Register SRC.SRPN. For example:

- DMA Channel 000 equates to SRC.SRPN = 0 programmed in IR.
- DMA Channel 001 equates to SRC.SRPN = 1 programmed in IR.
- DMA Channel 002 equates to SRC.SRPN = 2 programmed in IR.
- DMA Channel 003 equates to SRC.SRPN = 3 programmed in IR.
- DMA Channel 004 equates to SRC.SRPN = 4 programmed in IR.

The routing of a hardware Service Request to a service provider destination is determined by the IR Type of Service Control bit field SRC.TOS. The DMA will acknowledge all Service Requests. If the value programmed in the SRC.SRPN is for an Invalid Channel then the DMA Controller will take no action.

The user must programme valid SRC.SRPN values for a DMA peripheral.

Daisy Chain

DMA Channels can be daisy chained by setting the bit CHCFGRz.PRSEL.

When a higher priority DMA channel completes a DMA transaction then it will initiate a DMA transaction on the next lower priority DMA channel by setting the access pending bit TSRz.CH bit. Daisy chaining is limited to a higher priority DMA channel initiating the next lower priority DMA Channel.

Enabling the daisy chain disables the channel transfer interrupt trigger in the next higher priority channel. In a typical DMA daisy chain application only the lowest priority DMA channel is required to generate a channel transfer interrupt trigger. When the sequence of DMA transactions from the highest to lowest priority DMA channel has completed then the lowest priority DMA channel in the daisy chain will generate a channel transfer interrupt trigger to signal the end of the DMA operation.

Only the Service Request for the highest priority DMA Channel in the daisy chain is initiated by the ICU. The lower priority Service Requests bypass the ICU in order to improve interrupt latency.

15.4.3.5 Channel Reset Operation

A DMA transaction of DMA channel z can be stopped (channel is reset) by setting bit TSRz.RST. If a read or write on chip bus access of DMA channel z is executed at the time when TSRz.RST is set then the On Chip Bus access is finished normally. This behavior guarantees data consistency.

When a channel reset is applied to an active DMA channel, the on-going DMA transfer executing in the move engine will complete before the DMA channel transitions to the reset state. A pending DMA channel must become active and service its pending request before making a transition to the channel reset state.

DMA Channel Reset State

On completion of a DMA channel reset:

- Bits TSRz.HTRE, TSRz.CH, TSRz.TRL, CHCSRz.ICH, CHCSRz.IPM, CHCSRz.WRPD, CHCSRz.WRPS, CHCSRz.LXO, and bit field CHCSRz.TCOUNT are reset.
- If a circular buffer ADICRz.SCBE and/or ADICRz.DCBE is enabled then the source and/or destination address register will be set to the wrap boundary else the address registers are cleared. SHADRz will be cleared.
- All automatic functions are stopped for channel z.

Resetting a DMA Channel

A user program must execute the following steps to reset a DMA channel:

1. If hardware requests are enabled for the DMA channel z, disable the DMA channel z hardware requests by setting TSRz.DCH = 1.
2. Writing a 1 to TSRz.RST.
3. Waiting (polling) until TSRz.RST = 0 to indicate the reset has completed.

During a DMA channel reset operation, software requests must not be initiated by setting CHCSRz.SCH = 1.

Restarting a DMA Channel

A user program must execute the following steps for restarting a DMA channel after it was reset:

1. Optionally (re-)configuring the address and other channel registers.
2. Restarting the DMA channel z by setting TSRz.ECH = 1 for hardware requests or CHCSRz.SCH = 1 for software requests.

15.4.3.6 Channel Halt Operation

A DMA channel can be halted during a DMA transaction and the state frozen to allow a background RAM test to be run over a destination memory in order to detect stuck bits and distinguish between static errors and transient errors. On completion of the RAM test the DMA channel can be re-started and the DMA transaction completed.

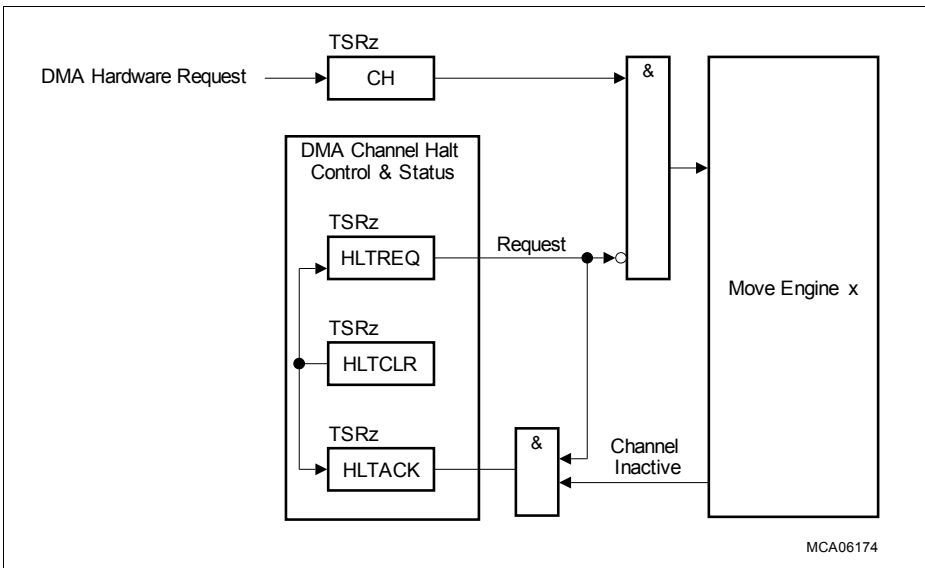


Figure 15-10 DMA Channel Halt Control

The halt control utilizes a set/clear mechanism to request the channel transitions to and from the HALT state on completion of the current DMA transfer. Only writing a logic '1' to set or clear a halt request to a DMA channel has an effect and the status of other channels can be ignored.

The DMA channel halt operation is shown in [Figure 15-11](#).

Entering DMA Channel Halt

Channel Halt Mode of DMA channel z is entered if the halt request bit `TSRz.HLTREQ` is set by software. The DMA channel enters the halt state mode automatically after the current DMA transfer is completed. The DMA channel z halt state is signalled by the halt acknowledge status flag `TSRz.HLTACK`.

When the DMA channel is in the halt state the transaction control set can be modified. These modifications are taken into account for further DMA transactions or DMA transfers of the related DMA channel after Halt Mode has been left again.

Direct Memory Access (DMA)

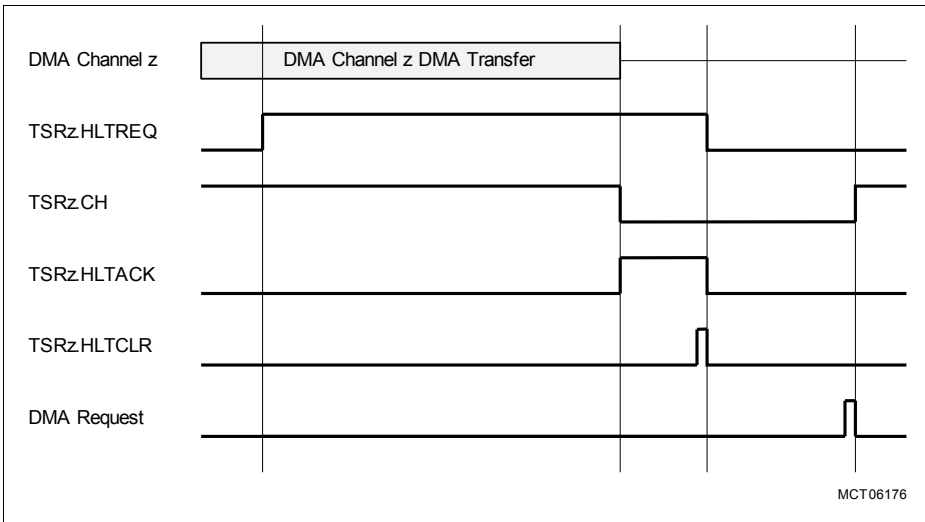


Figure 15-11 DMA Channel Halt Operation

Exiting Channel Halt Mode

DMA channel z is released from the halt state by software setting the halt clear bit TSRz.HLTCLR. Normal DMA operation is resumed.

DMA Hardware Request during Channel Halt Mode

If DMA channel z is in the halt state (TSRz.HLTACK = 1_B) and the hardware transaction request enable bit TSRz.HTRE is set then the DMA channel z will respond to DMA hardware request service requests as follows:

- Idle channel (TSRz.CH = 0_B): the access pending bit will be set and the DMA hardware request will be serviced when DMA channel z exits halt mode.
- Active channel (TSRz.CH = 1_B): the TSRz.TRL bit is set as the transaction will be lost. If the enable transaction request lost interrupt bit ADICRz.ETRL is set then an interrupt will be generated for a transaction request lost event.

Direct Memory Access (DMA)

15.4.3.7 Transfer Count and Move Count

The move count determines the number of moves (consisting of one read and one write each) to be done in each DMA transfer. It allows the user to indicate to the DMA the number of moves to be done after one request. The number of DMA moves per DMA transfer is selected by the block mode settings (CHCFGRz.BLKM).

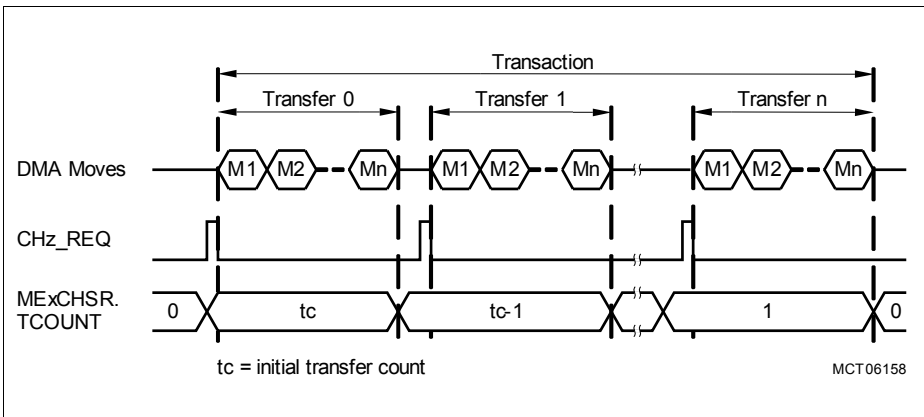


Figure 15-12 Transfer and Move Count

After a DMA move, the next source and destination addresses are calculated. Source and destination addresses are calculated independently of each other. The following address calculation parameters can be selected:

- The address offset, which is a multiple of the selected data width
- The offset direction: addition, subtraction, or none (unchanged address)

Control bits in address and interrupt control word ADICRz determine how the addresses are incremented/decremented. Further, the data width as defined in CHCFGRz.CHDW is taken into account for the address calculation.

Figure 15-13 and Figure 15-14 show two examples of address calculation. In both examples, a data width of 16-bit (CHCFGRz.CHDW = 001_B) is assumed.

Direct Memory Access (DMA)

Programmable Address Modification - Example 1

In **Figure 15-13**, 16-bit half-words are transferred from a source memory with an incrementing source address offset of 10_H to a destination memory with decrementing destination addresses offset of 08_H.

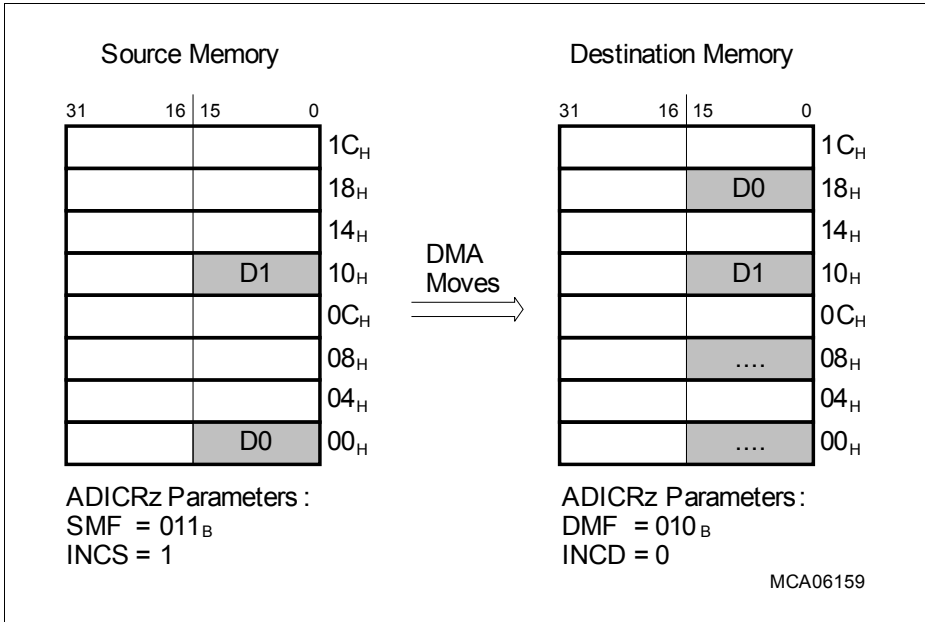


Figure 15-13 Programmable Address Modification - Example 1

Direct Memory Access (DMA)

Programmable Address Modification - Example 2

In **Figure 15-14**, 16-bit half-words are transferred from a source memory with an incrementing source address offset of 02_H to a destination memory with incrementing destination addresses offset of 04_H .

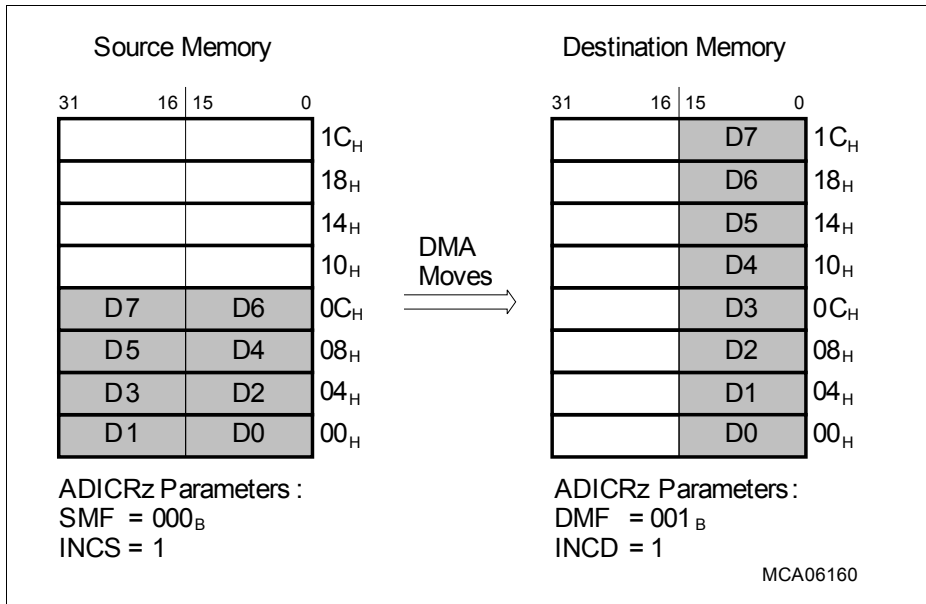


Figure 15-14 Programmable Address Modification - Example 2

15.4.3.8 Circular Buffer

Destination and source address can be configured to build a circular buffer separately for source and destination data. Within this circular buffer, addresses are updated as defined in **Figure 15-13** and **Figure 15-14** with a wrap-around at the buffer limits. The circular buffer length is determined by bit fields ADICRz.CBLS (for the source buffer) and ADICRz.CBLD (for the destination buffer). These 4-bit wide bit fields determine which bits of the 32-bit address remain unchanged at an address update. Possible buffer sizes of the circular buffers can be 2^{CBLS} or 2^{CBLD} bytes (= 1, 2, 4, 8, 16, ... up to 64k bytes).

Source and destination addresses are updated (incremented or decremented) during a DMA move, all upper bits [31:CBLS] of source address and [31:CBLD] of destination address are frozen and remain unchanged, even if a wrap-around from the lower address bits [CBLS-1:0] or [CBLD-1:0] occurred. This address-freezing mechanism always causes the circular buffers to be aligned to a multiple integer value of its size.

Direct Memory Access (DMA)

If the circular buffer size is less or equal than the selected address offset (see **Figure 15-13**), the same circular buffer address will always be accessed.

The source and destination circular buffers are enabled by setting the circular buffer enable bits ADICRz.SCBE and ADICRz.DCBE respectively.

15.4.3.9 Address Counter

If ADICRz.SCBE = 0 then the source circular buffer is not enabled and the address will increment/decrement determined by ADICRz.INCS across the entire 32-bit address field. The address offset is determined by CHCFGRz.CHDW. The source address will wrap around on the 32-bit address boundary:

- If incrementing will wrap from FFFFFFFF_H to 00000000_H.
- If decrementing will wrap from 00000000_H to FFFFFFFF_H.

15.4.3.10 Flow Control

A timestamp can optionally be attached at the end a DMA transaction to record the occurrence of the event. The timestamp is the time at which the event was recorded by the DMA controller and not the real time.

Timestamp Generation

The timestamp is generated from a 32-bit binary upwards synchronous counter clocked by the SPB clock divided by 8 as shown in **Figure 15-15**. The counting starts automatically after an application reset. It is not possible to affect the counter during normal operation.

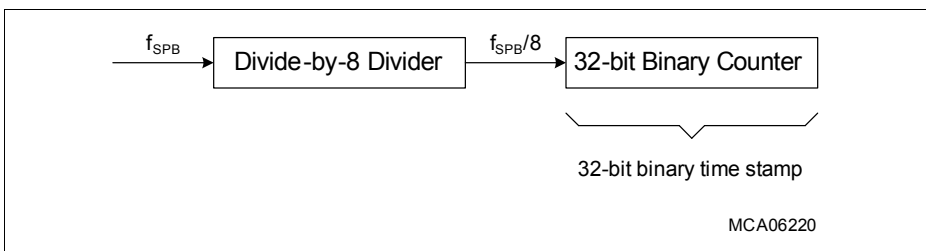


Figure 15-15 Timestamp Generation

The current 32-bit count value can be read through the DMA_TIME register.

Timestamp Appendage

The appendage of a timestamp to the destination data at the end of a DMA transaction is controlled by the ADICRz.STAMP bit. If enabled then a 32-bit timestamp will be appended by a DMA write move to the next word aligned address in the destination data

Direct Memory Access (DMA)

sequence. If the address control increment of destination address bit $ADICRz.INCD = 1_B$ then the timestamp will be written at the next higher word aligned destination address and if $ADICRz.INCD = 0_B$ then the timestamp will be written at the next lower word aligned destination address.

If a DMA channel is configured to support circular buffer operation then the timestamp must be appended at the next word aligned address above the circular buffer ($ADICRz.INCD = 1_B$) or below the circular buffer ($ADICRz.INCD = 0_B$). The appendage of the timestamp must not overwrite or change the destination address of circular buffer DMA move data.

Direct Memory Access (DMA)

Appendage of Timestamp - Example 1

In **Figure 15-16**, 16-bit half-words are transferred from a source memory with an incrementing source address offset of 10_H to a destination memory with decrementing destination addresses offset of 08_H .

The destination address is incrementing ($ADICRz.INCD = 1_B$) and the timestamp is appended above the destination data.

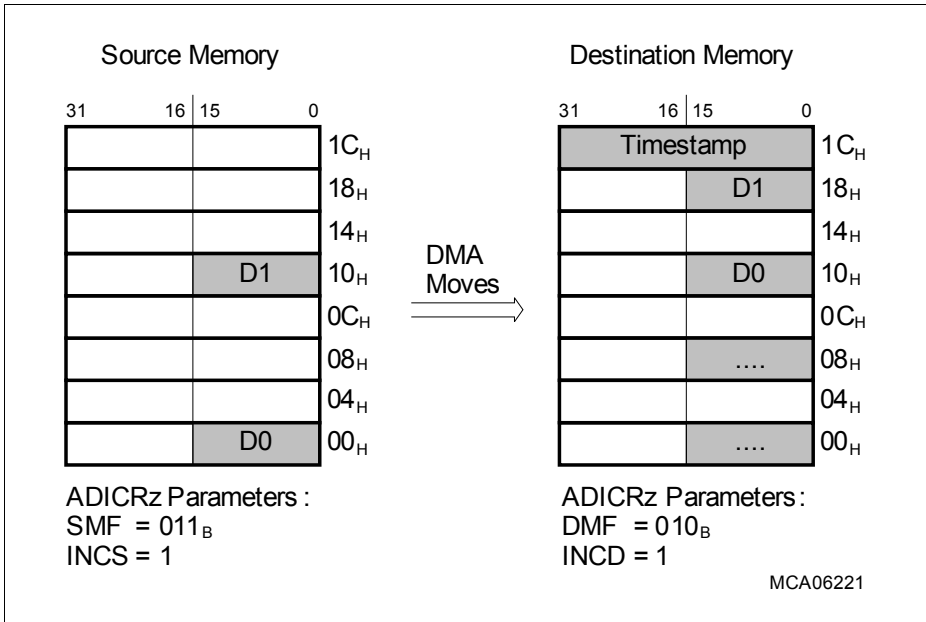


Figure 15-16 Appendage of Timestamp - Example 1 ($ADICRz.INCD = 1$)

Appendage of Timestamp - Example 2

In **Figure 15-17**, 16-bit half-words are transferred from a source memory with an incrementing source address offset of 02_H to a destination memory with decrementing destination addresses offset of 04_H .

The destination address is decrementing ($ADICRz.INCD = 0$) and the timestamp is appended below the destination data.

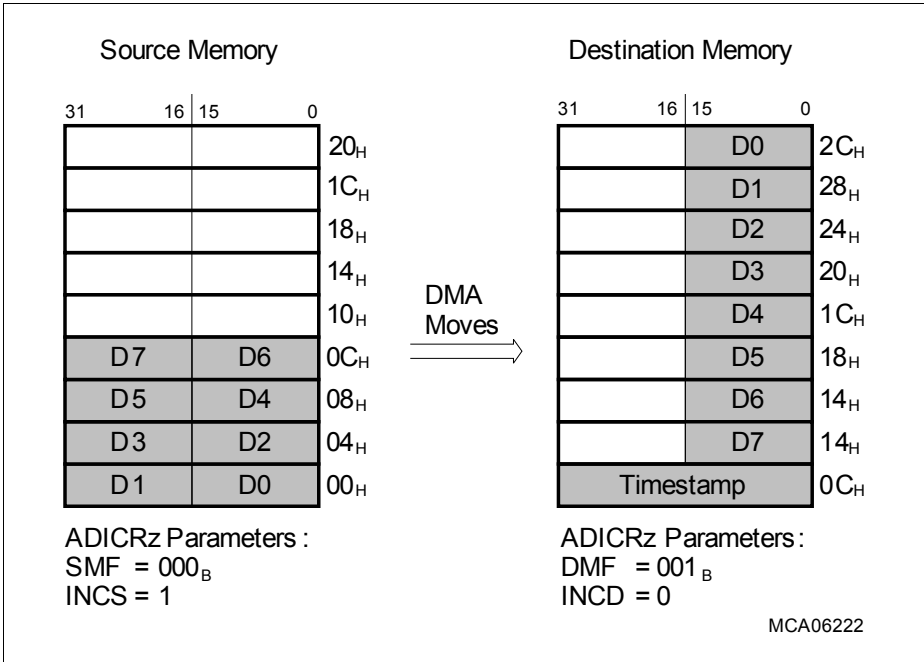


Figure 15-17 Appendage of Timestamp - Example 2 (ADICRz.INCD = 0)

Flow Control

The appendage of timestamps to the end of DMA transactions support flow control. The relative count value of the timestamp can be used to determine the sequence of DMA writes to different destination addresses.

15.4.3.11 Double Buffering Operation

The DMA supports double buffering [Figure 15-18](#) shows an example of double destination buffering. DMA read moves transfer a continuous data stream from a peripheral to the DMA controller and DMA write moves transfer the read data from the DMA controller to one of two destination buffers stored in memory. The data stream can be re-directed via a software switch. The dormant data buffer can be frozen and be available for cyclic software tasks while the other data buffer continues to be filled.

Direct Memory Access (DMA)

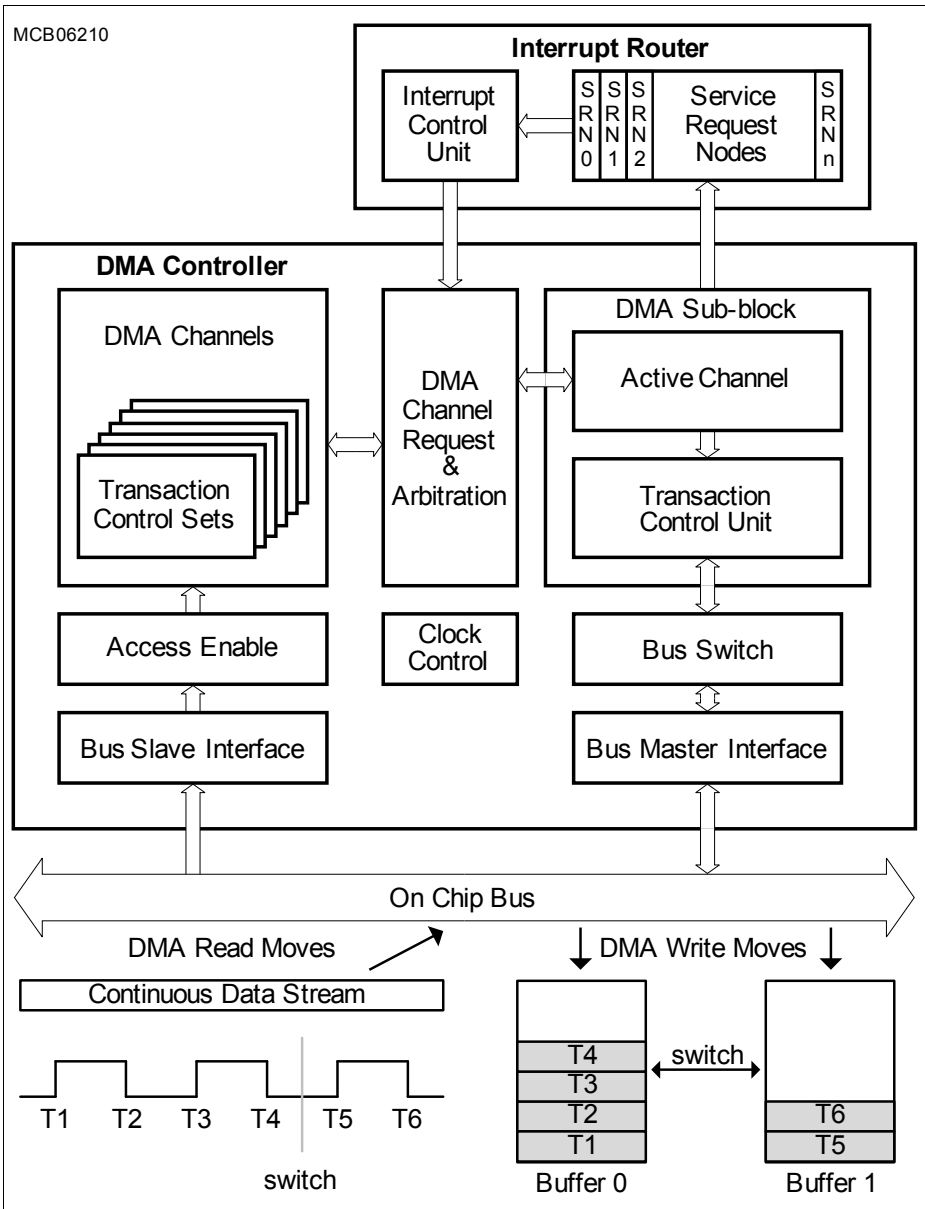


Figure 15-18 DMA Double Buffering

Direct Memory Access (DMA)**Double Source Addressing**

A DMA Channel can be configured to support double source addresses (ADICRz.SHCT[3:1] = 100_B).

Data is streamed by the DMA from one of two data buffers (buffer 0 and buffer 1) stored in memory to a continuous output data stream. Double buffer source addresses:

- The source address SADRz is used to address buffer 0.
- The shadow address SHADRz is used as a source address to address buffer 1.

The size of the two source double buffers is equal and determined by the following DMA transaction control parameters:

- The channel data width (CHCFGRz.CHDW).
- The source circular buffer enable/disable control (ADICRz.SCBE).
- The source address modification factor (ADICRz.SMF).
- The increment of the source address (ADICRz.INCS).
- The block mode value (CHCFGRz.BLKM).
- The transfer reload value (CHCFGRz.TREL).

The size of one source buffer is equal to the size of one DMA transaction.

Double Destination Addressing

A DMA Channel can be configured to support double destination addresses (ADICRz.SHCT[3:1] = 101_B).

A continuous input data stream is directed by the DMA to one of two data buffers (buffer 0 and buffer 1) stored in memory. Double buffer destination addresses:

- The destination address DADRz is used to address buffer 0.
- The shadow address SHADRz is used as a destination address to address buffer 1.

The size of the two destination double buffers is equal and determined by the following DMA transaction parameters:

- The channel data width (CHCFGRz.CHDW).
- The destination circular buffer enable/disable control (ADICRz.DCBE).
- The destination address modification factor (ADICRz.DMF).
- The increment of the destination address (ADICRz.INCD).
- The block mode value (CHCFGRz.BLKM).
- The transfer reload value (CHCFGRz.TREL).

The size of one destination buffer is equal to the size of one DMA transaction.

Active Buffer

During DMA double buffering the CHCSRz.BUFFER status bit is used to indicate which buffer is active.

- MExCHSR.BUFFER = 0_B: buffer 0 is read or filled by the DMA.

Direct Memory Access (DMA)

- MExCHSR.BUFFER = 1_B: buffer 1 is read or filled by the DMA.

Buffer Switching

The re-direction of the data stream from one buffer to the other can be controlled by software and additionally automatically by hardware. Automatic hardware switching is selected by setting ADICRz.SHCT[0] = 1_B. The different DMA double buffering modes are shown in [Table 15-1](#).

Table 15-1 DMA Double Buffering Modes

SHCT	Description
1000 _B	DMA Double Source Buffering Software Switch Only Switching from the current source buffer to the other source buffer is controlled by the software switch CHCSRz.SWB
1001 _B	DMA Double Source Buffering Automatic Hardware and Software Switch DMA controller automatically switches source buffers when current source buffer is emptied. Software source buffer switching controlled by CHCSRz.SWB
1010 _B	DMA Double Destination Buffering Software Switch Only Switching from the current destination buffer to the other destination buffer is controlled by the software switch CHCSRz.SWB
1011 _B	DMA Double Destination Buffering Automatic Hardware and Software Switch DMA controller automatically switches destination buffers when current destination buffer is full. Software destination buffer switching controlled by CHCSRz.SWB

Software Buffer Switch

Before a DMA transaction completes and the active buffer is full software can re-direct the data stream from the active buffer to the other buffer by setting CHCSRz.SWB. If a move engine is actively servicing a DMA channel z configured for double buffering then the current DMA transfer completes before the buffer switch is made.

On completion of the DMA transfer the DMA controller will:

- Automatically re-load MExCHSR.TCOUNT with CHCFGRz.TREL
- Switch the address pointer between the source/destination address and the shadow
 - Buffer 0 address pointer (source or destination address).
 - Buffer 1 address pointer (shadow address).

Direct Memory Access (DMA)

The address control factors remain the same. The next DMA transaction controlling the filling or reading of the new buffer will start when a DMA request is received.

During a software buffer switch no DMA requests are lost by the DMA controller and there is no loss, duplication or split of data across the two buffers.

Buffer Empty or Full

If the current buffer is emptied (in the case of DMA double source buffering) or filled (in the case of DMA double destination buffering) before a software buffer switch is received then the DMA channel will stop and no more DMA transfers are made. The transaction control set is frozen. The transfer count status (MExCHSR.TCOUNT and CHCSRz.TCOUNT) when the DMA channel is stopped is equal to 0000_H.

The signalling of a buffer empty or full event can be generated by using the DMA channel traffic interrupt service request. The interrupt control ADICRz.INTCT is set to 10_B and the interrupt raise detect value ADICRz.IRDV set to 0000_B. When the transfer count equals zero then the DMA channel will raise a DMA channel interrupt service request.

The DMA channel interrupt handler can interrogate the DMA channel transaction control set to identify which empty or full buffer generated the interrupt. Software can reset the channel ready to restart DMA double buffering operations.

Fast Data Rate

If the data rate is faster than the DMA controller is able to transfer the data to one of the buffers then a transaction request lost event will occur.

If the DMA controller is servicing a DMA access pending request (TSRz.CH = 1_B) and a DMA hardware request is detected then the transaction request lost bit TSRz.TRL will be set. If the enable transaction request lost bit is set (ADICRz.ETRL = 1_B) then a DMA error interrupt service request will be generated.

The DMA error interrupt handler can interrogate the DMA channels and identify the source of the error. Software can reset the DMA channel and restart DMA double buffer operations.

Automatic Hardware Buffer Switch

The DMA controller will automatically switch from the active buffer to the other buffer when the DMA transaction reading or filling the active buffer completes. On switching buffers the MExCHSR.FROZEN bit is set to indicate that one buffer is frozen and available for cyclic software tasks. The interrupt control bits ADICRz.INTCT and interrupt raise detect value MExADICR.IRDV can be used to generate a DMA channel interrupt service request.

On completion of the DMA transaction when MExCHSR.TCOUNT equals 0000_H the DMA controller will:

Direct Memory Access (DMA)

- Automatically re-load MExCHSR.TCOUNT with CHCFGRz.TREL
- Switch the address pointer between the source/destination address and the shadow address:
 - Buffer 0 address pointer (source or destination address).
 - Buffer 1 address pointer (shadow address).

The address control factors remain the same. The next DMA transaction controlling the filling or reading of the new buffer will start when a DMA request is received.

During an automatic hardware buffer switch no DMA requests are lost by the DMA controller and there is no loss, duplication or split of data across the two buffers.

Frozen Buffer

When the DMA controller switches buffer the frozen bit MExCHSR.FROZEN is set. The frozen buffer is then available for a cyclic software task. On completion of the software task the status bit MExCHSR.FROZEN is cleared by software and the buffer address pointer is re-programmed to the start address ready for the next DMA transaction that reads/fills the buffer.

Transaction Request Lost

If MExCHSR.FROZEN is equal to 1_B then the automatic hardware switch buffer will not occur and the DMA controller will stop. If a DMA hardware request is detected before MExCHSR.FROZEN is cleared then the transaction request lost bit TSRz.TRL is set and a DMA error interrupt service request will be generated if ADICRz.ETRL is 1_B.

Direct Memory Access (DMA)**DMA Double Buffer Operation**

A typical DMA double buffer application is shown in [Figure 15-19](#):

- A DMA read move loads a continuous stream of 32-bit data words from an SPB connected peripheral into the DMA controller.
- A DMA write move stores the data from the DMA controller into one of two destination data buffers connected to the SRI-XBAR.

Double destination buffering with software buffer switch only is selected by using the shadow operation control bits (ADICRz.SHCT = 1010_B) as shown in [Table 15-1](#). The shadow mechanism is not available.

Transaction Control Set

The DMA active channel points at the two buffers as follows:

- The destination address register MExDADR.DADR is the address pointer to buffer 0.
- The shadow address register MExSHADR.SHADR is the address pointer to buffer 1.

The transfer count (CHCFGRz.TREL), block mode (CHCFGRz.BLKM) and address modification factors (CHCFGRz.CHDW, ADICRz.DCBE, ADICRz.DMF and ADICRz.INCD) are the same for the two buffers.

The buffer depth of the two data buffers is identical and is equal to one DMA transaction. The size of the buffer depth is a multiple of the address modification factors, the DMA transfer block mode and the DMA transaction transfer count.

DMA Operation

The DMA channel z configured for double destination buffering is programmed to execute DMA transfers of 2 X 32-bit words from a source peripheral to one of two destination buffers.

The DMA channel receives a DMA request and on winning arbitration executes a DMA transfer. During the second DMA transfer, the DMA channel z receives a swap buffer command. On completion of the current DMA transfer the frozen buffer bit MExCHSR.FROZEN is set.

At the start of the next DMA transfer the incoming data stream is re-directed to buffer 1. No data is lost, duplicated or split between the buffers. MExCHSR.TCOUNT is reloaded with CHCFGRz.TREL and executes DMA moves to the buffer 1 starting at the address stored in MExSHADR. The status bit MExCHSR.BUFFER switches state to indicate that the DMA controller is filling buffer 1. The cyclic software task operates on buffer 0 and on completion clears MExCHSR.FROZEN and re-programmes the destination address register to the buffer 0 start address. Buffer 0 is now available to receive data and the software will issue a switch buffer instruction and in turn buffer 1 will become available for a cyclic software task.

Direct Memory Access (DMA)

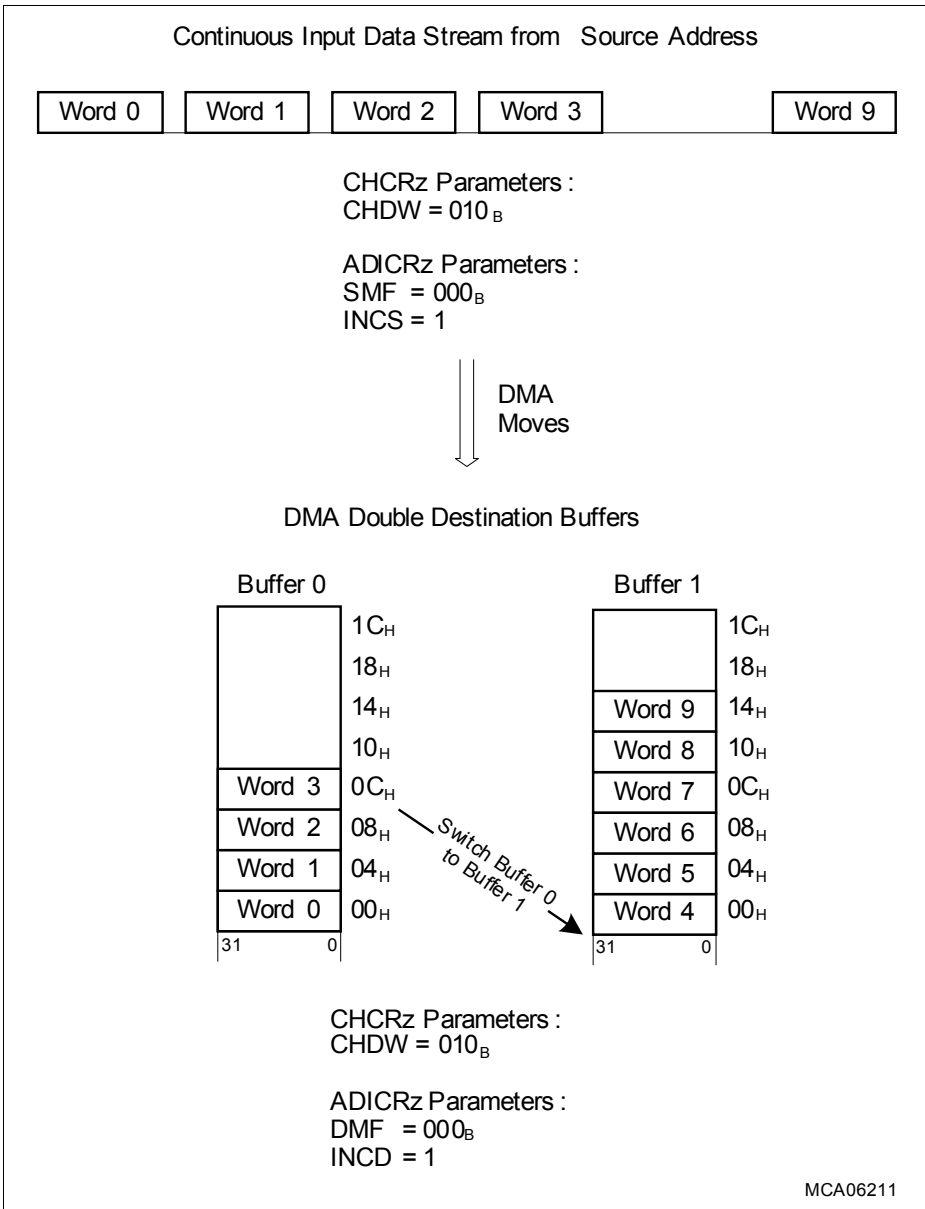


Figure 15-19 DMA Double Destination Buffering

Direct Memory Access (DMA)**15.4.3.12 Linked Lists**

Linked lists are an extension of the DMA channel functionality. Linked lists consist of a series of DMA transactions executed by the same DMA channel z. Each DMA transaction has a unique transaction control set. The source and destination areas do not have to exist in contiguous areas of memory. The new DMA transaction can issue a DMA software request and auto start the DMA transaction bypassing the Interrupt Router and so reducing the cumulative latency over a number of DMA transactions.

In a linked list, the current DMA channel transaction control set defines the shadow register as an address pointer to read the next 32-byte transaction control set from anywhere in memory and overwrite the current transaction control set in the DMARAM.

The following types of linked lists are supported:

- DMA Linked List (see [Section 15.4.3.13](#))
- Accumulated Linked List (see [Section 15.4.3.14](#))
- Safe Linked List (see [Section 15.4.3.15](#))
- Conditional Linked List (see [Section 15.4.3.16](#))

Linked lists support greater flexibility in the use of the DMA controller.

Circular Operation of Linked List

Linked lists may be configured for circular operation when the same series of DMA transactions in the linked list are endlessly repeated.

Pattern Detection During a Linked List

If a DMA transaction in a DMA, Accumulated or Safe Linked List is configured for pattern detection then a pattern match (see [Section 15.4.7](#)) will terminate linked list operation. If the pattern match occurs during the last DMA move of the DMA transaction then the current DMA move status is preserved and the next transaction control set is not loaded. Software must reconfigure the DMA channel.

Software Termination of a Linked List

If the linked list is terminated by software then the current DMA transaction is completed and the linked list is disabled.

Direct Memory Access (DMA)**15.4.3.13 DMA Linked List**

A DMA channel z supports a DMA linked list consisting of 32-Byte transaction control sets by using the shadow operation control bits (ADICRz.SHCT = 1100_B) as shown in [Table 15-15](#).

If DMA channel z is configured to support a DMA linked list then when the DMA move engine x completes the servicing of a DMA request from DMA channel z it will read the next transaction control set from memory and overwrite the DMARAM channel z with the new transaction control set. The current DMA transaction uses the shadow address as an address pointer to the next transaction control set. The address pointer to the next transaction control set must be mapped to a 32-byte aligned address in either internal or external memory and configured as shown in [Figure 15-20](#). A BTR4 access can be used to read the new transaction control set. The DMA controller will start reading the new transaction control set from the RDCRCRz word upwards. The last word to be read is CHCSRz.

If CHCSRz.SCH is set to 1 then the new transaction control set will auto start and initiate a new transaction request for DMA channel z. The access pending bit TSRz.CH will be set. The DMA controller will arbitrate against other pending requests on completion of the current DMA transfer. When DMA channel z wins arbitration the new DMA transaction will be serviced. There is no limit to the number of DMA transactions in a DMA linked list.

The DMA transactions in the linked list can be started by hardware or software requests. Auto start of a new DMA channel z transaction control set is limited to software requests.

Last DMA Transaction in DMA Linked List

The DMA traffic management interrupt service requests may be used to signal the completion of the last DMA transaction in the DMA linked list. The last DMA transaction will load a new transaction control set. The auto start is not to be set.

Loading Non Linked List Transaction Control Set

If a DMA linked list loads a non linked list transaction control set then the DMA channel exits linked list mode. The non linked list transaction control set will not auto start.

Direct Memory Access (DMA)

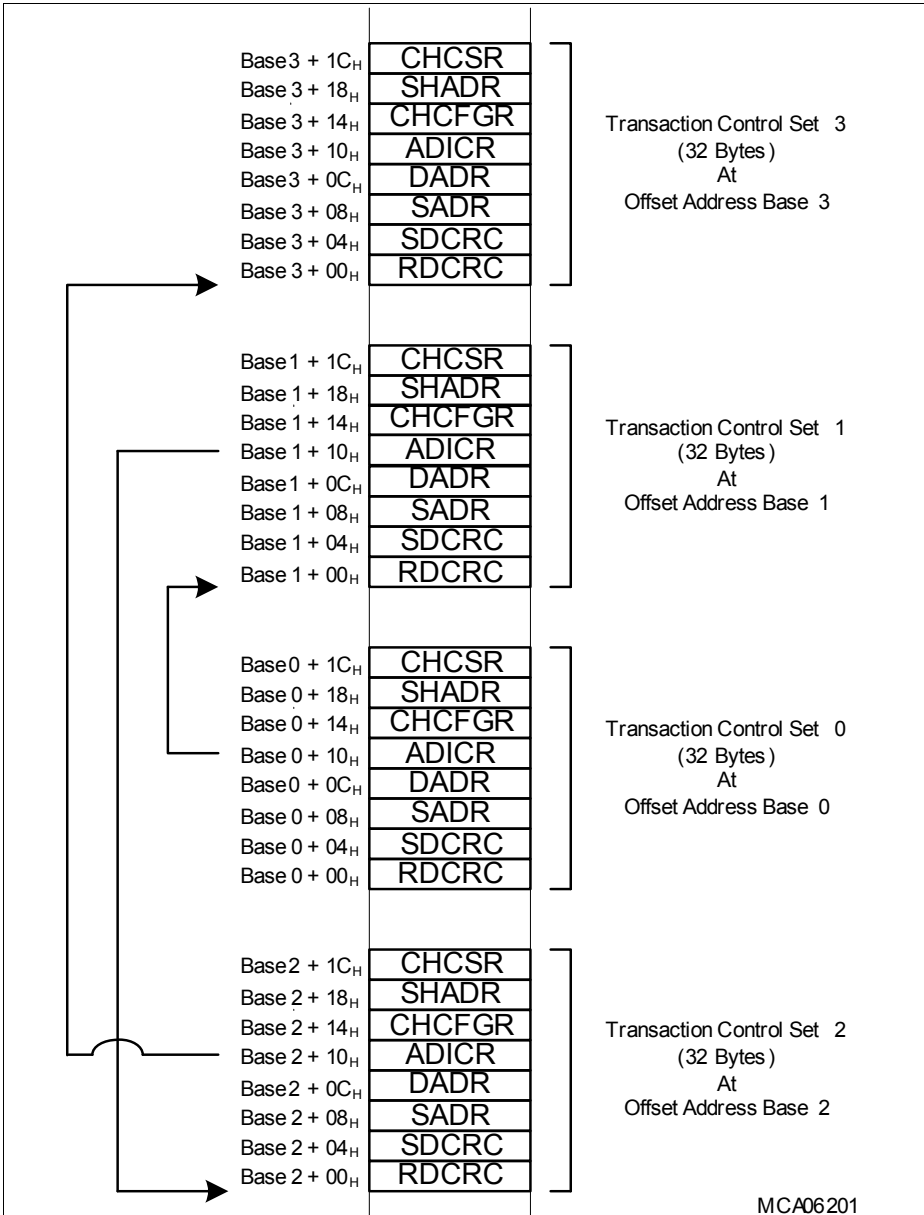


Figure 15-20 DMA Linked List

15.4.3.14 Accumulated Linked List

A DMA channel z supports an accumulated linked list by using the shadow operation control bits ($ADICRz.SHCT = 1101_B$) as shown in [Table 15-15](#). The accumulated linked list is a variant of the DMA linked list and has an identical footprint in memory (size and structure). The $SDCRCRz$ and $RDCRCRz$ words are not overwritten when the new transaction control set is written into the DMARAM channel z and are accumulated across DMA transactions.

SDCRC & RDCRC Checksums

The SDCRC and RDCRC checksums are not overwritten when the new transaction control set is loaded into the DMA channel. On completion of the linked list operation (on completion of the last DMA transaction) the SDCRC and RDCRC checksums are the values calculated across all DMA transactions.

15.4.3.15 Safe Linked List

A Safe linked list provides protection against software errors. If the shadow address in the linked list is incorrectly written then a link address pointer could point to any random area of memory and load a new transaction control set and execute it. A DMA channel z supports a safe linked list by using the shadow operation control bits ($ADICRz.SHCT = 1110_B$) as shown in [Table 15-15](#).

Proceeding from one DMA transaction to the next DMA transaction in the linked list sequence is dependent on the current SDCRC checksum matching the expected SDCRC checksum stored in the next transaction control set.

SDCRC Checksum

The user is required to load the SDCRC word with an expected CRC checksum value. When the safe link list reads the new transaction control set from memory the SDCRC checksum generated by the running DMA transaction is compared against the new expected SDCRC checksum. If the checksums match then the DMA controller proceeds with the execution of the new transaction control set. And if not then a DMA error service interrupt request is triggered, error flag $ERRxSR.SLLER$ set and the DMA stops the execution of the linked list. Assuming all the SDCRC checksum values match at the end of the linked list then the checksum is the value calculated across all DMA transactions.

RDCRC Checksum

The RDCRC checksum word is not overwritten when the new transaction control set is loaded into the DMA channel. The RDCRC checksum value at the end of the linked list is the accumulated checksum across the read data for all DMA transactions.

Direct Memory Access (DMA)

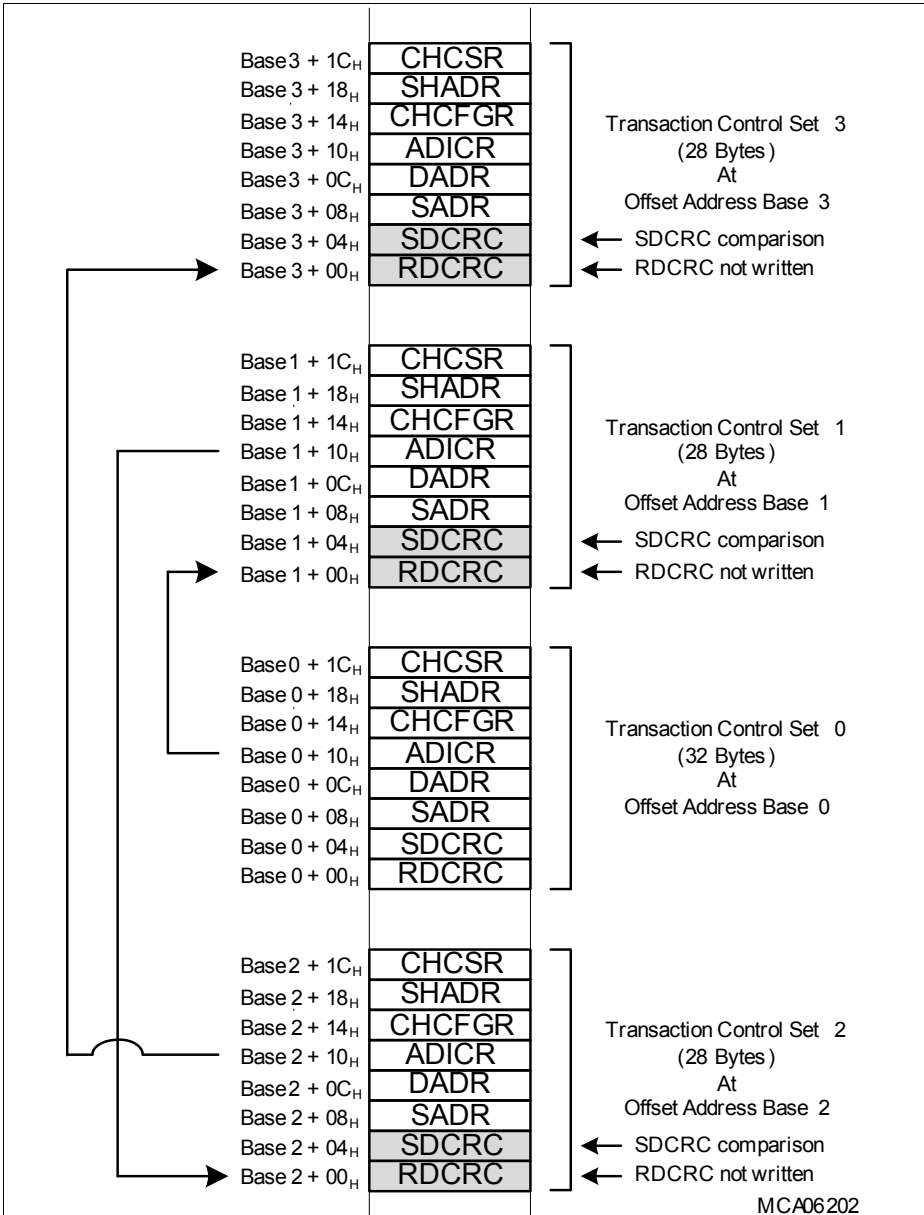


Figure 15-21 Safe Linked List

15.4.3.16 Conditional Linked List

A special use of linked lists is the Conditional Linked Lists (CLL) as shown in **Figure 15-22**. Selection of the address pointer is determined by a conditional state.

A DMA channel supports a Conditional Linked List by using the shadow operation control bits ($ADICRz.SHCT = 1111_B$) as shown in **Table 15-15**. DMA usage of CLL is limited to 8-bit DMA read moves ($CHCFGRz.CHDW = 000_B$). CLL does not support the appendage of timestamps ($ADICRz.STAMP = 0_B$).

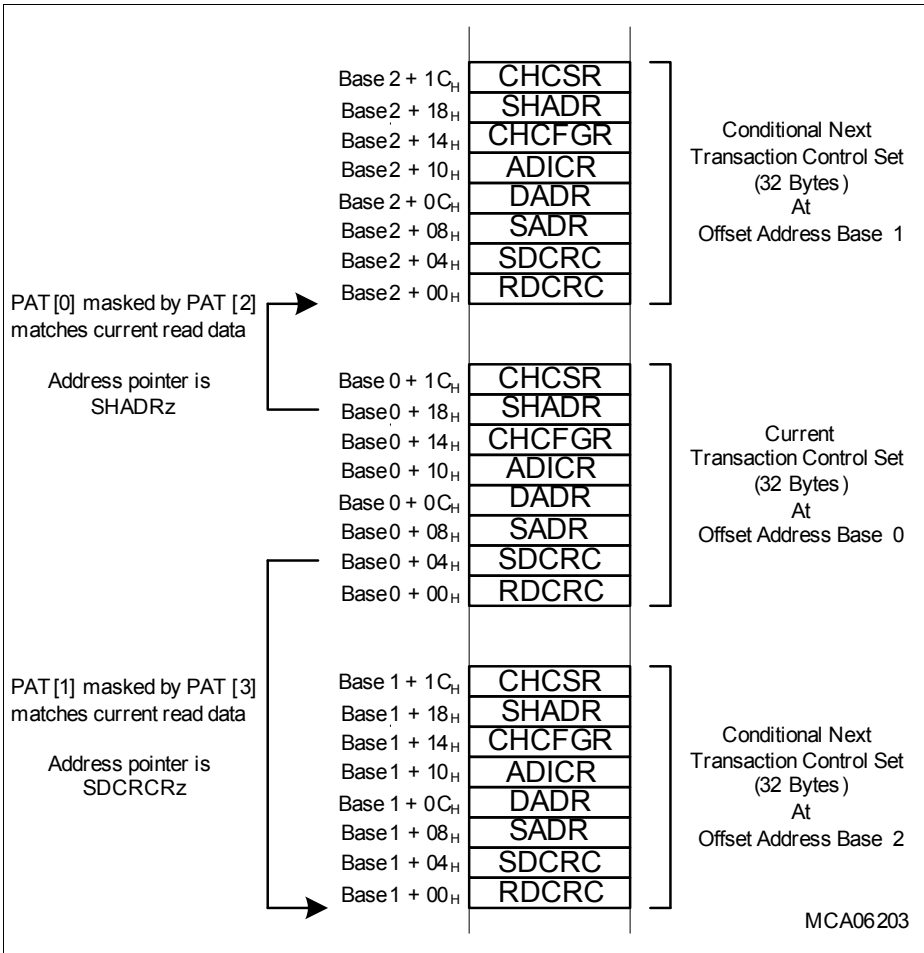


Figure 15-22 Conditional Linked List

Direct Memory Access (DMA)

In addition to using the shadow address as an address pointer CLL utilizes the source and destination address CRC checksum register SDCRCRz as a second address pointer.

CLL Pattern Detection

The DMA channel must be programmed to select one of the pattern read registers:

- PRR0 is selected if CHCFGRz.PATSEL = 000_B
- PRR1 is selected if CHCFGRz.PATSEL = 100_B

Note: If a DMA channel is configured for CLL then CHCFGRz.PATSEL[1:0] = 00_B

The configuration of the pattern detection logic to support conditional linked lists is shown in **Figure 15-23**.

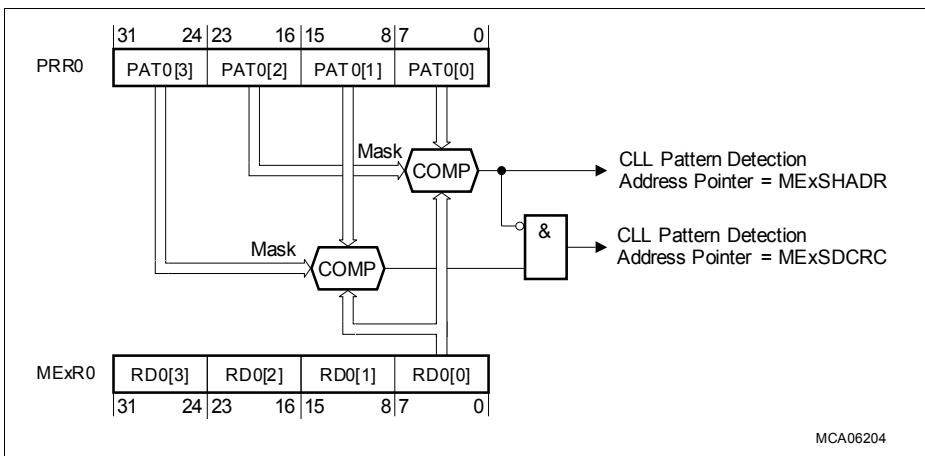


Figure 15-23 Conditional Linked List and Pattern Detection Logic

During each DMA move the pattern detection logic is used to determine the selection of the address pointer:

- If PAT[0] masked by PAT[2] is matching with the current read data MEx0R.RD0[0] then when the current DMA move is completed load a new transaction control set. The shadow address register MExSHADR stores the address pointer to the next transaction control set in the linked list.
- If PAT[1] masked by PAT[3] is matching with the current read data MEx0R.RD0[0] then when the current DMA move is completed load a new transaction control set. The source and destination address CRC checksum register MExSDCRC stores the address pointer to the next transaction control set in the linked list.
- If both PAT[0] masked by PAT[2] and PAT[1] masked by PAT[3] are matching with the current read data MEx0R.RD0[0] then when the current DMA move is completed

Direct Memory Access (DMA)

load a new transaction control set. The shadow address register MExSHADR stores the address pointer to the next transaction control set in the linked list.

- If there is no match then continue with the DMA transaction.

If a pattern match is detected then:

- The DMA transaction ends with the current DMA move.
- TSRz.CH will clear when the DMA write move has completed.

Completion of Current DMA Transaction

If the current DMA transaction completes when MExCHSR.TCOUNT = 0 and there is no pattern match then the conditional linked list will load a new transaction control set. The shadow address register MExSHADR stores the address pointer to the next transaction control set in the linked list.

Multiple Pattern Detection Conditions

The user may intentionally program PAT[0] masked by PAT[2] not to match with the current read data MEx0R.RD0[0] and transition across a series of DMA transactions in the Conditional Linked List. In each DMA transaction the user may programme a series of different PAT[1] masked by PAT[3] values and test if each value matches with the current read data MEx0R.RD0[0].

If PAT[1] masked by PAT[3] is matching with the current read data MEx0R.RD0[0] then when the current DMA move is completed load a new transaction control set. The source and destination address CRC checksum register MExSDCRC stores the address pointer to the next transaction control set in the linked list.

15.4.4 Transaction Control Engine

Each DMA sub-block has a Transaction Control Unit. The Transaction Control Unit in the DMA sub-block, as shown in the DMA Controller block diagram in [Figure 15-24](#), contains the transaction control set stored in the active channel register set and a move engine.

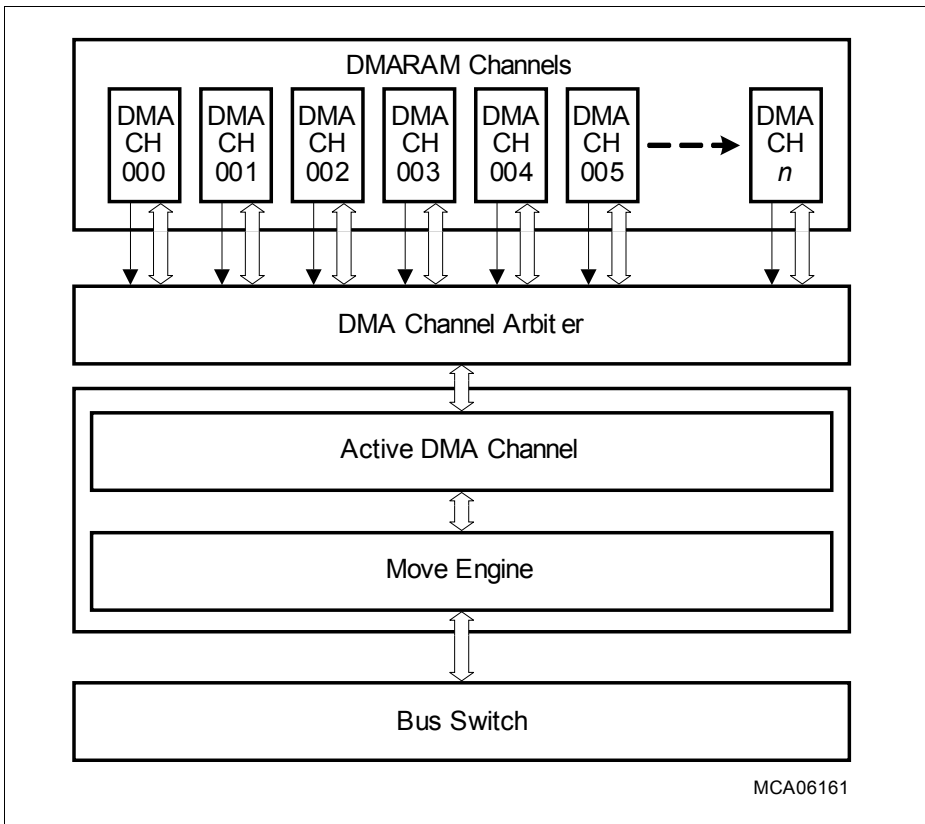


Figure 15-24 Transaction Control Engine

Any move engine can service hardware or software requests from any DMA channel.

DMA Channel Arbitration

The DMA channel arbitrator continuously monitors DMA channel transfer access pending requests. The highest number DMA channel with an access pending wins the DMA channel arbitration. On winning arbitration the transaction control set is read from the

Direct Memory Access (DMA)

DMARAM and written to the DMA sub-block active channel register set. The DMA channel parameters are passed to the move engine and a DMA transfer composed of a number of DMA moves is processed.

On completion of each DMA transfer the move engine re-arbitrates for the highest priority DMA channel and services the next DMA transfer request. If there is a higher priority DMA transfer request then the current lower priority DMA channel transaction control set is copied to the DMARAM and the active channel register set overwritten with the parameters of the higher priority DMA channel.

If the DMA channel is halted, suspended or has finished the DMA transaction then the move engine will also re-arbitrate for the highest priority DMA request.

DMA Read Buffer

Each DMA move engine includes a 256-bit buffer to store read data from DMA read moves. The read buffer supports the reading of four double words (=256-bit) of data.

A read move to a cached area of memory (Segments 8 and 9) will be translated into an SRI-Bus BTR4 transaction. The 256-bit read data will be stored together with the related 64-bit aligned address in the move engine. If the next and subsequent read access to a segment 8 or 9 address is identical (64-bit aligned) to the current read buffer contents, the requested read data will be read from the read buffer instead of from the SRI-Bus.

If the next read from a segment 8 or 9 address is not identical (64-bit alignments) to the current read buffer contents, the content of the read buffer is invalidated. A BTR4 read request is generated, the read buffer will be updated with the new 256-bit data and its related addresses.

The loading of a new DMA transfer into the move engine or a DMA write to a segment 8 or 9 address invalidates the DMA read buffer.

Address range of segments 8 & 9 is 8000 0000_H - 9FFF FFFF_H.

DMA Move Engine

The move engine requests the required buses and loads or stores data according to the parameters of a DMA transfer. It is able to wait if a targeted bus is not available.

The processing of a DMA transfer (composed of several DMA moves) of a DMA transaction by the move engine cannot be interrupted and is always completed. A DMA channel interrupt, reset, halt request or debug suspend only becomes active when the DMA channel completes the current DMA transfer.

On completion of a DMA transfer the move engine will send back the updated address register information to the active channel register set. If the current DMA channel continues to win arbitration then the move engine will continue processing DMA transfers until the DMA transaction is completed.

Possible source and destination error conditions are also reported.

15.4.4.1 Error Conditions

The DMA move engine reports bus error and source/destination error status. In the case of multiple errors, the error bits are set according to the error conditions. This means that more than one bus error flag and source/destination flag can be set.

The DMA channel should be configured to generate an error interrupt service request.

Bus Errors

The bus error flag ERRSRx.SPBER indicates an SPB Bus error occurred during a move (read or write) of a DMA module transaction.

The bus error flag ERRSRx.SRIER indicates an SRI Bus error that occurred during a move (read or write) of a DMA module transaction.

Source and Destination Errors

The source error flags ERRSRx.SER indicate that an error occurred during a DMA read move from a source address during a DMA transaction of DMA sub-block x.

The destination error flags ERRSRx.DER indicate that an error occurred during a DMA write move to a destination address during a DMA transaction of DMA sub-block x.

If a source or destination error is reported then the DMA transaction completes. If a source error is reported during a DMA read move then the DMA write move is not executed, but the destination address is updated.

Source and destination errors include unsupported types of bus transaction.

Source and Destination Errors in a Linked List

If a DMA channel is configured as a linked list and a source or destination error is reported then the current DMA transaction must complete. In the case of the Conditional Linked List pattern matches will be disabled. The new transaction control set must not be loaded and the linked list stops to allow debug of the current DMA transaction.

Transaction Control Set Load Error in a Linked List

During all linked list operations (DMA, Accumulated, Safe and Conditional) if an error is reported during the loading of a next transaction control set then the error flag ERRSRx.DLLER is set, the last error channel ERRSRx.LEC is recorded and the source of the error is set. The linked list operation will be aborted and the transaction request state bit TSRz.CH bit cleared.

Safe Linked List SDCRC Errors

A Safe Linked List must confirm that the calculated SDCRC checksum matches the expected SDCRC checksum before the next transaction control set is loaded. If the

Direct Memory Access (DMA)

SDCRC checksum does not match then a Safe Linked List error ERRSRx.SLLER is reported.

RAM Errors

When a DMA channel wins arbitration the transaction control set is copied from the DMARAM to the DMA sub-block active channel register set. If an ECC error is signalled during the DMARAM read then the error flag ERRSRx.RAMER is set and the last error channel ERRSRx.LEC recorded. The DMA transfer will be aborted and the transaction request state bit TSRz.CH bit cleared.

Transaction Request Lost

The transaction request lost error flag TSRz.TRL indicates if a DMA request for DMA channel z has been lost.

15.4.5 Bus Switch, Bus Switch Priorities

The bus switch of the DMA controller provides the connection from the DMA sub-blocks and system peripheral masters to on chip bus master interfaces (see [Figure 15-25](#)). One access can be buffered in the bus interfaces.

The Bus Switch supports Switch Address Routing to the whole memory range.

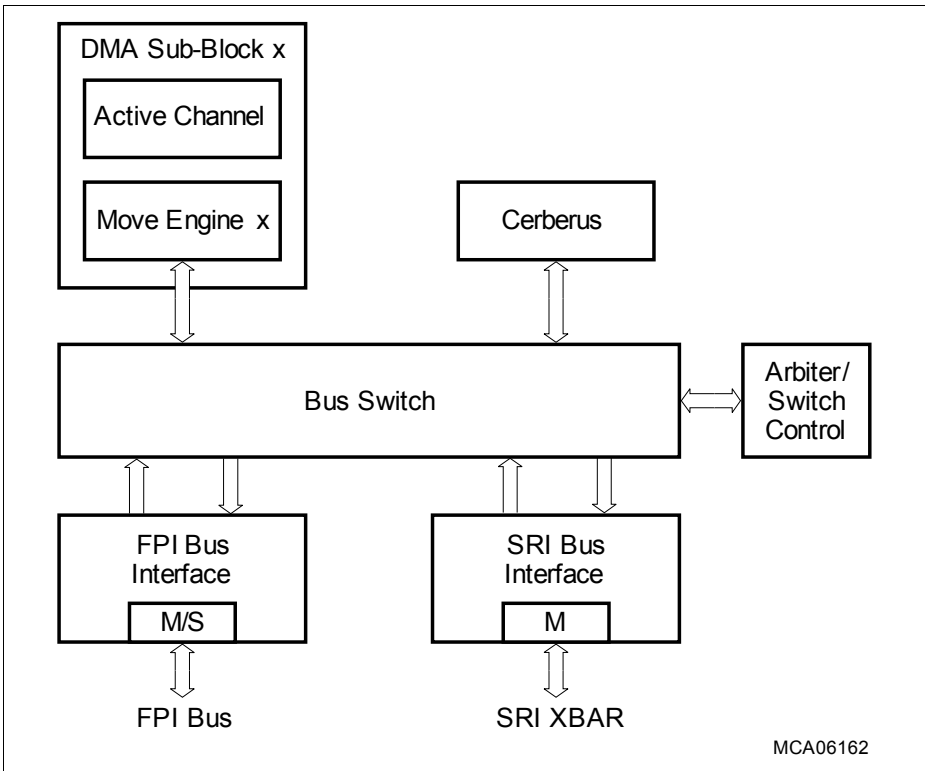


Figure 15-25 Bus Switch

Note: The accesses of the DMA Move Engine’s bus interfaces to the On Chip Bus interfaces are always done in Supervisor Mode.

The arbiter/switch control unit arbitrates the requests from the connected active interfaces (DMA move engines and Cerberus) and grants the buses connected to the switch for data transfers. [Table 15-2](#) and [Table 15-3](#) defines the Bus Switch priorities for requests to the same on chip bus interface. The arbitration scheme is valid in case of a collision of requests from active peripherals connected to the DMA bus switch (move engines) for the same resource (SPB Bus Interface, SRI Bus Interface). The arbitration is done for each bus switch request.

In case of a collision between DMA move engine requests and a Cerberus request on the DMA bus switch for the same resource, the priorities are as given in [Table 15-2](#).

Direct Memory Access (DMA)
Table 15-2 DMA Bus Switch Priorities

Priority	Agent Requests	Comment
Highest	Cerberus to On Chip Bus High	Priority selection by software in Cerberus.
	DMA move engine write	-The detailed Bus Switch priorities for the two engines with concurrent reads or concurrent writes are listed in Table 15-3 .
	DMA move engine read	-The detailed Bus Switch priorities for the two engines with concurrent reads or concurrent writes are listed in Table 15-3 .
Lowest	Cerberus to On Chip Bus Low	Priority selection by software in Cerberus.

Concurrent SRI Accesses

In case of a collision of both move engines with concurrent read requests or concurrent write requests for the SRI resource, the highest channel wins.

Concurrent SPB Accesses

In case of a collision of both move engines with concurrent read requests or concurrent write requests for the SPB resource, the move engine with the highest DMA priority determined by MExCHCR.DMAPRIO determines the winning arbitration on the DMA bus switch (see [Table 15-3](#)).

Table 15-3 DMA SPB Resource Switch Priorities of DMA Move Engines

Priority	DMA Move Engine Request	Comment
Highest	Move Engine x MExCHCR.DMAPRIO = "11"	Move engine with high DMA priority.
	Move Engine x MExCHCR.DMAPRIO = "10"	Move engine with medium DMA priority.
	Move Engine x MExCHCR.DMAPRIO = "01"	Move engine with medium DMA priority.
Lowest	Move Engine x MExCHCR.DMAPRIO = "00"	Move engine with low DMA priority.

If the move engines are processing DMA moves with concurrent read requests or concurrent write requests of equal DMA priority then the move engine processing the highest DMA channel number wins the arbitration at the DMA bus switch.

Direct Memory Access (DMA)**15.4.6 DMA Module Priorities on On Chip Busses**

Every active peripheral connected to the DMA bus switch that requests for access to SPB Bus or SRI Bus has to go through two arbitration stages before accessing the on chip bus: DMA internal arbitration at the DMA bus switch and DMA module external arbitration at the on chip bus.

The DMA can be connected to the SPB Bus and to the SRI Bus with master interfaces.

- The complete list of SPB master priorities can be found in the SPB Bus Control Unit Chapter.
- The complete list of SRI master priorities can be found in the Shared Resource Interconnect Chapter.

15.4.6.1 On Chip Bus Access Rights, RMW support

All accesses triggered by the DMA Move Engines are always done in SV mode.

The DMA module does not support read/modify write instructions.

15.4.6.2 On Chip Bus Master Interfaces

This chapter describes the features of the DMA on chip bus master interfaces to the SPB Bus and to the SRI Bus.

The DMA SPB master interface supports:

- single data read and write transactions (8-bit, 16-bit, 32-bit)
- generation of interleaved FPI transactions from different sources (move engines)
- de-assertion of request after retry in order to prevent bus blocking.
- out of order transactions from different sources in order to avoid side effects (blocking) between the different sources (move engines)
- three dedicated SPB requests (low, medium, high priority)¹⁾.

A single move engine supports only one DMA transaction from one DMA channel at a time and generates a sequence of read - write sequences. The DMA sub-block will not generate permanent, pipelined, high priority requests.

The DMA SRI master interface supports:

- single data read and write transactions (8-bit, 16-bit, 32-bit, 64-bit)²⁾
- block transfer read and write transactions (128-bit, 256-bit)³⁾
- generation of interleaved SRI transactions from different sources (move engines)

1) The complete list of SPB master priorities can be found in the SPB Bus Control Unit Chapter.

2) 64-bit DMA move supported for DMA read move from SRI source to DMA write move to SRI destination

3) Block transfer DMA move supported for DMA read move from SRI source to DMA write move to SRI destination

Direct Memory Access (DMA)

A single move engine supports only one DMA transaction from one DMA channel at a time and generates a sequence of read - write sequences. The DMA sub-block will not generate permanent, pipelined, high priority requests.

15.4.7 Pattern Detection

Pattern detection is only supported for 8-bit, 16-bit and 32-bit channel data width. Only the read data in MEx0R is compared with the value in one of the pattern read registers.

After a DMA read move, read data in the move engine least significant word read register MEx0R can be compared with data stored in one of the pattern read registers. Any move engine can service any DMA channel request. If pattern detection is selected then the DMA channel must be configured to select one of the pattern read registers (PRR0 or PRR1) for the pattern compare.

The result of a pattern compare match is always stored in a bit (MExCHSR.LXO) of the channel status register of the active DMA channel in the move engine x that is currently executing the DMA move. Therefore, the pattern match result LXO of the previous read move can also be combined together with the pattern match result of the actual read move. Move engine x read register MEx0R is overwritten with each read move.

The configuration and capabilities of the pattern detection logic further depends on the settings of MExCHCR.CHDW. CHDW determines the data width for the DMA read and write moves for each individual DMA channel z. The control bits, MExCHCR.PATSEL, selects among the different pattern detection modes for a specific value of CHDW.

If a DMA channel z is configured for pattern detection then the appendage of timestamps is not supported (ADICRz.STAMP = 0_B).

Interrupt Service Request

If MExCHCR.PATSEL[1:0] is not equal to 00_B and a pattern match is detected then:

- The DMA transaction ends with the current DMA move.
- TSRz.HTRE is cleared to stop DMA transfers on the current active channel z.
- TSRz.CH will clear when the DMA write move has completed.
- A pattern detection interrupt service request will be raised corresponding to the current active DMA channel z except in the case that a source error is reported.

The value of MExCHSR.TCOUNT and MEx0R can be read out by the interrupt software.

If MExCHCR.PATSEL=000_B or 100_B then no action will be taken when a pattern match is detected. A wrap interrupt can be used.

The software will have to service the interrupt and to activate the channel again.

15.4.7.1 Pattern Compare Logic

Read move data and compare match patterns are compared on a bit-wise level. The logic as shown in [Figure 15-26](#) is implemented in each COMP block of [Figure 15-27](#), [Figure 15-28](#), and [Figure 15-29](#). One COMP block controls either 8 bits or 16 bits of data and makes it possible to mask each data bit for the compare operation.

In the compare logic for one bit of the COMP block, a data bit from register MEx0R is compared to the corresponding pattern bit stored in a pattern read register. If both bits

Direct Memory Access (DMA)

are equal and a pattern mask bit stored in another part of pattern read register is 0, the compare matched condition becomes active. When the pattern mask bit is set to 1, the compare matched condition is always active (set) for the related bit. When the compare matched conditions for each bit within a COMP block are true, the compare match output line of the COMP block becomes active.

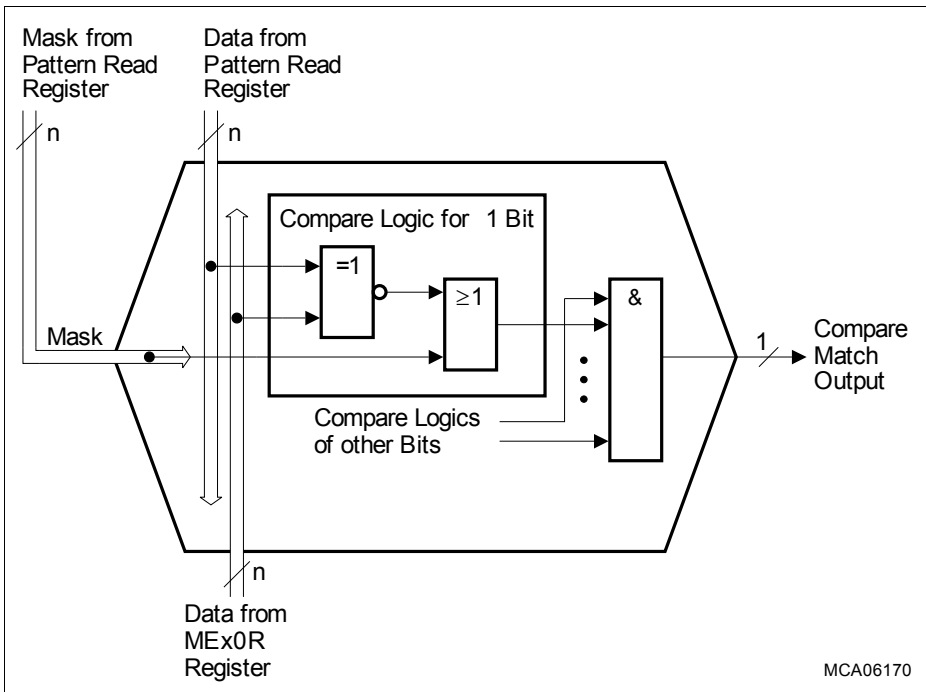


Figure 15-26 Pattern Compare Logic (COMP Block)

15.4.7.2 Pattern Detection for 8-bit Data Width

When 8-bit channel data width is selected ($MExCHCR.CHDW = 000_B$) and the channel is configured to select PRR0 ($MExCHCR.PATSEL[2] = 0_B$), the pattern detection logic is configured as shown in [Figure 15-27](#).

Direct Memory Access (DMA)

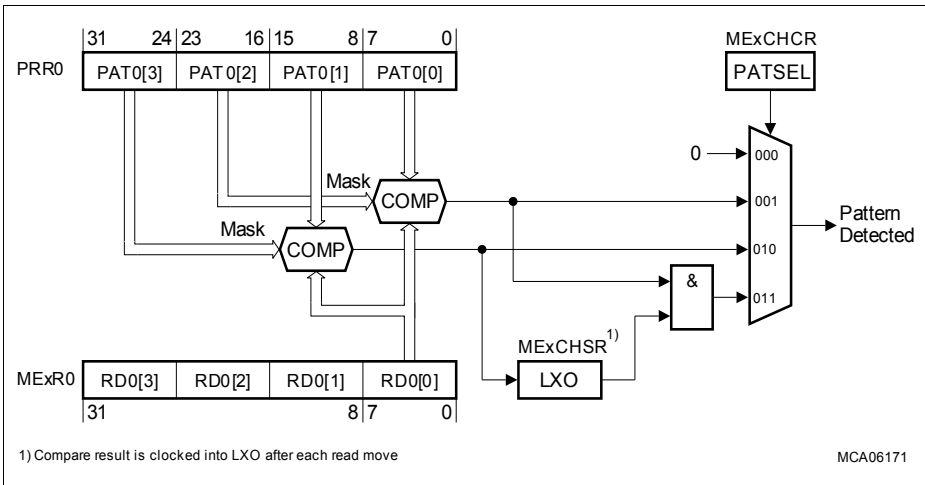


Figure 15-27 Pattern Detection for 8-bit Data Width (MExCHCR.CHDW = 000_B)

When 8-bit channel data width is selected, the pattern detection logic allows the byte of a DMA read move to be compared with two different patterns. Three compare match configurations are possible (see [Table 15-4](#)). A mask operation of each compared bit is possible.

Table 15-4 Pattern Detection for 8-bit Data Width

MExCHCR.PATSEL	Pattern Detection Operating Modes
000 _B	Pattern detection disabled
001 _B	Pattern compare of MExR.RD0[0] to PAT0[0], masked by PAT0[2]
010 _B	Pattern compare of MExR.RD0[0] to PAT0[1], masked by PAT0[3]
011 _B	Pattern compare of MExR.RD0[0] to PAT0[0], masked by PAT0[2] of the <u>actual</u> DMA read move and Pattern compare of MExR.RD0[0] to PAT0[1], masked by PAT0[3] of the <u>previous</u> DMA read move of DMA channel z
100 _B	Pattern detection disabled
101 _B	Pattern compare of MExR.RD0[0] to PAT1[0], masked by PAT1[2]

Direct Memory Access (DMA)

Table 15-4 Pattern Detection for 8-bit Data Width

MExCHCR.PATSEL	Pattern Detection Operating Modes
110 _B	Pattern compare of MExR.RD0[0] to PAT1[1], masked by PAT1[3]
111 _B	Pattern compare of MExR.RD0[0] to PAT1[0], masked by PAT1[2] of the <u>actual</u> DMA read move and Pattern compare of MExR.RD0[0] to PAT1[1], masked by PAT1[3] of the <u>previous</u> DMA read move of DMA channel z

Two Byte Pattern Detection Sequence

If the DMA channel pattern selection is configured to select a two byte sequence (MExCHCR.PATSEL[1:0] = 11_B) against PRR0 (MExCHCR.PATSEL[2] = 0_B) then after each DMA read move the pattern match result "MExOR.RD0[0] with PRR0.PAT0[1], masked by PRR0.PAT0[3]" is stored in bit MExCHSR.LXO.

This operating mode allows, for example, two-byte sequences to be detected in an 8-bit data stream coming from a serial peripheral unit with 8-bit data width (e.g.: recognition of carriage-return, line-feed characters).

15.4.7.3 Pattern Detection for 16-bit Data Width

When 16-bit channel data width is selected (MExCHCR.CHDW = 001_B) and the DMA channel is configured to select to PRR0 (MExCHCR.PATSEL[2] = 0_B), the pattern detection logic can be configured as shown in [Figure 15-28](#).

Direct Memory Access (DMA)

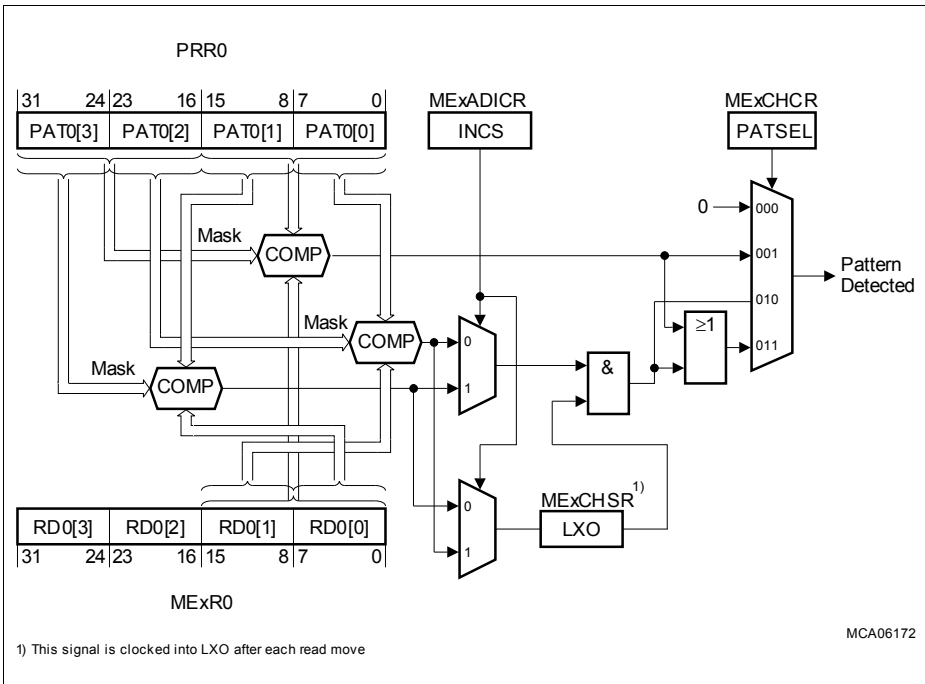


Figure 15-28 Pattern Detection for 16-bit Data Width (CHCFGRz.CHDW = 001_B)

When 16-bit channel data width is selected, the pattern detection logic makes it possible to compare the complete half-word of one read move only (aligned mode) or to compare upper and lower byte of two consecutive read moves (unaligned modes). Both modes can be combined (combined mode) too. Three compare match configurations are possible (see [Table 15-5](#)). A mask operation of each compared bit is possible.

Table 15-5 Pattern Detection for 16-bit Data Width

MExCHCR. PATSEL	MExADICR. INCS	Pattern Detection Operating Modes
000 _B	—	Pattern detection disabled
001 _B	—	Aligned Mode: Pattern compare of MEx0R.RD0[1:0] to PAT0[1:0], masked by PAT0[3:2]

Direct Memory Access (DMA)

Table 15-5 Pattern Detection for 16-bit Data Width (cont'd)

MExCHCR. PATSEL	MExADICR. INCS	Pattern Detection Operating Modes
010 _B	0	Unaligned Mode 1 (Source Address Decrement): Pattern compare of MEx0RMEx1R.RD0[1] to PAT0[0], masked by PAT0[2] of the <u>actual</u> DMA read move and Pattern compare of MEx0R.RD0[0] to PAT0[1], masked by PAT0[3] (LXO) of the <u>previous</u> DMA read move of DMA channel z
	1	Unaligned Mode 2 (Source Address Increment): Pattern compare of MEx0R.RD0[0] to PAT0[1], masked by PAT0[3] of the <u>actual</u> DMA read move and Pattern compare of MEx0R.RD0[1] to PAT0[0], masked by PAT0[2] (LXO) of the <u>previous</u> DMA read move of DMA channel z
011 _B	0 or 1	Combined Mode: Pattern compare for aligned mode (PATSEL = 001 _B) or unaligned modes (PATSEL = 010 _B)
100 _B	–	Pattern detection disabled
101 _B	–	Aligned Mode: Pattern compare of MEx0R.RD0[1:0] to PAT1[1:0], masked by PAT1[3:2]
110 _B	0	Unaligned Mode 1 (Source Address Decrement): Pattern compare of MEx0R.RD0[1] to PAT1[0], masked by PAT1[2] of the <u>actual</u> DMA read move and Pattern compare of MEx0R.RD0[0] to PAT1[1], masked by PAT1[3] (LXO) of the <u>previous</u> DMA read move of DMA channel z
	1	Unaligned Mode 2 (Source Address Increment): Pattern compare of MEx0R.RD0[0] to PAT1[1], masked by PAT1[3] of the <u>actual</u> DMA read move and Pattern compare of MEx0R.RD0[1] to PAT1[0], masked by PAT1[2] (LXO) of the <u>previous</u> DMA read move of DMA channel z
111 _B	0 or 1	Combined Mode: Pattern compare for aligned mode (PATSEL = 101 _B) or unaligned modes (PATSEL = 110 _B)

Unaligned Mode 1

In unaligned mode 1 (source address decremented), the high byte (MEx0R.RD01) of the current and the low byte (MEx0R.RD00) of the previous 16-bit DMA read move are compared.

Unaligned Mode 2

In unaligned mode 2 (source address incremented), the low byte (MEx0R.RD00) of the current and the high byte (MEx0R.RD01) of the previous 16-bit DMA read move are compared.

Combined Mode

If it is not known on which byte boundary (even or odd address) the 16-bit pattern to be detected is located, the combined mode must be used. This mode is the most flexible mode that combines the pattern search capability for aligned and unaligned 16-bit data searches.

15.4.7.4 Pattern Detection for 32-bit Data Width

When 32-bit channel data width is selected (MExCHCR.CHDW = 010_B) the pattern detection logic and the DMA channel is configured to select to PRR0 (MExCHCR.PATSEL[2] = 0_B), is configured as shown in [Figure 15-29](#).

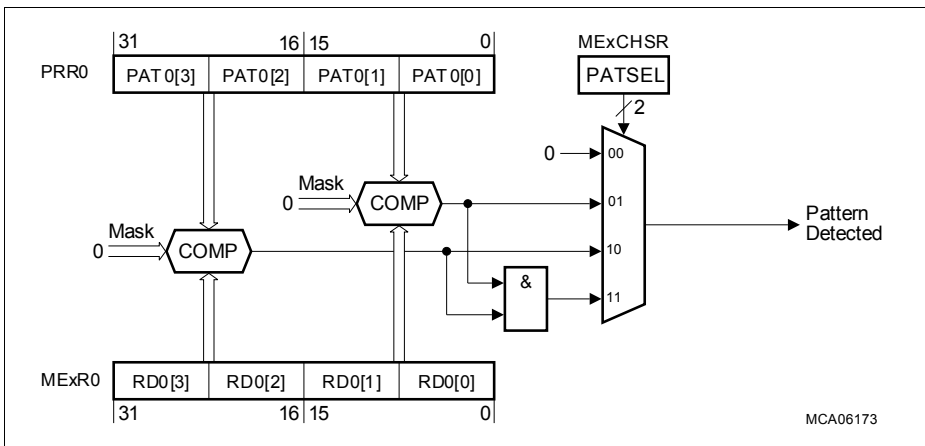


Figure 15-29 Pattern Detection for 32-bit Data Width (MExCHCR.CHDW = 010_B)

In 32-bit channel data width mode, the pattern detection logic makes it possible to compare the lower half-word only, the upper half-word only, or the complete 32-bit word

Direct Memory Access (DMA)

with a pattern stored in the pattern read register. Three compare match configurations are possible (see [Table 15-6](#)). A mask operation is not possible.

Table 15-6 Pattern Detection for 32-bit Data Width

MExCHCR.PATSEL	Pattern Detection Operating Modes
000 _B	Pattern detection disabled
001 _B	Unmasked pattern compare of RD0[1:0] to PAT0[1:0]
010 _B	Unmasked pattern compare of RD0[3:2] to PAT0[3:2]
011 _B	Unmasked pattern compare of RD0[3:0] to PAT0[3:0]
100 _B	Pattern detection disabled
101 _B	Unmasked pattern compare of RD0[1:0] to PAT1[1:0]
110 _B	Unmasked pattern compare of RD0[3:2] to PAT1[3:2]
111 _B	Unmasked pattern compare of RD0[3:0] to PAT1[3:0]

Direct Memory Access (DMA)**15.4.8 DMA Configuration Interface**

The DMA controller implements a standard FPI slave interface compliant with the FPI bus protocol on the SPB bus. The SPB slave interface supports the following functions:

- Control of the DMA general control registers to support clock control, power management, debug, etc.
- Reading and writing DMA registers that manage DMA transactions.
- Reading and writing the transaction control set for each DMARAM channel z.
- Reading the active channel transaction registers in the DMA sub-block.
- Configuring and reading the move engine and error registers.

The DMA controller supports single data transfers and does not support block transfers.

15.4.8.1 DMARAM Channel Control and Status Word

The DMARAM Control and Status Word CHCSRz supports two functions:

- Storing channel status bits including the transfer count status. The status is updated when the active channel is written back on completion of a DMA transaction or on completion of a DMA transfer on losing channel arbitration.
- Write only triggers to set and clear state in the channel configuration and status.

The write only control and clear triggers (CHCSRz.SCH, TSRz.ECH, TSRz.DCH, CHCSRz.SWB, CHCSRz.CICH, CHCSRz.CWRP and CHCSRz.SIT) can be written independently of the channel transaction state (active, pending or idle).

15.4.8.2 DMA Active Channel Write Back

A DMA channel is active when a move engine services a hardware or software request. When a pending DMA channel request wins arbitration the transaction control set is copied from the DMARAM and loaded into the DMA sub-block active channel registers and the active channel field MExSR.CH is updated. The DMA channel is defined to be active.

The point at which transaction control set is written back to the DMARAM is dependent on the MExCHCR.RROAT:

- MExCHCR.RROAT = 0, the transaction control set is written back on completion of each DMA transfer.
- MExCHCR.RROAT = 1, the transaction control set is written back when the DMA channel loses channel arbitration and completes the current DMA transfer.

If a write back has occurred then the transaction control set stored in the DMARAM will not match the transaction control set held in the DMA sub-block active channel registers.

Direct Memory Access (DMA)

15.4.8.3 DMA Active Channel Shadow Control

An SPB write to a DMARAM channel z addresses other than the CHCSRz word when DMA channel z transaction request state bit TSRz.CH has an access pending will result in a bus error with the following shadow address source or destination address buffering exceptions as defined in [Table 15-7](#).

Table 15-7 DMA Active Channel SPB Write Shadow Control Exceptions

SHCT	Description
0001 _B	<p>Source Address Buffering (Read Only)</p> <p>Active DMA channel z SPB writes to the source address SADRz are permitted. The shadow register MExSHADR in DMA sub-block x stores the SADRz value while DMA channel z is active in move engine x.</p> <p>On write back from DMA sub-block to DMARAM:</p> <ul style="list-style-type: none"> • The source address SADRz stored in DMARAM is updated to the shadow address SHADRz value. • The shadow address SHADRz stored in DMARAM is set to 00000000_H.
0101 _B	<p>Source Address Buffering (Direct Write)</p> <p>Active DMA channel z SPB writes to the shadow address SHADRz are permitted and result in a update to the active channel shadow register MExSHADR in DMA sub-block x.</p> <p>On write back from DMA sub-block to DMARAM:</p> <ul style="list-style-type: none"> • The shadow address SHADRz stored in DMARAM is updated on completion of the DMA transfer. • The source address SADRz is updated to the SHADRz value.
0010 _B	<p>Destination Address Buffering (Read Only)</p> <p>Active DMA channel z SPB writes to the destination address DADRz are permitted. The shadow register MExSHADR in DMA sub-block x stores the DADRz value while DMA channel z is active in move engine x.</p> <p>On write back from DMA sub-block to DMARAM:</p> <ul style="list-style-type: none"> • The source address DADRz stored in DMARAM is updated to the shadow address SHADRz value. • The shadow address SHADRz stored in DMARAM is set to 00000000_H.
0110 _B	<p>Destination Address Buffering (Direct Write)</p> <p>Active DMA channel z SPB writes to the shadow address SHADRz are permitted and result in a update to the active channel shadow register MExSHADR in DMA sub-block x.</p> <p>On write back from DMA sub-block to DMARAM:</p> <ul style="list-style-type: none"> • The shadow address SHADRz stored in DMARAM is updated on completion of the DMA transfer. • The destination address DADRz is updated to the SHADRz value.

Direct Memory Access (DMA)

15.4.8.4 DMARAM Write Back During Linked List Execution

When a linked list is executed by move engine x the contents of the move engine read register are selectively written back to the DMARAM transaction control set as defined in [Table 15-8](#).

Table 15-8 Linked List Write Backs

SHCT	Description
1100 _B	<p>DMA Linked List</p> <p>The DMA controller reads a DMA channel transaction control set and overwrites 8 X words in the DMARAM.</p> <p><i>Note: The SDCRC and RDCRC checksums are unique for each DMA transaction in the DMA Linked List.</i></p>
1101 _B	<p>Accumulated Linked List</p> <p>The DMA controller reads a DMA channel transaction control set and overwrites 6 X words in the DMA RAM. The SDCRC and RDCRC words are not overwritten.</p> <p><i>Note: The SDCRC and RDCRC checksums are accumulated across all DMA transactions of the DMA Linked List.</i></p>
1110 _B	<p>Safe Linked List</p> <p>The DMA controller reads a DMA channel transaction control set. The Linked List only proceeds with the next DMA transaction if the existing SDCRC checksum matches the expected SDCRC checksum in the loaded from the new DMA transaction control set.</p> <p><i>Note: The SDCRC and RDCRC checksums are accumulated across all DMA transactions of the DMA Linked List.</i></p>
1111 _B	<p>Conditional Linked List</p> <p>Shadow address register (MExSHADR) and source and destination address CRC register (MExSDCRC) are used as address pointers to a DMA linked lists. The selection of the address pointer is determined by DMA channel pattern detection conditions.</p> <p><i>Note: Calculation of SDCRC checksums is not supported.</i></p>

15.4.9 Interrupt Service Requests

The interrupt structure of the DMA controller is very flexible. There are five different types of interrupt events. The service control registers are located in the Interrupt Router.

- Each DMA channel has its own interrupt service request that covers DMA channel traffic management events:
 - Channel transfer interrupt service request
 - Channel pattern detection interrupt service request
 - Channel source and destination wrap buffer interrupt service requests
- The DMA controller has one interrupt service request covering all error events:
 - Channel transaction request lost interrupt service request
 - Channel safe linked list SDCRC checksum comparison error
 - Move engine source and destination error interrupt service request
 - Read DMARAM ECC error interrupt service request
 - DMA linked list transaction control set load error interrupt service request

The DMA controller interrupt service requests are shown in [Figure 15-30](#). The number of interrupt service requests in the DMA controller is one greater than the number of DMA channels.

The channel interrupt service requests detect normal DMA traffic events and raise an interrupt request via the shared DMA channel interrupt service request output. The interrupt event is internally generated as a request pulse and always stored in the active DMA Channel Status Register MEXCHSR.

The move engine interrupt service requests detect error conditions and raise an interrupt request via the shared error interrupt service request output. Source and destination errors are stored in the move engine error status register ERRSRx. The transaction request lost condition is stored in the channel status register TSRz.TRL. A transaction request lost error interrupt service request is generated if ADICRz.ETRL enables the service request when channel z becomes an active DMA channel.

The routing of both channel transaction request lost and move engine source and destination error conditions via the error interrupt service request centralizes the reporting of all errors in each DMA sub-block to one interrupt service request. In the event that a DMA error interrupt request is raised then the error handler will read the contents of the move engine error status register ERRSRx and all the TSRz.TRL bits to identify the source of the error.

Software Activation of Interrupt Service Requests

Each channel interrupt service request can be activated by programming CHCSRz.SIT. The error interrupt service request can be activated by programming ERRINTR.SIT.

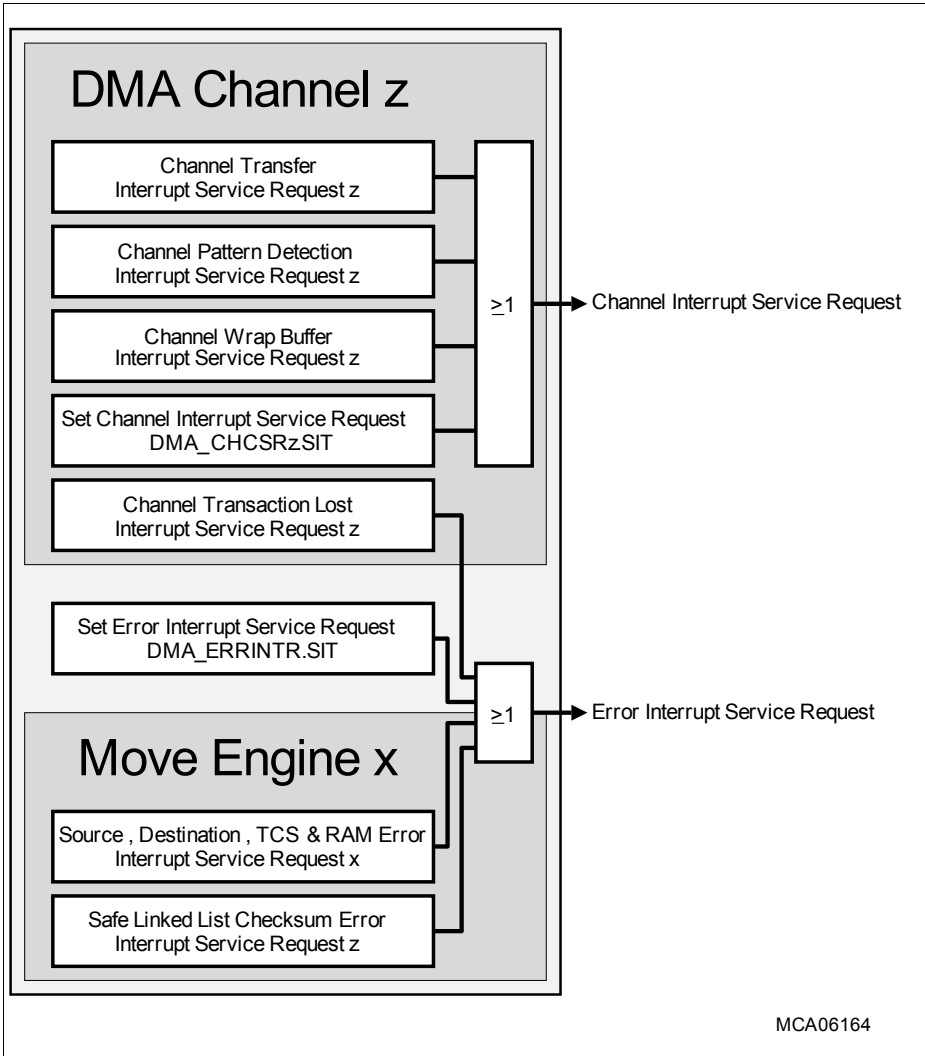


Figure 15-30 DMA Controller Interrupt Service Requests

The following sections describe each of the interrupt service request types in detail.

15.4.9.1 Channel Transfer Interrupt Service Request

The channel transfer interrupt service request is always activated after a DMA transfer, or when MExCHSR.TCOUNT matches with the value of bit field MExADICR.IRDV after it has been decremented after a DMA transfer. A channel transfer interrupt is indicated when status flag MExCHSR.ICH (CHCSRz.ICH) is set. The status flag can be reset together by software when setting bit CHCSRz.CICH (or TSRz.RST). The channel interrupt of DMA channel z is enabled when bit ADICRz.INTCT[1] is set.

Bit MExADICR.INTCT[0] selects one of two types of interrupt source. For the compare operation, bit field MExADICR.IRDV (4-bit) is zero-extended to 14-bit and then compared with the 14-bit TCOUNT value. This means that a TCOUNT match interrupt can be generated after one of the last 16 DMA transfers of a DMA transaction. Note that with $IRDV = 0000_B$, the match interrupt is generated at the end of a DMA transaction (after the last DMA transfer).

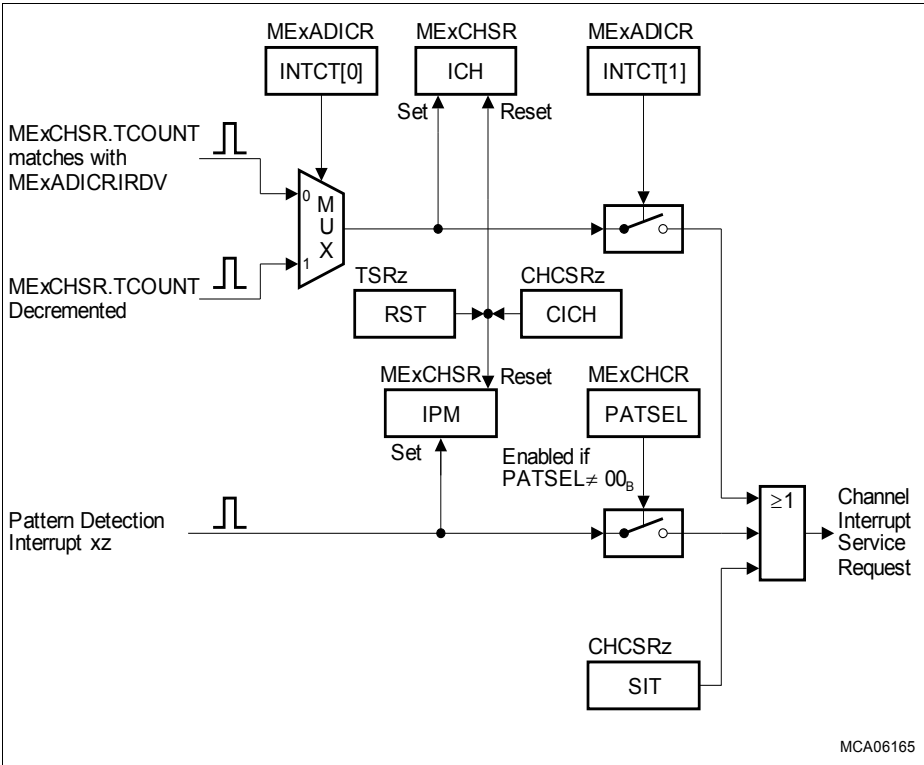
If MExCHCR.PRSEL = 1B for DMA channel z then the channel transfer service request interrupt is masked and the access pending bit is set in the next lower priority DMA channel z-1.

15.4.9.2 Channel Pattern Detection Interrupt Service Request

The channel pattern detection interrupt service request is enabled when MExCHCR.PATSEL[1:0] is not equal 00_B . The pattern detection interrupt is indicated when status flag MExCHSR.IPM is set. The status flag MExCHSR.IPM can be reset together by software when setting bit CHCSRz.CICH (or TSRz.RST).

Further details on the pattern detection are described in [Section 15.4.7](#).

Direct Memory Access (DMA)



MCA06165

Figure 15-31 Channel Transfer and Pattern Detection Interrupt Service Requests

Figure 15-31 shows the implementation of the channel transfer interrupt service request, the pattern detection interrupt service request and the channel set interrupt service request functions.

15.4.9.3 Channel Wrap Buffer Interrupt Service Request

Each DMA channel z is able to generate a wrap buffer interrupt for source buffer or destination buffer overflow.

A wrap source buffer interrupt of DMA channel z is indicated by status flag MExCHSR.WRPS. A wrap destination buffer interrupt of DMA channel z is indicated by the status flag MExCHSR.WRPD. Both interrupt status flags can be reset by software when bit CHCSRz.CWRP (or TSRz.RST) becomes set. The wrap source buffer interrupt is enabled when bit MExADICR.WRPSE is set. The wrap destination buffer interrupt is enabled when bit MExADICR.WRPDE is set. The two interrupts for wrap source buffer and wrap destination buffer are OR-ed together with other channel interrupt service requests to form one common channel interrupt service request.

The channel pattern match detection must not be enabled while a wrap interrupt is enabled for the same channel.

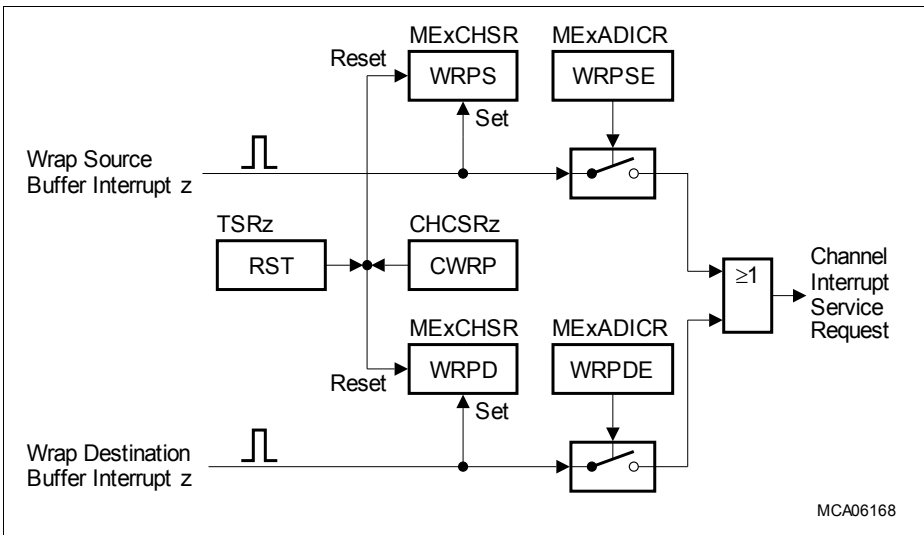


Figure 15-32 Channel Wrap Buffer Interrupt Service Requests

15.4.9.4 Transaction Request Lost Interrupt Service Request

Each DMA channel z is able to detect a transaction request lost condition. This condition becomes true when a new hardware or software DMA request occurs while the previous transaction or transfer on DMA channel z is not finished, indicated by TSRz.CH is still set. If such a transaction request lost condition occurs then bit TSRz.TRL is set.

A transaction request lost condition of DMA channel z is indicated by status flag TSRz.TRL, which can be reset by setting bit TSRz.CTL or TSRz.RST.

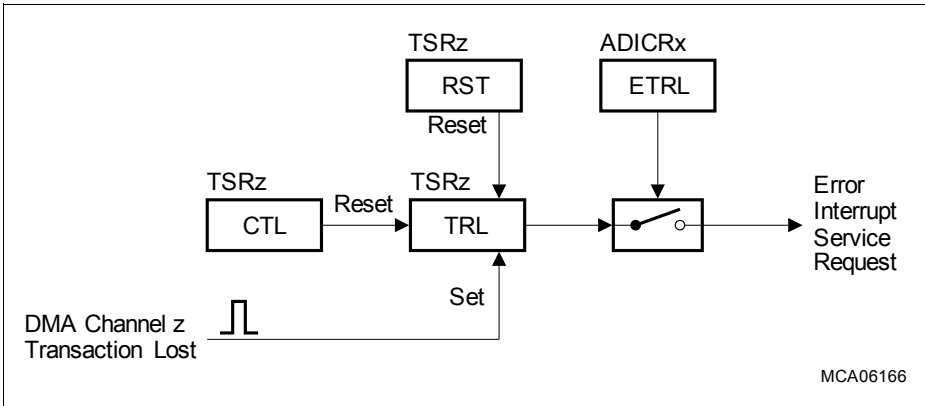


Figure 15-33 Transaction Request Lost Interrupt Service Requests

Interrupt Service Request

The transaction request lost interrupt service request for DMA channel z is enabled when bit ADICRz.ETRL is set. The interrupt service request is routed via the DMA error service request.

15.4.9.5 Source and Destination Error Interrupt Service Requests

The Move Engine is able to detect error conditions that occur during accesses to the bus interfaces of the Bus Switch (see [Figure 15-25](#)). Two error conditions can be detected:

- Source error indicates a bus error occurred during a read move from the data source.
- Destination error indicates a bus error that occurred during a write move to the data destination.

A source error of Move Engine x is indicated by the status flag ERRSRx.SER. Status flag ERRSRx.SER can be reset by software when setting bit CLREx.CSER. The source error interrupt of Move Engine x is enabled when bit EERx.ESER is set. Separate reset, status, and enable bits are available in the Move Engines for source error condition, as well as for destination error condition.

Note that in case of a read move error, the write move is not executed but the destination address is updated.

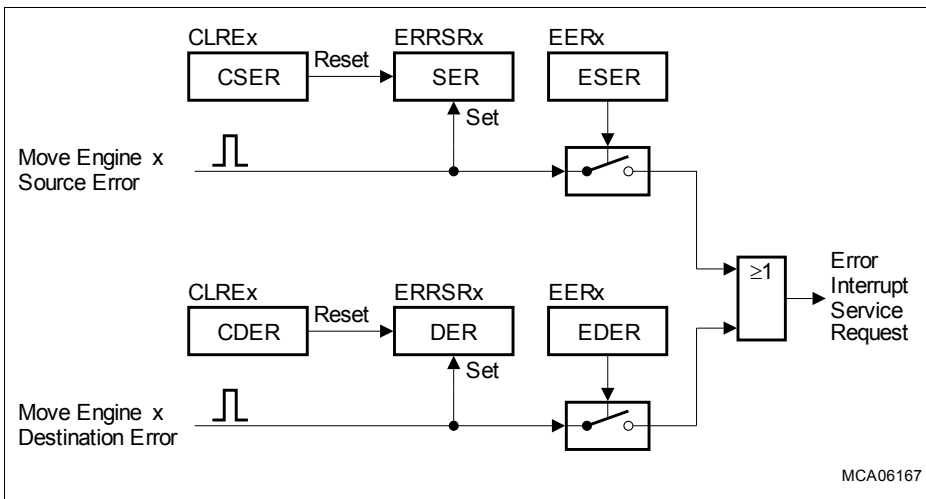


Figure 15-34 Move Engine Interrupt Service Requests

When a move engine source or destination error occurs, additional status bits and bit fields are provided in the error status register ERRSRx to indicate the following two status conditions:

- At which on chip bus interface a move engine error occurred.
- For which DMA channel z a move engine read or write move error was reported (ERRSRx.LEC).

Direct Memory Access (DMA)

These error status bits and bit fields are required by error handler software to detect in detail at which on chip bus interface and DMA channel z the move engine error has been generated.

- ERRSRx.SPBER is reset when bit CLREx.CSPBER is set.
- ERRSRx.SRIER is reset when bit CLREx.CSRIER is set.

The move engine interrupt service request can be activated by programming ERRINTR.SIT.

15.4.9.6 DMA Linked List Error Interrupt Service Request

During all DMA linked list operations the move engine is able to detect an error if the next transaction control set is not loaded correctly. This condition becomes true when a bus error is reported.

A DMA linked list error is indicated by the error status flag ERRSRx.DLLER. Status flag ERRSRx.DLLER can be reset by software when setting bit CLREx.CDLLER. Additional status bits are provided to indicate the following:

- At which on chip bus interface a move engine error occurred.
- For which DMA channel z the error was reported (ERRSRx.LEC).

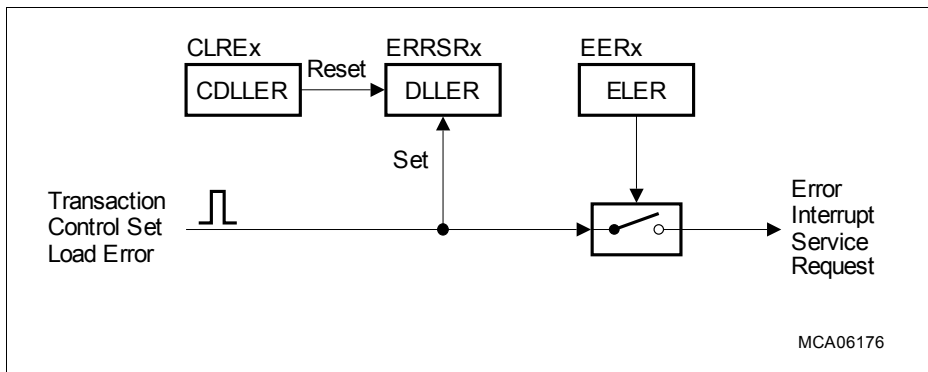


Figure 15-35 DMA Linked List Error Interrupt Service Requests

Interrupt Service Request

The DMA linked list error interrupt service request is enabled when bit EERx.ELER is set. The interrupt service request is routed via the DMA error service request.

15.5 Power Modes

15.5.1 Sleep Mode

Sleep Mode is enabled by the DMA Clock Control Register active low Sleep Mode Enable Control bit DMA_CLC.EDIS. If the enable bit is active when the sleep request input is activated then the DMA clock will be switched off when the move engines have completed any pending DMA transfers.

15.6 Functional Safety Features

15.6.1 Access Protection

A slave destination controls access to its bus peripheral interfaces and kernel address registers. Each on chip resource with bus master capability has a unique master tag identifier that is used to identify the source of an on chip bus transaction. The master tag identifier based access protection is used to enable write accesses to individual slave address ranges. There is no master tag identifier access control associated with read accesses.

Each DMA channel *z* including its transaction control set stored in DMARAM is assigned to one of *y* hardware resource partitions (default 0) and is write access control protected. Write accesses to DMA channel *z* control registers are only possible if the on chip bus master performing the write access has its master tag identifier enabled by the assigned hardware resource partition. Groups of DMA channels can be assigned to a hardware resource and write accesses are limited to individual on chip bus masters.

DMA Master Tag Identifiers

Each hardware resource partition supports a unique master tag identifier as shown in [Table 15-9](#). The master tag identifier is driven onto the bus during a read or write access and allows the bus controller to identify the hardware resource requesting the access.

Table 15-9 Bus Master Tag Identifiers

Hardware Partition	Master Tag Identifier
DMA Hardware Resource 0	000110
DMA Hardware Resource 1	000111
DMA Hardware Resource 2	001000
DMA Hardware Resource 3	001001

Direct Memory Access (DMA)**DMA Supervisor/User Mode**

The supervisor or user mode setting for a hardware resource is determined by the control bit `MODEy.MODE`. All the DMA channels assigned to a hardware resource assume the hardware resource supervisor or user mode setting. If the DMA bus master accesses a supervisor mode protected slave interface in user mode then the slave will reply with `ERR` acknowledge. The response will be recorded in the Move Engine x Error Status Register.

DMA Write Move

A DMA channel is programmed to read data from a source address and write the data to a destination address. For a DMA write move operation to successfully complete the slave access control logic must enable write accesses to the destination address for the bus master tag identifier of the requesting DMA hardware resource.

15.6.2 Data Integrity

The following data integrity error conditions are detected in the DMA and reported to the SMU:

- Error Correcting Code (ECC) errors generated by the DMARAM.
- Error Correcting Code (ECC) errors generated from SRI data.

15.6.2.1 DMARAM

The DMA controller detects several different classes of ECC error. These errors will set a status flag in the `MEMCON` register and send a trigger to the SMU for the processing of the error condition. ECC errors are reported for the following error conditions:

Internal ECC Error

The DMARAM is accessed by the internal DMA controller logic during the loading of the transaction control set into the DMA sub-block. If an ECC error occurs then the `MEMCON.INTERR` is set and a trigger is sent to the SMU. The DMA transaction will not take place and the channel number will be recorded in the move engine last error channel bit field `ERRSRx.LEC` and the error status bit `ERRSRx.RAMER` set.

SPB Read Access

If an ECC error is signalled during an SPB read access to the DMARAM then the `MEMCON.DATAERR` status bit will be set and a trigger sent to the SMU.

SPB Write Access

An SPB write access will require the DMARAM to perform an internal Read Modify Write (iRMW). If an ECC error is reported by the DMARAM during the read phase then the

Direct Memory Access (DMA)

MEMCON.RMWERR bit will be set and a trigger sent to the SMU. The write phase of the iRMW will not take place.

15.6.2.2 DMA SRI Read and Write Data

The SRI Bus protocol supports the reliable delivery of data between sources and destinations by extending the protocol to include ECC. The ECC word is generated across the address and data words. The DMA read move supports data integrity checking for data sourced from SRI Bus destinations. The ECC word is recoded for a subsequent DMA Write Move to an SRI Bus destination address.

ECC errors can be generated at two locations in the DMA:

- The DMA checks the integrity of read data received across the SRI Bus from SRI sources during a DMA read move.
- Write data to SRI destinations will be sourced from the Move Engine Read Register and is checked for data integrity during a DMA write move.

System Peripheral Bus

The SPB-Bus protocol does not support data integrity checking. ECC extensions will be generated when the read data is stored in the Move Engine Read Register. The integrity of SPB sourced data will be checked when it is written to the SRI Bus.

Data Integrity Testing

The alarm can also be triggered by software. The MBIST is used to generate an ECC error condition in a memory used as a DMA source. When the data is passed through the DMA it will generate an ECC error condition. See MTU chapter for details.

15.7 Debug Features

The DMA controller supports the following debugging capabilities:

- Channel suspend
- OCDS Trigger Bus
- MCDS Trace Interface

15.7.1 Channel Suspend Mode

The on chip debug control unit is able to generate a channel suspend mode request (SUSREQ) for the DMA controller. When this channel suspend request becomes active, the state of a DMA channel becomes frozen regarding hardware changes to ensure that the state of the DMA channels can be analyzed by reading the register contents. Pending transfers in the DMA module on chip Bus master interfaces (SPB master interface, SRI master interface) are completed. The DMA controller signals its channel suspend mode back to the on chip debug control via a suspend acknowledge.

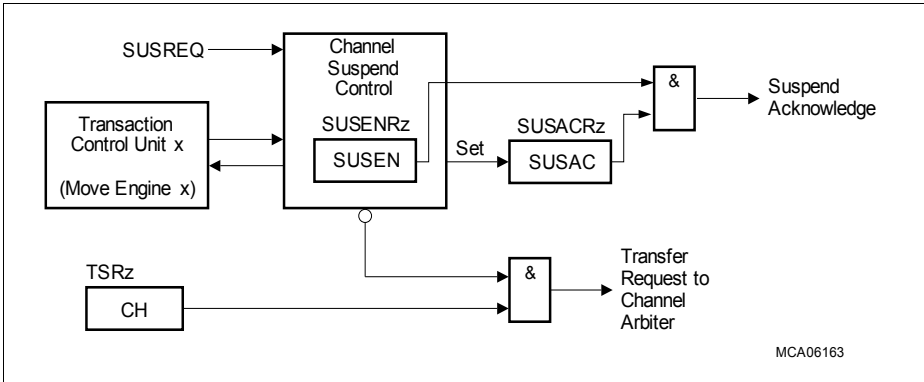


Figure 15-36 Channel Suspend Mode Control

Entering Channel Suspend Mode

Channel suspend mode of DMA channel z is entered if its suspend enable bit in the Suspend Mode Enable Register SUSENRz.SUSEN is set. When SUSREQ becomes active, the operation of all DMA channels enabled for channel suspend mode is stopped automatically after their current DMA transfer has finished in the transaction control unit. Afterwards, the suspend active status flag SUSACRz.SUSAC is set, indicating that DMA channel z is in channel suspend mode.

The DMA suspend acknowledge becomes active when all DMA channels z that are enabled for the Channel Suspend Mode have set their suspend active status flag SUSACRz.SUSAC. In Channel Suspend Mode, register contents can be modified. These modifications are taken into account for further DMA transactions or DMA transfers of the related DMA channel after Suspend Mode has been left again.

DMA channels that are disabled for Suspend Mode (SUSENRz.SUSEN = 0) continue with its normal operation.

Exiting Channel Suspend Mode

Channel suspend mode of DMA channel z is left and its normal operation continues if either the SUSREQ signal becomes inactive, or if the enable bit SUSENRz.SUSEN is reset by software.

15.7.2 OCDS Trigger Bus (OTGB) Interface

A Trigger Set is a collection of signals which supports a specific debug use case. The OCDS Trigger Set Select Register controls which Trigger Set is applied to the bus. The register is cleared by Debug Reset and by each System Reset when OCDS is disabled.

Direct Memory Access (DMA)

It is not touched by the System Reset when OCDS is enabled. The trigger sets are routed by the OCDS Trigger Mux (OTGM) to the OCDS Trigger Switch (OTGS) and the MCDS. A DMA Trigger Set is unique to each DMA module dependent on the number of channels and move engines.

15.7.3 MCDS Trace Interface

The DMA module provides two 8 bit vectors for debug and trace signal generation purposes that are used for OCDS Level 3 and for OCDS Level 1:

- One 8 bit vector from SPB master interface via BCU_SPB to BAL_SPB inside MCDS (i.e. spb_clk domain).
- One 8 bit vector from SRI master interface via SRI_XBAR to BAL_SRI inside MCDS (i.e. sri_clk domain).

For each DMA master interface trace is captured to identify the DMA move engine and channel making the request.

Table 15-10 Trace Vector Definition

Bit	Trace
[6:0]	DMA Channel
[7]	DMA Move Engine 0 _B Move Engine 0 1 _B Move Engine 1

DMA Trace Signal Generation for OCDS Level 3

The trace mechanism allows the identity of the DMA move engine and channel accessing the on chip bus to be determined. The mechanism enables the MCDS system to trace transactions on the on chip bus generated by one or a group of dedicated DMA internal sources.

DMA Trace Signal Generation for OCDS Level 1

For OCDS Level 1 purposes the DMA provides two 8 bit vectors.

15.8 Register Description

Figure 15-37 and **Table 15-12** show all registers associated with the DMA Controller Kernel. All DMA kernel register names described in this section are also referenced in other parts of the TC26x B-Step User's Manual by the module name prefix "DMA_".

DMA Register Index

The following index numbers are used:

- Index "x" is used to refer to the DMA sub-block (x = 0-1).
- Index "y" is used to refer to the hardware resource partitions (y = 0-3).
- Index "z" is used to refer to the DMA channel (z = 000-047).

DMA Register Classes

The registers fall into the following distinctive classes:

- Control registers that support clock control and the access protection to all registers and the DMARAM.
- Channel control registers are DMA channel control bits directly written into registers:
 - Channel reset
 - Hardware transfer request enable
 - Access pending set by either hardware triggers or software transfer requests
 - Transaction request lost status
 - Suspend enable
 - Suspend acknowledge
 - 2-bit hardware resource to partition channels across DMA partitions
- Active channel control read only registers store DMA channel control and status information for the current DMA move performed by the sub-block move engine.
- Move engine and error registers store control, status and read data for the current DMA transaction.

DMA Registers Overview

Clock Control Register	OCDS Trigger Select Register	Hardware Resource Partition	Sub-block Error Registers
CLC	OTSS	HRRz	EERx
Module Identification Register	Error Interrupt Register	Channel Suspend Registers	ERRSRx
ID	ERRINTR	SUSENRz	CLREx
Memory Control Register	Pattern Read Registers	SUSACRz	Sub-block Move Engine Registers
MEMCON	PRR0	Transaction State Register	ME _x SR
Access Enable Registers	PRR1	TSRz	ME _x 0R
ACCEN _y 0	Flow Control Timestamp	Transaction Control Set	ME _x 1R
ACCEN _y 1	TIME	RDCRCz	ME _x 2R
	Hardware Resource Mode	SDCRCz	ME _x 3R
	MODE _y	SADRz	ME _x 4R
		DADRz	ME _x 5R
		ADICRz	ME _x 6R
		CHCFGRz	ME _x 7R
		SHADRz	Sub-block Active Channel Registers
		CHCSRz	ME _x RDCRC
			ME _x SDCRC
			ME _x SADR
			ME _x DADR
			ME _x ADICR
			ME _x CHCR
			ME _x SHADR
			ME _x CHSR

MCA06175

Figure 15-37 DMA Kernel Registers (x = 0-1, y = 0-3, z = 000-047)

Direct Memory Access (DMA)**Table 15-11 Registers Address Space - DMA Registers**

Module	Base Address	End Address	Note
DMA	F001 0000 _H	F001 3FFF _H	DMA Registers

Direct Memory Access (DMA)
Table 15-12 Registers Overview - DMA Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
DMA_CLC	DMA Clock Control Register	0000 _H	U, SV	SV, E, P00, P01 ²⁾	3	Page 15-86
-	Reserved	0004 _H	BE	BE	-	-
DMA_ID	DMA Module Identification Register	0008 _H	U, SV	BE	-	Page 15-87
-	Reserved	000C _H - 001C _H	BE	BE	-	-
DMA_MEMCON	DMA Memory Control	0020 _H	SV, 32	SV, E, P00, P01, 32	3	Page 15-88
-	Reserved	0024 _H - 003C _H	BE	BE	-	-
DMA_ACCE Ny0	DMA Hardware Resource y Access Enable Register 0 (y = 0-3)	0040 _H + (y * 8 _H)	U, SV	SV, SE	3	Page 15-90
DMA_ACCE Ny1	DMA Hardware Resource y Access Enable Register 1 (y = 0-3)	0044 _H + (y * 8 _H)	U, SV	SV, SE	3	Page 15-91
-	Reserved	0060 _H - 011C _H	BE	BE	-	-
DMA_EERx	DMA Enable Error Register x (x = 0-1)	0120 _H + (x * 1000 _H)	U, SV	SV	3	Page 15-92
DMA_ERRS Rx	DMA Error Status Register x (x = 0-1)	0124 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-94
DMA_CLREx	DMA Clear Error Register x (x = 0-1)	0128 _H + (x * 1000 _H)	U, SV	SV	3	Page 15-97

Direct Memory Access (DMA)

Table 15-12 Registers Overview - DMA Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
-	Reserved (x = 0-1)	012C _H + (x * 1000 _H)	BE	BE	-	-
DMA_MExSR	DMA Move Engine x Status Register (x = 0-1)	0130 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-99
-	Reserved (x = 0-1)	0134 _H + (x * 1000 _H) - 013C _H + (x * 1000 _H)	BE	BE	-	-
DMA_MEx0R	DMA Move Engine x Read Register 0 (x = 0-1)	0140 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-100
DMA_MEx1R	DMA Move Engine x Read Register 1 (x = 0-1)	0144 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-101
DMA_MEx2R	DMA Move Engine x Read Register 2 (x = 0-1)	0148 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-102
DMA_MEx3R	DMA Move Engine x Read Register 3 (x = 0-1)	014C _H + (x * 1000 _H)	U, SV	BE	3	Page 15-103
DMA_MEx4R	DMA Move Engine x Read Register 4 (x = 0-1)	0150 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-104
DMA_MEx5R	DMA Move Engine x Read Register 5 (x = 0-1)	0154 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-105
DMA_MEx6R	DMA Move Engine x Read Register 6 (x = 0-1)	0158 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-106

Direct Memory Access (DMA)

Table 15-12 Registers Overview - DMA Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
DMA_MEx7R	DMA Move Engine x Read Register 7 (x = 0-1)	015C _H + (x * 1000 _H)	U, SV	BE	3	Page 15-10 7
-	Reserved (x = 0-1)	0160 _H + (x * 1000 _H) - 017C _H + (x * 1000 _H)	BE	BE	-	-
DMA_MExR DCRC	DMA Move Engine x Channel Read Data CRC Checksum Register (x = 0-1)	0180 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-12 9
DMA_MExSD CRC	DMA Move Engine x Channel Source and Destination Address CRC Checksum Register (x = 0-1)	0184 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-12 8
DMA_MExSA DR	DMA Move Engine x Source Address Register (x = 0-1)	0188 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-12 7
DMA_MExDA DR	DMA Move Engine x Channel Destination Address Register (x = 0-1)	018C _H + (x * 1000 _H)	U, SV	BE	3	Page 15-12 6
DMA_MExAD ICR	DMA Move Engine x Channel Address and Interrupt Control Register (x = 0-1)	0190 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-11 6

Direct Memory Access (DMA)
Table 15-12 Registers Overview - DMA Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
DMA_MExC HCR	DMA Move Engine x Channel Control Register (x = 0-1)	0194 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-11 2
DMA_MExSH ADR	DMA Move Engine x Channel Shadow Address Register (x = 0-1)	0198 _H + (x * 1000 _H)	U, SV	BE	3	Page 15-11 1
DMA_MExC HSR	DMA Move Engine x Channel Status Register (x = 0-1)	019C _H + (x * 1000 _H)	U, SV	BE	3	Page 15-10 8
-	Reserved (x = 0-1)	01A0 _H + (x * 1000 _H)- 11FC _H	BE	BE	-	-
DMA_OTSS	DMA OCDS Trigger Set Select Register	1200 _H	U, SV	SV	1	Page 15-13 0
DMA_ERRIN TR	DMA Error Interrupt Register	1204 _H	U, SV	SV	3	Page 15-13 3
DMA_PRR0	DMA Pattern Read Register 0	1208 _H	U, SV	SV	3	Page 15-13 4
DMA_PRR1	DMA Pattern Read Register 1	120C _H	U, SV	SV	3	Page 15-13 5
DMA_TIME	DMA Time Register	1210 _H	U, SV	BE	3	Page 15-13 6
-	Reserved	1214 _H - 12FC _H	BE	BE	-	-
DMA_MODE y	DMA Hardware Resource y Mode Register (y = 0-3)	1300 _H + (y * 4 _H)	U, SV	SV, SE, P00, P01	3	Page 15-13 7
-	Reserved	1310 _H - 17FC _H	BE	BE	-	-

Direct Memory Access (DMA)

Table 15-12 Registers Overview - DMA Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
DMA_HRRz	DMA Channel Hardware Resource Register z (z = 000-047)	1800 _H + (z * 0004 _H)	U, SV	SV, SE, P00, P01	3	Page 15-13 8
-	Reserved (ch = 48)	1800 _H + (ch * 0004 _H) - 19FC _H	BE	BE	-	-
DMA_SUSE NRz	DMA Channel Suspend Enable Register z (z = 000-047)	1A00 _H + (z * 0004 _H)	U, SV	SV, E, Py	1	Page 15-13 9
-	Reserved (ch = 48)	1A00 _H + (ch * 0004 _H) - 1BFC _H	BE	BE	-	-
DMA_SUSA CRz	DMA Channel Suspend Acknowledge Register z (z = 000-047)	1C00 _H + (z * 0004 _H)	U, SV	BE	1	Page 15-14 0
-	Reserved (ch = 48)	1C00 _H + (ch * 0004 _H) - 1DFC _H	BE	BE	-	-
DMA_TSRz	DMA Channel Transaction State Register z (z = 000-047)	1E00 _H + (z * 0004 _H)	U, SV	SV, Py	3	Page 15-14 1
-	Reserved (ch = 48)	1E00 _H + (ch * 0004 _H) - 1FFC _H	BE	BE	-	-
DMA_RDCCR Cz	DMA Channel z Read Data CRC (z = 000-047)	2000 _H + (z * 0020 _H)	U, SV	SV, Py	-	Page 15-14 4

Direct Memory Access (DMA)
Table 15-12 Registers Overview - DMA Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
DMA_SDCR Cz	DMA Channel z Source & Destination Address CRC (z = 000-047)	2004 _H + (z * 0020 _H)	U, SV	SV, Py	-	Page 15-14 5
DMA_SADRz	DMA Channel z Source Address (z = 000-047)	2008 _H + (z * 0020 _H)	U, SV	SV, Py	-	Page 15-14 6
DMA_DADRz	DMA Channel z Destination Address (z = 000-047)	200C _H + (z * 0020 _H)	U, SV	SV, Py	-	Page 15-14 7
DMA_ADICR z	DMA Channel z Address and Interrupt Control (z = 000-047)	2010 _H + (z * 0020 _H)	U, SV	SV, Py	-	Page 15-14 8
DMA_CHCF GRz	DMA Channel z Configuration (z = 000-047)	2014 _H + (z * 0020 _H)	U, SV	SV, Py	-	Page 15-15 0
DMA_SHAD Rz	DMA Channel z Shadow Address (z = 000-047)	2018 _H + (z * 0020 _H)	U, SV	SV, Py ³⁾	-	Page 15-15 2
DMA_CHCS Rz	DMA Channel z Control and Status (z = 000-047)	201C _H + (z * 0020 _H)	U, SV	SV, Py	-	Page 15-15 3
-	Reserved (ch = 48)	2000 _H + (ch * 0020 _H)- 3FFC _H	BE	BE	-	-

- 1) The absolute register address is calculated as follows:
Module Base Address ([Table 15-11](#)) + Offset Address (shown in this column)
Further, the following ranges for parameters x, y, and z are valid: x = 0-1, y = 0-3, z = 000-047.
- 2) Registers (addresses) protected by the default Hardware Resource Partition 0 Access Enable Registers ACCEN00 and ACCEN01 are marked P00 and P01.
- 3) In shadow address modes write access to DMA_SHADRz is controlled by the register bit DMA_ADICRz.SHCT[2]. If DMA_ADICRz.SHCT = '0001' or '0010' then Access Mode Write for DMA_SHADRz is BE. If DMA_ADICRz.SHCT = '0101' or '0110' then Access Mode Write for DMA_SHADRz is SV.

Direct Memory Access (DMA)

15.8.1 DMA General Module Control Registers

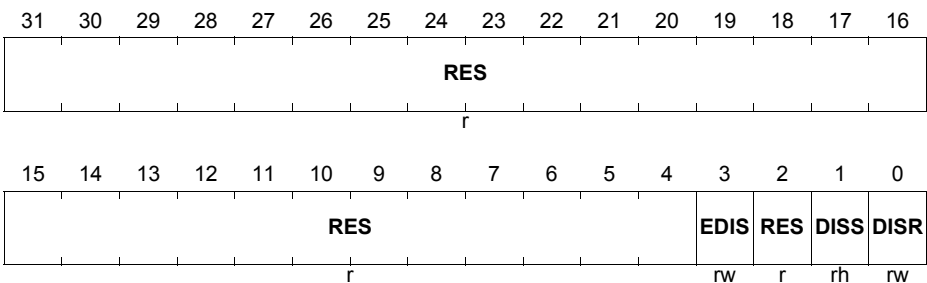
DMA Clock Control Register

The Clock Control Registers DMA_CLC controls the internal f_{DMA} clock signal and sleep mode in order to control the power consumption.

DMA_CLC

DMA Clock Control Register

 (0000_H)

 Reset Value: 0000 0008_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module
EDIS	3	rw	Sleep Mode Enable Control Used for module sleep mode control. 0 _B Sleep mode request is regarded. Module is enabled to go into sleep mode. 1 _B Sleep mode request is disregarded. Sleep mode cannot be entered on a request.
RES	2, [31:4]	r	Reserved Read as 0; must be written with 0.

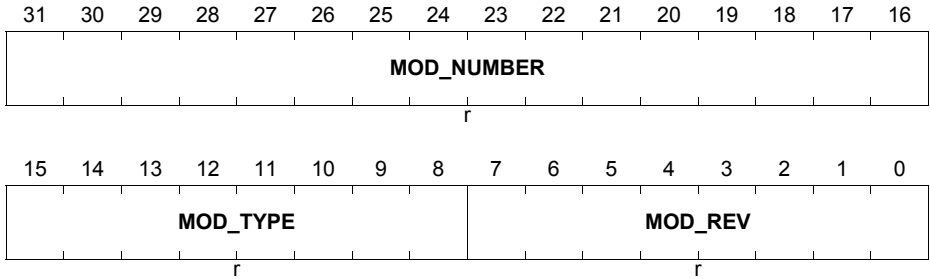
Note: After a hardware reset operation, the DMA module is enabled.

Note: The sleep mode does not modify any of the registers.

Direct Memory Access (DMA)

DMA Module Identification Register

DMA_ID
Module Identification Register (0008_H) **Reset Value: 0087 C003_H**



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number This bit field defines the module revision number.
MOD_TYPE	[15:8]	r	Module Type The bit field is set to C0 _H which defines the module as a 32-bit module.
MOD_NUMBER	[31:16]	r	Module Number Value This bit field defines a module identification number. The value for the DMA module is 0087 _H .

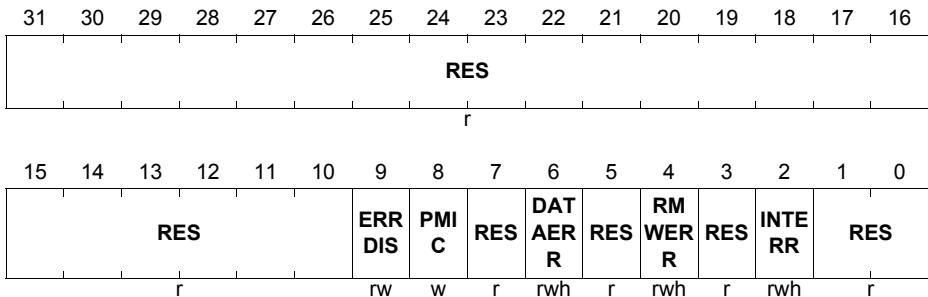
Direct Memory Access (DMA)

DMA Memory Control Register

Provides control of the memory integrity error checking, error signalling to the SMU and error injection for ECC logic test. The register is cleared by an Application Reset.

DMA_MEMCON

DMA Memory Control Register (0020_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
INTERR	2	rwh	Internal ECC Error Flag set by hardware when the DMA logic detects an ECC error while accessing the RAM. This bit is cleared by writing 0 _B but cannot be set by software. 0 _B No error has occurred 1 _B An error has been observed during a RAM access.
RMWERR	4	rwh	Internal Read Modify Write Error Flag set by hardware when the DMA logic detects an ECC error on the read phase of an internal RMW operation. This bit is cleared by writing 0 _B but cannot be set by software. 0 _B No error has occurred 1 _B An error has been observed during an internal RMW operation.

Direct Memory Access (DMA)

Field	Bits	Type	Description
DATAERR	6	rwh	SPB Data Phase ECC Error Flag set by hardware when the SPB interface detects an ECC error during the data phase of an incoming read transaction. This bit is cleared by writing 0 _B but cannot be set by software. 0 _B No error has occurred 1 _B An ECC error has been observed on an SPBI transaction addressed to the DMARAM
PMIC	8	w	Protection Bit for Memory Integrity Control Bit Will always return 0 _B when read 0 _B Bit Protection: Bit 9 remains unchanged after DMA_MEMCON write. 1 _B Bit 9 will be updated by the current write to DMA_MEMCON
ERRDIS	9	rw	ECC Error Disable When set SPB bus errors caused by ECC errors in data read from the SRAM will be disabled 0 _B Normal behavior. SPB bus error will occur on SRAM ECC errors. Default after reset 1 _B Test Mode. SPB bus errors will not be generated on an SRAM ECC error. This does not affect the generation of interrupts.
RES	[1:0],3,5,7,[31:10]	r	Reserved Read as 0; must be written with 0.

DMARAM Physical Implementation

If the number of DMA channels is small then the DMARAM may be implemented using logic. A logic implementation shall not generate ECC errors (MEMCON.INTERR = 0_B, MEMCON.RMWERR = 0_B and MEMCON.DATAERR = 0_B).

Direct Memory Access (DMA)
15.8.2 DMA Access Protection Registers

Write access to each DMA channel z transaction control set is controlled by the respective pair of Access Control Registers for the Hardware Resource assigned to the DMA channel z.

DMA Hardware Resource y Access Enable Register 0

The DMA Access Enable Register 0 controls read / write access for transactions with the on chip bus master TAG ID 000000_B to 011111_B. Each register bit enables one possible 6 bit TAG ID coding.

Mapping of TAG IDs to DMA_ACCEN0.ENn:

- EN0 -> TAG ID 000000_B
- EN1 -> TAG ID 000001_B
- ...
- EN31 -> TAG ID 011111_B

DMA_ACCENy0 (y = 0-3)
DMA Hardware Resource y Access Enable Register 0

(0040_H + y * 8_H)

Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN3	EN3	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN1	EN1	EN1	EN1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN1	EN1	EN1	EN1	EN1	EN1	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
5	4	3	2	1	0										
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables read / write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

The DMA Hardware Resource y Access Enable Register 0 is safe endinit protected.

Direct Memory Access (DMA)

DMA Hardware Resource y Access Enable Register 1

The DMA Access Enable Register y1 controls read / write access for transactions with the on chip bus master TAG ID 100000_B to 111111_B. Each register bit enables one possible 6 bit TAG ID coding.

Mapping of TAG IDs to DMA_ACCENx1.ENn:

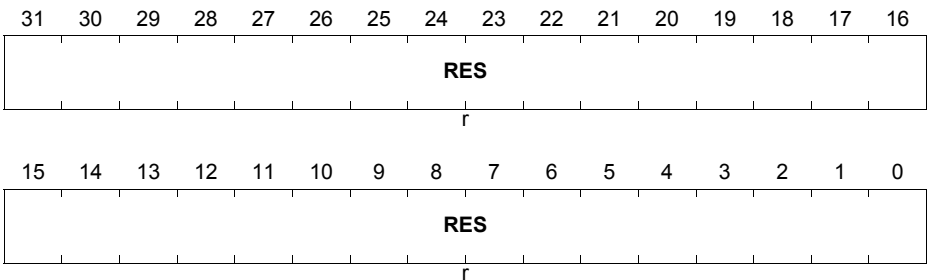
- EN0 -> TAG ID 100000_B
- EN1 -> TAG ID 100001_B
- ...
- EN31 -> TAG ID 111111_B

DMA_ACCENy1 (y = 0-3)

DMA Hardware Resource y Access Enable Register 1

$$(0044_H + y * 8_H)$$

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RES	[31:0]	r	Reserved Read as 0; must be written with 0.

The DMA Hardware Resource y Access Enable Register 1 is safe endinit protected.

15.8.3 DMA Sub-block Error Registers

DMA Enable Error Register

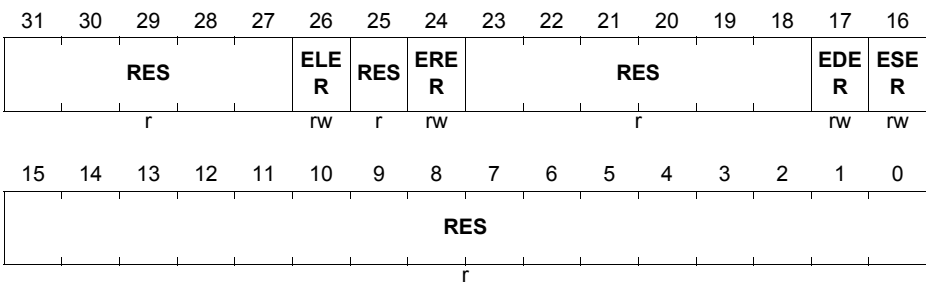
The Enable Error Register describes how the DMA controller reacts to errors. It enables the interrupts for Move Engine errors.

DMA_EERx (x = 0-1)

DMA Enable Error Register x

(0120_H + x * 1000_H)

Reset Value: 0503 0000_H



Field	Bits	Type	Description
RES	[15:0]	r	Reserved Read as 0; must be written with 0.
ESER	16	rw	Enable Move Engine x Source Error This bit enables the generation of a Move Engine x source error interrupt. 0 _B Move Engine x source error interrupt is disabled. 1 _B Move Engine x source error interrupt is enabled.
EDER	17	rw	Enable Move Engine x Destination Error This bit enables the generation of a Move Engine x destination error interrupt. 0 _B Move Engine x destination error interrupt is disabled. 1 _B Move Engine x destination error interrupt is enabled.
RES	[23:18]	r	Reserved Read as 0; must be written with 0.

Direct Memory Access (DMA)

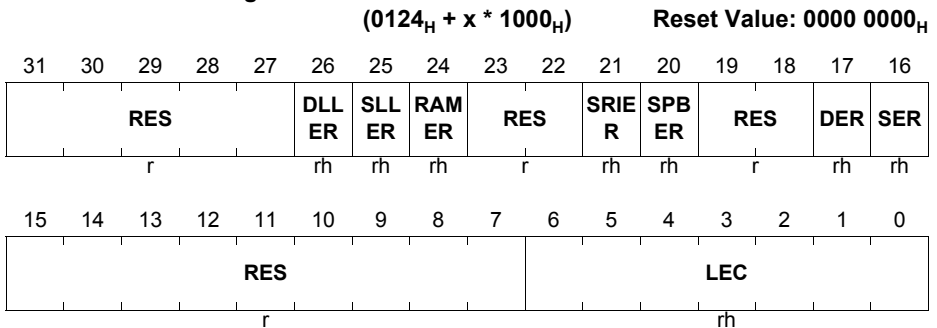
Field	Bits	Type	Description
ERER	24	rw	Enable Move Engine x RAM Error This bit enables the generation of a Move Engine x RAM error interrupt. 0 _B Move Engine x RAM error interrupt is disabled. 1 _B Move Engine x RAM error interrupt is enabled.
RES	25	r	Reserved Read as 0; must be written with 0.
ELER	26	rw	Enable Move Engine x DMA Linked List Error This bit enables the generation of a Move Engine x DMA Linked List error interrupt. 0 _B Move Engine x DMA Linked List error interrupt is disabled. 1 _B Move Engine x DMA Linked List error interrupt is enabled.
RES	[31:27]	r	Reserved Read as 0; must be written with 0.

Direct Memory Access (DMA)

DMA Error Status Register

The Error Status Register indicates if the DMA controller could not answer to a request because the previous request was not terminated (see [Section 15.4.4.1](#)).

It indicates that an SPB Bus access or an SRI Bus access has terminated with errors.

DMA_ERRSRx (x = 0-1)
DMA Error Status Register x


Field	Bits	Type	Description
LEC	[6:0]	rh	Move Engine x Last Error Channel This bit field indicates the channel number of the last channel of Move Engine x leading to an On Chip Bus error that has occurred. <i>Note: Last Error Channel DMA_ERRSRx.LEC is not set on RAM error DMA_ERRSRx.RAMER or Safe Linked List error DMA_ERRSRx.SLLER</i>
RES	[15:7]	r	Reserved Read as 0; must be written with 0.
SER	16	rh	Move Engine x Source Error This bit is set whenever a Move Engine x error occurred during a source (read) move of a DMA transfer, or a request could not be serviced due to the access protection. 0 _B No Move Engine x source error has occurred. 1 _B A Move Engine x source error has occurred.

Direct Memory Access (DMA)

Field	Bits	Type	Description
DER	17	rh	Move Engine x Destination Error This bit is set whenever a Move Engine x error occurred during a destination (write) move of a DMA transfer, or a request could not be serviced due to the access protection. 0 _B No Move Engine x destination error has occurred. 1 _B A Move Engine x destination error has occurred.
RES	[19:18]	r	Reserved Read as 0; must be written with 0.
SPBER	20	rh	Move Engine x SPB Bus Error This bit is set whenever a Move Engine x DMA Move that has been started by the DMA SPB master interface leads to an error on the SPB Bus. 0 _B No error occurred. 1 _B An error occurred on SPB Bus interface.
SRIER	21	rh	Move Engine x SRI Bus Error This bit is set whenever a Move Engine x DMA Move that has been started by the DMA SRI master interface leads to an error on the SRI Bus. 0 _B No error occurred. 1 _B An error occurred on SRI Bus interface.
RES	[23:22]	r	Reserved Read as 0; must be written with 0.
RAMER	24	rh	Move Engine x RAM Error This bit is set whenever a Move Engine x error occurred during the loading of a transaction control set from the DMARAM to the DMA sub-block x channel registers. 0 _B No error occurred. 1 _B An error occurred during the load of a transaction control set.
SLLER	25	rh	Move Engine x Safe Linked List Error This bit is set whenever a Move Engine x error occurred during the comparison of a SDCRC checksums during a safe linked list. 0 _B No error occurred. 1 _B An error occurred during the SDCRC checksum comparison.

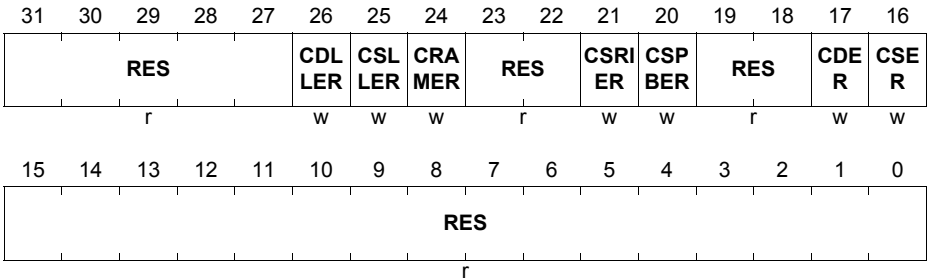
Direct Memory Access (DMA)

Field	Bits	Type	Description
DLLER	26	rh	<p>Move Engine x DMA Linked List Error</p> <p>This bit is set whenever a Move Engine x error occurred during the loading of a new transaction control set from anywhere in memory to overwrite the current transaction control set stored in the DMARAM.</p> <p>0_B No error occurred. 1_B An error occurred during the loading of a new transaction control set.</p> <p><i>Note: Error reporting includes all DMA Linked List operations included Accumulated Linked List, Safe Linked List and Conditional Linked List.</i></p>
RES	[31:27]	r	<p>Reserved</p> <p>Read as 0; must be written with 0.</p>

Direct Memory Access (DMA)
DMA Clear Error Register

The Clear Error contains bits to clear the corresponding Move Engine error flags.

DMA_CLRE_x (x = 0-1)

DMA Clear Error Register x
(0128_H + x * 1000_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RES	[15:0]	r	Reserved Read as 0; must be written with 0.
CSER	16	w	Clear Move Engine x Source Error 0 _B No action 1 _B Clear source error flag DMA_ERRSRx.SER.
CDER	17	w	Clear Move Engine x Destination Error 0 _B No action 1 _B Clear destination error flag DMA_ERRSRx.DER.
RES	[19:18]	r	Reserved Read as 0; must be written with 0.
CSPBER	20	w	Clear SPB Error 0 _B No action 1 _B Clear error flag DMA_ERRSRx.SPBER.
CSRIER	21	w	Clear SRI Error 0 _B No action 1 _B Clear error flag DMA_ERRSRx.SRIER.
RES	[23:22]	r	Reserved Read as 0; must be written with 0.

Direct Memory Access (DMA)

Field	Bits	Type	Description
CRAMER	24	w	Clear RAM Error 0 _B No action 1 _B Clear error flag DMA_ERRSRx.RAMER.
CSLLER	25	w	Clear SLL Error 0 _B No action 1 _B Clear error flag DMA_ERRSRx.SLLER.
CDLLER	26	w	Clear DLL Error 0 _B No action 1 _B Clear error flag DMA_ERRSRx.DLLER.
RES	[31:27]	r	Reserved Read as 0; must be written with 0.

Direct Memory Access (DMA)

15.8.4 DMA Sub-block Move Engine Registers

DMA Move Engine Status Register

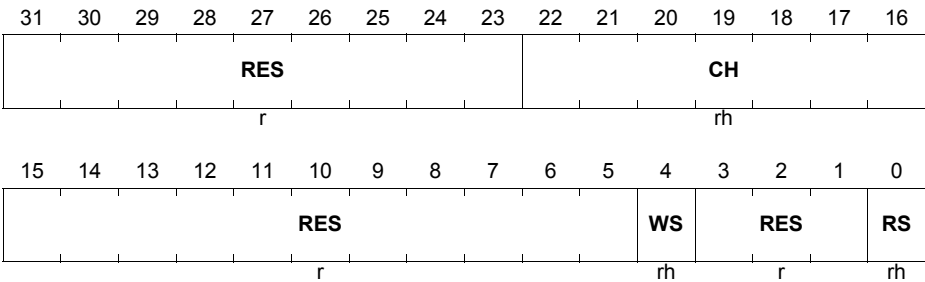
The Move Engine Status Register is a read-only register that holds status information about the transaction handled by the Move Engines.

DMA_MExSR ($x = 0-1$)

DMA Move Engine x Status Register

($0130_H + x * 1000_H$)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RS	0	rh	Move Engine x Read Status 0 _B Move Engine x is not performing a read. 1 _B Move Engine x is performing a read.
RES	[3:1]	r	Reserved Read as 0; must be written with 0.
WS	4	rh	Move Engine x Write Status 0 _B Move Engine x is not performing a write. 1 _B Move Engine x is performing a write.
RES	[15:5]	r	Reserved Read as 0; must be written with 0.
CH	[22:16]	rh	Active Channel z in Move Engine x This bit field indicates the number of the DMA Channel z currently being processed by the Move Engine x.
RES	[31:23]	r	Reserved Read as 0; must be written with 0.

Direct Memory Access (DMA)
DMA Move Engine x Read Register 0

The Move Engine x Read Register indicate the value that has just been read by Move Engine x.

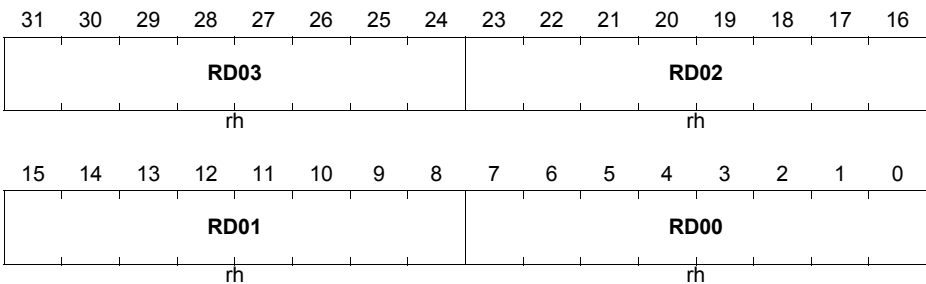
The value in the Read Register is compared to the bits in Move Engine Pattern Register depending on the pattern detection configuration and the channel data width.

DMA_MEx0R (x = 0-1)

DMA Move Engine x Read Register 0

(0140_H + x * 1000_H)

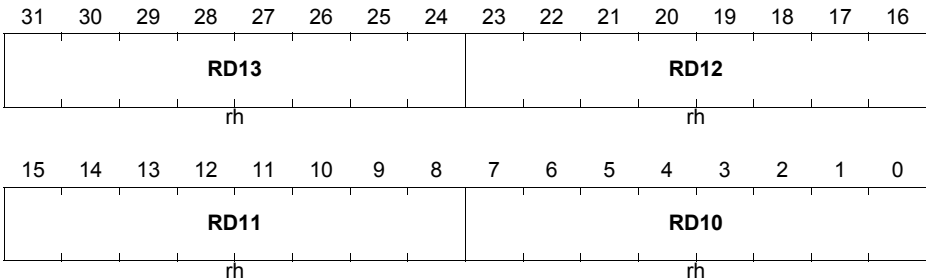
Reset Value: 0000 0000_H



Field	Bits	Type	Description
RD00, RD01, RD02, RD03	[7:0], [15:8], [23:16], [31:24]	rh	Read Value for Move Engine x Contains the 32-bit read data (four bytes RD0[3:0]) that is stored in the Move Engine x after each read move. The content of MExR is overwritten after each read move of a DMA channel belonging to DMA Sub-block x.

Direct Memory Access (DMA)
DMA Move Engine x Read Register 1

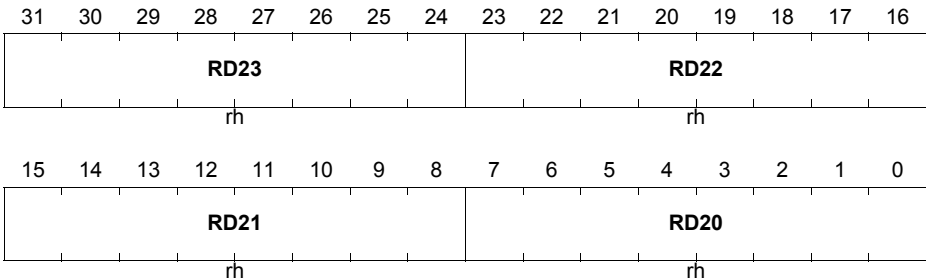
The Move Engine x Read Register indicate the value that has just been read by Move Engine x.

DMA_MEx1R (x = 0-1)
DMA Move Engine x Read Register 1
(0144_H + x * 1000_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
RD10, RD11, RD12, RD13	[7:0], [15:8], [23:16], [31:24]	rh	Read Value for Move Engine x Contains the 32-bit read data (four bytes RD1[3:0]) that is stored in the Move Engine x after each read move. The content of MExR is overwritten after each read move of a DMA channel belonging to DMA Sub-block x.

Direct Memory Access (DMA)
DMA Move Engine x Read Register 2

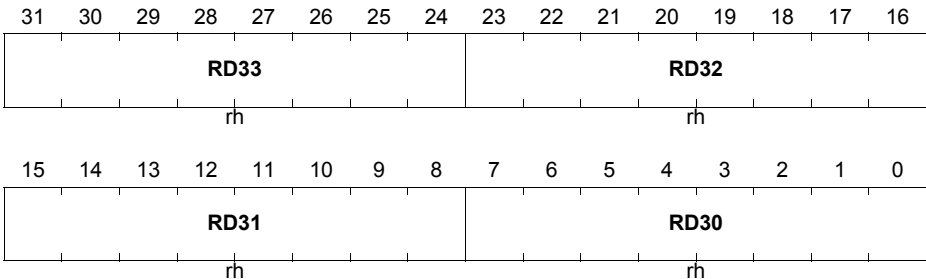
The Move Engine x Read Register indicate the value that has just been read by Move Engine x.

DMA_MEx2R (x = 0-1)
DMA Move Engine x Read Register 2
(0148_H + x * 1000_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RD20, RD21, RD22, RD23	[7:0], [15:8], [23:16], [31:24]	rh	Read Value for Move Engine x Contains the 32-bit read data (four bytes RD2[3:0]) that is stored in the Move Engine x after each read move. The content of MExR is overwritten after each read move of a DMA channel belonging to DMA sub-block x.

Direct Memory Access (DMA)
DMA Move Engine x Read Register 3

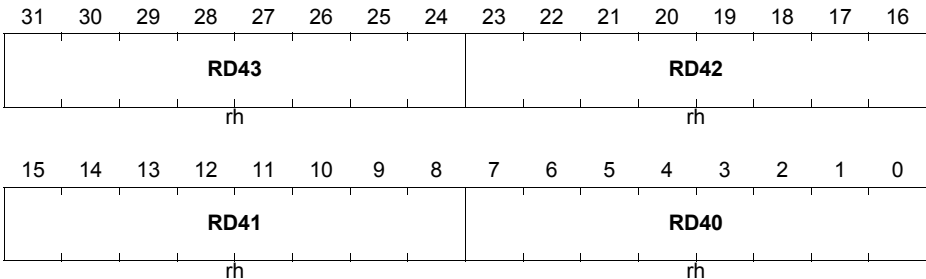
The Move Engine x Read Register indicate the value that has just been read by Move Engine x.

DMA_MEx3R (x = 0-1)
DMA Move Engine x Read Register 3
(014C_H + x * 1000_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RD30, RD31, RD32, RD33	[7:0], [15:8], [23:16], [31:24]	rh	Read Value for Move Engine x Contains the 32-bit read data (four bytes RD3[3:0]) that is stored in the Move Engine x after each read move. The content of MExR is overwritten after each read move of a DMA channel belonging to DMA sub-block x.

Direct Memory Access (DMA)
DMA Move Engine x Read Register 4

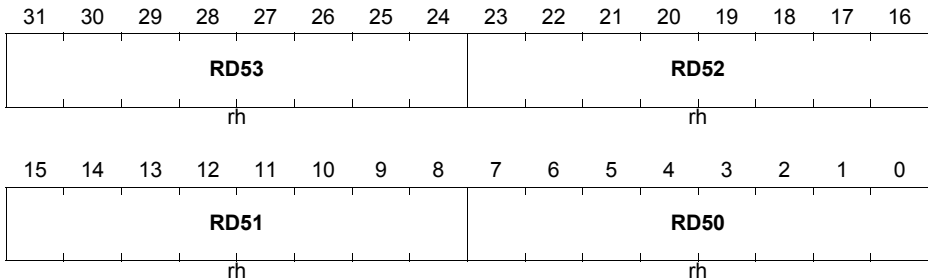
The Move Engine x Read Register indicate the value that has just been read by Move Engine x.

DMA_MEx4R (x = 0-1)
DMA Move Engine x Read Register 4
 $(0150_H + x * 1000_H)$
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RD40, RD41, RD42, RD43	[7:0], [15:8], [23:16], [31:24]	rh	Read Value for Move Engine x Contains the 32-bit read data (four bytes RD4[3:0]) that is stored in the Move Engine x after each read move. The content of MExR is overwritten after each read move of a DMA channel belonging to DMA sub-block x.

Direct Memory Access (DMA)
DMA Move Engine x Read Register 5

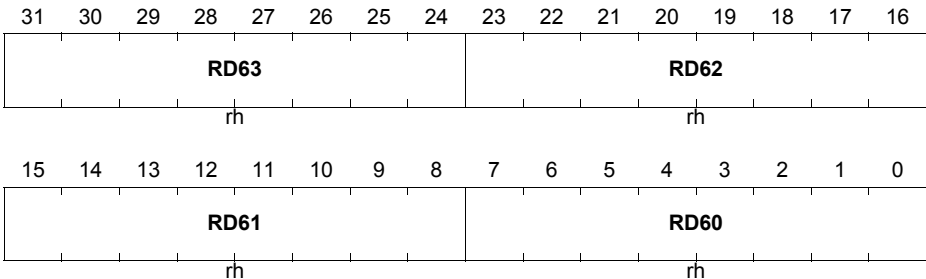
The Move Engine x Read Register indicate the value that has just been read by Move Engine x.

DMA_MEx5R (x = 0-1)
DMA Move Engine x Read Register 5
 $(0154_H + x * 1000_H)$
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RD50, RD51, RD52, RD53	[7:0], [15:8], [23:16], [31:24]	rh	Read Value for Move Engine x Contains the 32-bit read data (four bytes RD5[3:0]) that is stored in the Move Engine x after each read move. The content of MExR is overwritten after each read move of a DMA channel belonging to DMA sub-block x.

Direct Memory Access (DMA)
DMA Move Engine x Read Register 6

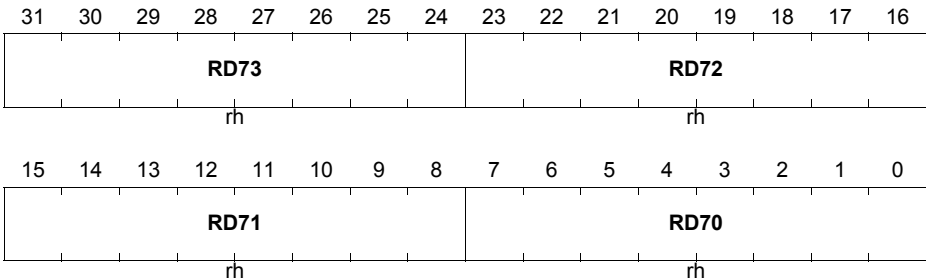
The Move Engine x Read Register indicate the value that has just been read by Move Engine x.

DMA_MEx6R (x = 0-1)
DMA Move Engine x Read Register 6
 $(0158_H + x * 1000_H)$
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RD60, RD61, RD62, RD63	[7:0], [15:8], [23:16], [31:24]	rh	Read Value for Move Engine x Contains the 32-bit read data (four bytes RD6[3:0]) that is stored in the Move Engine x after each read move. The content of MExR is overwritten after each read move of a DMA channel belonging to DMA sub-block x.

Direct Memory Access (DMA)
DMA Move Engine x Read Register 7

The Move Engine x Read Register indicate the value that has just been read by Move Engine x.

DMA_MEx7R (x = 0-1)
DMA Move Engine x Read Register 7
(015C_H + x * 1000_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RD70, RD71, RD72, RD73	[7:0], [15:8], [23:16], [31:24]	rh	Read Value for Move Engine x Contains the 32-bit read data (four bytes RD7[3:0]) that is stored in the Move Engine x after each read move. The content of MExR is overwritten after each read move of a DMA channel belonging to DMA sub-block x.

Direct Memory Access (DMA)

15.8.5 DMA Move Engine Active Channel Registers

DMA Move Engine Channel Status Register

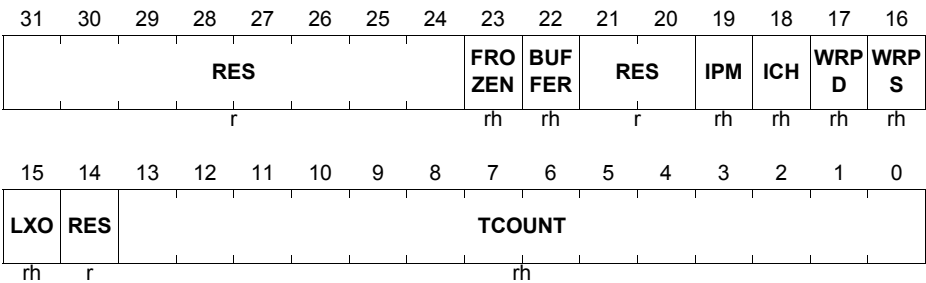
The read only DMA active channel Channel Status Register contains the current transfer count, pattern detection compare result and the status of traffic management interrupt triggers.

DMA_MExCHSR (x = 0-1)

DMA Move Engine x Channel Status Register

(019C_H + x * 1000_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TCOUNT	[13:0]	rh	<p>Transfer Count Status</p> <p>TCOUNT holds the actual value of the DMA transfer count for DMA channel z.</p> <p>TCOUNT is loaded with the value of CHCFGRz.TREL when TSRz.CH becomes set (and TCOUNT = 0).</p> <p>After each DMA transfer, TCOUNT is decremented by 1.</p>
RES	14	r	<p>Reserved</p> <p>Read as 0; must be written with 0.</p>

Direct Memory Access (DMA)

Field	Bits	Type	Description
LXO	15	rh	<p>Old Value of Pattern Detection</p> <p>This bit contains the compare result of a pattern compare operation when 8-bit or 16-bit data width is selected.</p> <p>8-bit data width: see Table 15-4 and Figure 15-27</p> <p>16-bit data width: see Table 15-5 and Figure 15-28</p> <p>0_B The corresponding pattern compare operation did not find a pattern match during the previous DMA read move</p> <p>1_B The corresponding pattern compare operation found a pattern match during the previous DMA read move</p>
WRPS	16	rh	<p>Wrap Source Buffer</p> <p>Indicates a wrap-around of source buffer.</p> <p>0_B No wrap-around occurred for channel z.</p> <p>1_B A wrap-around occurred for channel z.</p> <p>This bit is reset by software by writing a 1 to CHCSRz.CWRP or TSRz.RST.</p>
WRPD	17	rh	<p>Wrap Destination Buffer</p> <p>Indicates a wrap-around of destination buffer.</p> <p>0_B No wrap-around occurred for channel z.</p> <p>1_B A wrap-around occurred for channel z.</p> <p>This bit is reset by software by writing a 1 to CHCSRz.CWRP or TSRz.RST.</p>
ICH	18	rh	<p>Interrupt from Channel</p> <p>This bit indicates that channel z has raised an interrupt for TCOUNT = IRDV or if TCOUNT has been decremented (depending on ADICRz.INTCT[0]. This bit (and IPM) is reset by software when writing a 1 to ADICRz.CICH or by a channel reset (writing TSRz.RST = 1).</p> <p>0_B A channel interrupt has not been detected.</p> <p>1_B A channel interrupt has been detected.</p>

Direct Memory Access (DMA)

Field	Bits	Type	Description
IPM	19	rh	Pattern Detection from Channel This bit indicates that a pattern has been detected for channel z while the pattern detection has been enabled. This bit (and ICH) is reset by software when writing a 1 to ADICRz.CICH or by a channel reset (writing TSRz.RST = 1). 0 _B A pattern has not been detected. 1 _B A pattern has been detected.
RES	[21:20]	r	Reserved Read as 0; must be written with 0.
BUFFER	22	rh	DMA Double Buffering Active Buffer This bit is active during DMA double buffering and indicates which buffer is read or filled. 0 _B Buffer 0 read or filled by DMA. 1 _B Buffer 1 read or filled by DMA.
FROZEN	23	rh	DMA Double Buffering Frozen Buffer This bit is active during DMA double buffering and indicates that one of the double buffers is frozen and available for a cyclic software task. 0 _B Buffer is not frozen. 1 _B Buffer is frozen.
RES	[31:24]	r	Reserved Read as 0; must be written with 0.

Direct Memory Access (DMA)

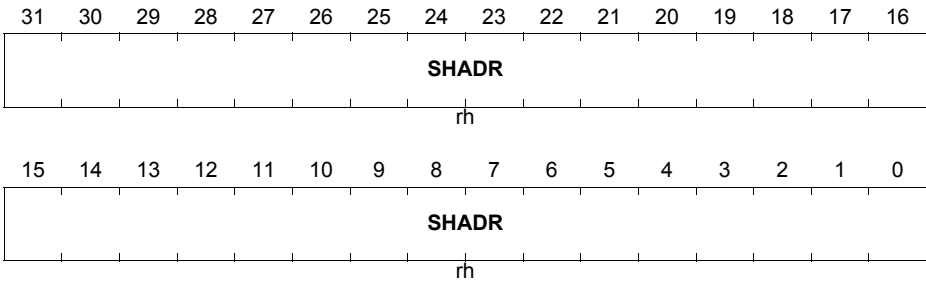
DMA Move Engine Channel Shadow Address Register

The read only DMA active channel Shadow Address Register holds the 32-bit shadow address.

DMA_MExSHADR (x = 0-1)

DMA Move Engine x Channel Shadow Address Register

(0198_H + x * 1000_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
SHADR	[31:0]	rh	<p>Shadowed Address</p> <p>This bit field holds the 32-bit shadow address of the active DMA channel. The function of the shadow address is set by the shadow control settings. See Table 15-15 for a description.</p>

Direct Memory Access (DMA)
DMA Move Engine Channel Control Register

The read only DMA active channel Channel Control Register contains the configuration and control bits.

DMA_MExCHCR (x = 0-1)
DMA Move Engine x Channel Control Register
 $(0194_H + x * 1000_H)$
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAprio		RES	PRSEL	RES	PATSEL			CHDW			CHMODE	RROAT	BLKM		
rh		r	rh	r	rh			rh			rh	rh	rh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		TREL													
r		rh													

Field	Bits	Type	Description
TREL	[13:0]	rh	Transfer Reload Value This bit field contains the number of DMA transfers for a DMA transaction of DMA channel z. This 14-bit transfer count value is loaded into MExCHSR.TCOUNT at the start of a DMA transaction (when TSRz.CH becomes set and CHCSRz.TCOUNT = 0). A write to CHCFGz.TREL during a running DMA transaction has no influence to the running DMA transaction. If CHCFGRz.TREL = 0 or if CHCFGRz.TREL = 1 then MExCHSR.TCOUNT will be loaded with 1 when a new transaction is started (at least one DMA transfer must be executed per DMA transaction).

Direct Memory Access (DMA)

Field	Bits	Type	Description
BLKM	[18:16]	rh	<p>Block Mode</p> <p>BLKM determines the number of DMA moves executed during one DMA transfer.</p> <p>000_B One DMA transfer has 1 DMA move 001_B One DMA transfer has 2 DMA moves 010_B One DMA transfer has 4 DMA moves 011_B One DMA transfer has 8 DMA moves 100_B One DMA transfer has 16 DMA moves 101_B One DMA transfer has 3 DMA moves 110_B One DMA transfer has 5 DMA moves 111_B One DMA transfer has 9 DMA moves</p> <p>See also Figure 15-12 on Page 15-23.</p>
RROAT	19	rh	<p>Reset Request Only After Transaction</p> <p>RROAT determines whether or not the TSRz.CH transfer request state flag is reset after each transfer.</p> <p>0_B TSRz.CH is reset after each transfer. A transfer request is required for each transfer. 1_B TSRz.CH is reset when CHSRz.TCOUNT = 0 after a transfer. One transfer request starts a complete DMA transaction</p>
CHMODE	20	rh	<p>Channel Operation Mode</p> <p>CHMODE determines the reset condition for control bit TSRz.HTRE of the DMA channel.</p> <p>0_B Single Mode operation is selected for DMA channel. After a transaction, DMA channel is disabled for further hardware requests (TSRz.HTRE is reset by hardware) TSRz.HTRE must be set again by software for starting a new transaction. 1_B Continuous Mode operation is selected for DMA channel z. After a transaction, bit TSRz.HTRE remains set</p>

Direct Memory Access (DMA)

Field	Bits	Type	Description
CHDW	[23:21]	rh	<p>Channel Data Width</p> <p>CHDW determines the data width for the read and write moves of DMA channel z.</p> <p>000_B 8-bit data width for moves selected <i>Note: Single Data Transfer Byte (SDTB)</i></p> <p>001_B 16-bit data width for moves selected <i>Note: Single Data Transfer Half-Word (SDTH)</i></p> <p>010_B 32-bit data width for moves selected <i>Note: Single Data Transfer Word (SDTW)</i></p> <p>011_B 64-bit data width transaction selected <i>Note: SRI-Bus: Single Data Transfer Double Word (SDTD)</i> <i>Note: SPB-Bus: not supported.</i></p> <p>100_B 128-bit data width transaction selected <i>Note: SRI-Bus: Block Transfer Request - 2 Transfers (BTR2)</i> <i>Note: SPB-Bus: not supported.</i></p> <p>101_B 256-bit data width transaction selected <i>Note: SRI-Bus: Block Transfer Request - 4 Transfers (BTR4)</i> <i>Note: SPB-Bus: not supported.</i></p> <p>110_B Reserved</p> <p>111_B Reserved</p>

Direct Memory Access (DMA)

Field	Bits	Type	Description
PATSEL	[26:24]	rh	<p>Pattern Select</p> <p>This bit field selects the mode of the pattern detection logic. Depending on the channel data width, PATSEL[1:0] selects different pattern detection configurations.</p> <p>If pattern detection is enabled (PATSEL[1:0] not equal 00_B), the pattern detection interrupt line will be activated on the selected pattern match.</p> <p>PATSEL[2] selects the pattern read register.</p> <p>0_B Pattern Read Register 0 1_B Pattern Read Register 1</p> <p>8-bit channel data width (CHDW = 000_B): Selected pattern detection configuration see Table 15-4 on Page 15-55.</p> <p>16-bit channel data width (CHDW = 001_B): Selected pattern detection configuration see Table 15-5 on Page 15-57.</p> <p>32-bit channel data width (CHDW = 010_B): Selected pattern detection configuration see Table 15-6 on Page 15-60.</p>
PRSEL	28	rh	<p>Peripheral Request Select</p> <p>This bit field controls the hardware request input multiplexer of DMA channel z-1 (see Figure 15-6 on Page 15-13).</p> <p>0_B Hardware trigger selected 1_B Daisy chain selected</p>
DMAPRIO	[31:30]	rh	<p>DMA Priority</p> <p>This bit determines the DMA the request priority that is used when a move operation related to channel z is requesting SPB-Bus. This bit has no effect in channel prioritization inside the Move Engine x in.</p> <p>00_B Low priority selected 01_B Medium priority selected 10_B Medium priority selected 11_B High priority selected</p>
RES	[15:14], 27, 29	r	<p>Reserved</p> <p>Read as 0; must be written with 0.</p>

Direct Memory Access (DMA)

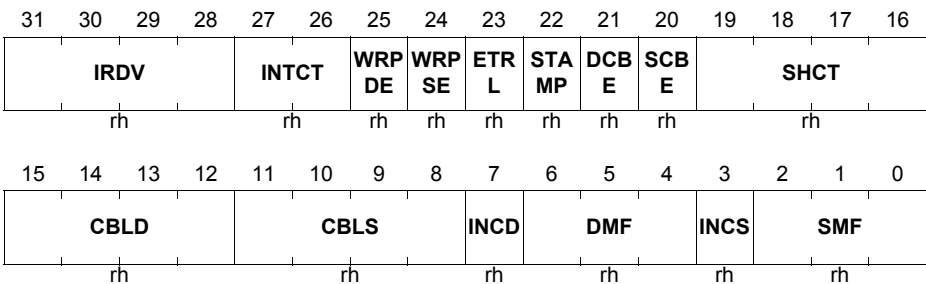
DMA Move Engine Channel Address and Interrupt Control Register

The read only DMA active channel Address and Interrupt Control Register controls how source and destination addresses are updated after a DMA move. Furthermore, it determines whether or not a source or destination address register update is shadowed. The interrupt control bits enable the generation of DMA traffic interrupt triggers.

DMA_MExADICR (x = 0-1)

DMA Move Engine x Channel Address and Interrupt Control Register

(0190_H + x * 1000_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description																
SMF	[2:0]	rh	<p>Source Address Modification Factor</p> <p>This bit field and the data width as defined in MExCHCR.CHDW determine an address offset value by which the source address is modified after each DMA move. See also Table 15-13.</p> <table style="width:100%; border-collapse: collapse;"> <tr><td style="width: 50px;">000_B</td><td>Address offset is 1 x MExCHCR.CHDW</td></tr> <tr><td>001_B</td><td>Address offset is 2 x MExCHCR.CHDW</td></tr> <tr><td>010_B</td><td>Address offset is 4 x MExCHCR.CHDW</td></tr> <tr><td>011_B</td><td>Address offset is 8 x MExCHCR.CHDW</td></tr> <tr><td>100_B</td><td>Address offset is 16 x MExCHCR.CHDW</td></tr> <tr><td>101_B</td><td>Address offset is 32 x MExCHCR.CHDW</td></tr> <tr><td>110_B</td><td>Address offset is 64 x MExCHCR.CHDW</td></tr> <tr><td>111_B</td><td>Address offset is 128 x MExCHCR.CHDW</td></tr> </table>	000 _B	Address offset is 1 x MExCHCR.CHDW	001 _B	Address offset is 2 x MExCHCR.CHDW	010 _B	Address offset is 4 x MExCHCR.CHDW	011 _B	Address offset is 8 x MExCHCR.CHDW	100 _B	Address offset is 16 x MExCHCR.CHDW	101 _B	Address offset is 32 x MExCHCR.CHDW	110 _B	Address offset is 64 x MExCHCR.CHDW	111 _B	Address offset is 128 x MExCHCR.CHDW
000 _B	Address offset is 1 x MExCHCR.CHDW																		
001 _B	Address offset is 2 x MExCHCR.CHDW																		
010 _B	Address offset is 4 x MExCHCR.CHDW																		
011 _B	Address offset is 8 x MExCHCR.CHDW																		
100 _B	Address offset is 16 x MExCHCR.CHDW																		
101 _B	Address offset is 32 x MExCHCR.CHDW																		
110 _B	Address offset is 64 x MExCHCR.CHDW																		
111 _B	Address offset is 128 x MExCHCR.CHDW																		

Direct Memory Access (DMA)

Field	Bits	Type	Description
INCS	3	rh	<p>Increment of Source Address</p> <p>This bit determines whether the address offset as selected by SMF will be added to or subtracted from the source address after each DMA move. The source address is not modified if CBL5 = 0000_B.</p> <p>0_B Address offset will be subtracted 1_B Address offset will be added.</p>
DMF	[6:4]	rh	<p>Destination Address Modification Factor</p> <p>This bit field and the data width as defined in MEXCHCR.CHDW determines an address offset value by which the destination address is modified after each DMA move. The destination address is not modified if CBLD = 0000_B. See also Table 15-13.</p> <p>000_B Address offset is 1 x MEXCHCR.CHDW 001_B Address offset is 2 x MEXCHCR.CHDW 010_B Address offset is 4 x MEXCHCR.CHDW 011_B Address offset is 8 x MEXCHCR.CHDW 100_B Address offset is 16 x MEXCHCR.CHDW 101_B Address offset is 32 x MEXCHCR.CHDW 110_B Address offset is 64 x MEXCHCR.CHDW 111_B Address offset is 128 x MEXCHCR.CHDW</p>
INCD	7	rh	<p>Increment of Destination Address</p> <p>This bit determines whether the address offset as selected by DMF will be added to or subtracted from the destination address after each DMA move. The destination address is not modified if CBLD = 0000_B.</p> <p>0_B Address offset will be subtracted 1_B Address offset will be added</p>

Direct Memory Access (DMA)

Field	Bits	Type	Description
CBLS	[11:8]	rh	<p>Circular Buffer Length Source</p> <p>This bit field determines which part of the 32-bit source address register remains unchanged and is not updated after a DMA move operation (see also Section 15.4.3.8).</p> <p>Therefore, CBLS also determines the size of the circular source buffer.</p> <p>0000_B Source address SADR[31:0] is not updated 0001_B Source address SADR[31:1] is not updated 0010_B Source address SADR[31:2] is not updated 0011_B Source address SADR[31:3] is not updated ..._B ... 1110_B Source address SADR[31:14] is not updated 1111_B Source address SADR[31:15] is not updated</p>
CBLD	[15:12]	rh	<p>Circular Buffer Length Destination</p> <p>This bit field determines which part of the 32-bit destination address register remains unchanged and is not updated after a DMA move operation (see also Page 15-25). Therefore, CBLD also determines the size of the circular destination buffer.</p> <p>0000_B Destination address DADR[31:0] is not updated 0001_B Destination address DADR[31:1] is not updated 0010_B Destination address DADR[31:2] is not updated 0011_B Destination address DADR[31:3] is not updated ..._B ... 1110_B Destination address DADR[31:14] is not updated 1111_B Destination address DADR[31:15] is not updated</p>
SHCT	[19:16]	rh	<p>Shadow Control</p> <p>This bit field determines the function of the shadow address register. See Table 15-15.</p>
SCBE	20	rh	<p>Source Circular Buffer Enable</p> <p>0_B Source circular buffer disabled 1_B Source circular buffer enabled</p>

Direct Memory Access (DMA)

Field	Bits	Type	Description
DCBE	21	rh	Destination Circular Buffer Enable 0 _B Destination circular buffer disabled 1 _B Destination circular buffer enabled
STAMP	22	rh	Time Stamp This bit enables the appendage of a timestamp after the end of the last DMA Move during a DMA transaction. 0 _B No action. 1 _B Enable the appendage of a 32-bit timestamp.
ETRL	23	rh	Enable Transaction Request Lost Interrupt This bit enables the generation of an interrupt when TSRz.TRLz is set. 0 _B The interrupt generation for a request lost event for channel z is disabled. 1 _B The interrupt generation for a request lost event for channel z is enabled.
WRPSE	24	rh	Wrap Source Enable 0 _B Wrap source buffer interrupt trigger disabled 1 _B Wrap source buffer interrupt trigger enabled
WRPDE	25	rh	Wrap Destination Enable 0 _B Wrap destination buffer interrupt trigger disabled 1 _B Wrap destination buffer interrupt trigger enabled
INTCT	[27:26]	rh	Interrupt Control 00 _B No interrupt trigger will be generated on changing the TCOUNT value. The bit CHSRz.ICH is set when TCOUNT equals IRDV. 01 _B No interrupt trigger will be generated on changing the TCOUNT value. The bit CHSRz.ICH is set when TCOUNT is decremented 10 _B Interrupt trigger is generated and bit CHSRz.ICH is set on changing the TCOUNT value and TCOUNT equals IRDV 11 _B Interrupt trigger is generated and bit CHSRz.ICH is set each time TCOUNT is decremented <i>Note: see Figure 15-31.</i> <i>Note: If DMA_CHCFGRz.PRSEL = 1_B for the next lower priority channel then the channel transfer trigger interrupt is disabled.</i>

Direct Memory Access (DMA)

Field	Bits	Type	Description
IRDV	[31:28]	rh	Interrupt Raise Detect Value Defines the Threshold Limit of CHSRz.TCOUNT for which a channel interrupt trigger will be raised.

Direct Memory Access (DMA)
Address Offset Calculation Tables

Table 15-13 and **Table 15-14** show the offset values that are added or subtracted to/from a source or destination address register after a DMA move.

Bit field SMF and bit INCS determine the offset value for the source address.

Bit field DMF and bit INCD determine the offset value for the destination address.

Table 15-13 Address Offset Calculation Table (8-bit, 16-bit and 32-bit)

CHCFGRz.CHDW = 000 _B (8-bit Data Width)			CHCFGRz.CHDW = 001 _B (16-bit Data Width)			CHCFGRz.CHDW = 010 _B (32-bit Data Width)		
SMF DMF	INCS INCD	Address Offset	SMF DMF	INCS INCD	Address Offset	SMF DMF	INCS INCD	Address Offset
000 _B	0	-1	000 _B	0	-2	000 _B	0	-4
	1	+1		1	+2		1	+4
001 _B	0	-2	001 _B	0	-4	001 _B	0	-8
	1	+2		1	+4		1	+8
010 _B	0	-4	010 _B	0	-8	010 _B	0	-16
	1	+4		1	+8		1	+16
011 _B	0	-8	011 _B	0	-16	011 _B	0	-32
	1	+8		1	+16		1	+32
100 _B	0	-16	100 _B	0	-32	100 _B	0	-64
	1	+16		1	+32		1	+64
101 _B	0	-32	101 _B	0	-64	101 _B	0	-128
	1	+32		1	+64		1	+128
110 _B	0	-64	110 _B	0	-128	110 _B	0	-256
	1	+64		1	+128		1	+256
111 _B	0	-128	111 _B	0	-256	111 _B	0	-512
	1	+128		1	+256		1	+512

Direct Memory Access (DMA)
Table 15-14 Address Offset Calculation Table (64-bit, 128-bit and 256-bit)

CHCFGRz.CHDW = 011 _B (64-bit Data Width)			CHCFGRz.CHDW = 100 _B (128-bit Data Width)			CHCFGRz.CHDW = 101 _B (256-bit Data Width)		
SMF DMF	INCS INCD	Address Offset	SMF DMF	INCS INCD	Address Offset	SMF DMF	INCS INCD	Address Offset
000 _B	0	-8	000 _B	0	-16	000 _B	0	-32
	1	+8		1	+16		1	+32
001 _B	0	-16	001 _B	0	-32	001 _B	0	-64
	1	+16		1	+32		1	+64
010 _B	0	-32	010 _B	0	-64	010 _B	0	-128
	1	+32		1	+64		1	+128
011 _B	0	-64	011 _B	0	-128	011 _B	0	-256
	1	+64		1	+128		1	+256
100 _B	0	-128	100 _B	0	-256	100 _B	0	-512
	1	+128		1	+256		1	+512
101 _B	0	-256	101 _B	0	-512	101 _B	0	-1024
	1	+256		1	+512		1	+1024
110 _B	0	-512	110 _B	0	-1024	110 _B	0	-2048
	1	+512		1	+1024		1	+2048
111 _B	0	-1024	111 _B	0	-2048	111 _B	0	-4096
	1	+1024		1	+2048		1	+4096

Direct Memory Access (DMA)

Shadow Control Table

Table 15-15 defines the functionality of the shadow address register control bits.

Table 15-15 Shadow Control Table

SHCT	Description
0000 _B	Source and destination registers written directly. <i>Note: Shadow address register is not used and set to 00000000_H.</i>
0001 _B	Source Address Buffering (Read Only) Shadow address used for source buffering. When writing to SADRz, the address is buffered in SHADRz and transferred to SADRz with the start of the next DMA transaction. <i>Note: Shadow address register is read only and is automatically set to 00000000_H when the shadow transfer takes place (equal to AudoNG).</i>
0010 _B	Destination Address Buffering (Read Only) Shadow address used for destination buffering. When writing to DADRz, the address is buffered in SHADRz and transferred to DADRz with the start of the next DMA transaction. <i>Note: Shadow address register is read only and is automatically set to 00000000_H when the shadow transfer takes place (equal to AudoNG).</i>
0011 _B	Reserved. <i>Note: Shadow address register is not used and set to 00000000_H.</i>
0100 _B	Reserved. <i>Note: Shadow address register is not used and set to 00000000_H.</i>
0101 _B	Source Address Buffering (Direct Write) Shadow address used for source buffering. Shadow address register can be read and can be directly written. The value stored in SHADRz is not automatically modified when the shadow transfer takes place.
0110 _B	Destination Address Buffering (Direct Write) Shadow address used for destination buffering. Shadow address register can be read and can be directly written. The value stored in SHADRz is not automatically modified when the shadow transfer takes place.
0111 _B	Reserved. <i>Note: Shadow address register is not used and set to 00000000_H.</i>
1000 _B	Double Source Buffering Software Switch Only Shadow address used for double buffering

Direct Memory Access (DMA)

Table 15-15 Shadow Control Table (cont'd)

SHCT	Description
1001 _B	Double Source Buffering Automatic Hardware and Software Switch Shadow address used for double buffering
1010 _B	Destination Double Destination Buffering Software Switch Only Shadow address used for double buffering
1011 _B	Destination Double Destination Buffering Automatic Hardware and Software Switch Shadow address used for double buffering
1100 _B	DMA Linked List The DMA controller reads a DMA channel transaction control set and overwrites 8 X words in the corresponding DMARAM channel z. <i>Note: The SDCRC and RDCRC checksums are unique for each DMA transaction in the DMA Linked List. RDCRCRz and SDCRCRz are overwritten after each DMA transaction in the DMA Linked List.</i>
1101 _B	Accumulated Linked List The DMA controller reads a DMA channel transaction control set and overwrites 6 X words in the corresponding DMARAM channel z. <i>Note: The SDCRC and RDCRC checksums are accumulated across all DMA transactions of the Accumulated Linked List. RDCRCRz and SDCRCRz are not overwritten</i>
1110 _B	Safe Linked List The DMA controller reads a DMA channel transaction control set. The Linked List only proceeds with the next DMA transaction if the existing SDCRC checksum matches the expected SDCRC checksum in the loaded from the new DMA transaction control set. <i>Note: The SDCRC and RDCRC checksums are accumulated across all DMA transactions of the Safe Linked List. RDCRCRz is not overwritten.</i>
1111 _B	Conditional Linked List Shadow address register (MExSHADR) and source and destination address CRC register (MExSDCRC) are used as address pointers to a Linked List. The selection of the address pointer is determined by DMA channel pattern detection conditions. <i>Note: Calculation of SDCRC checksums is not supported.</i>

Direct Memory Access (DMA)**Shadow Address used for Source or Destination Buffering**

The Shadow Address Register holds the shadowed source or destination address before it is written into the source or destination address register.

Shadow Address used for DMA Double Buffering

The Shadow Address Register holds a second source or destination address used to store data in a second source or destination location.

Shadow Address used for DMA Linked Lists

The Shadow Address Register holds an address pointer to the next DMA transaction control set.

Direct Memory Access (DMA)

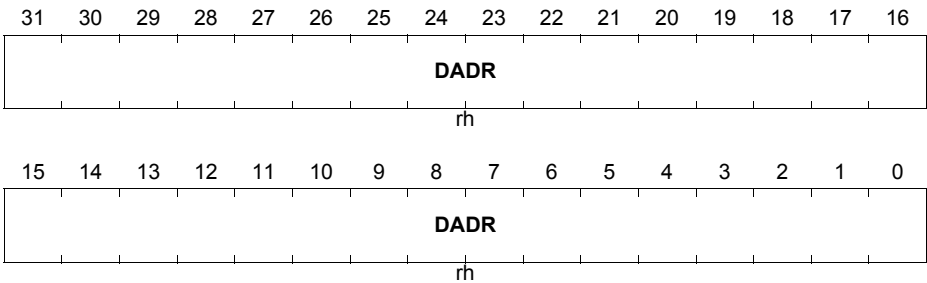
DMA Move Engine Channel Destination Address Register

The read only DMA active channel Destination Address Register holds the 32-bit destination address. If a DMA channel is active, MExDADR is updated continuously (if programmed) and shows the actual destination address that is used for write moves within DMA transfers.

DMA_MExDADR (x = 0-1)

DMA Move Engine x Channel Destination Address Register x

(018C_H + x * 1000_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
DADR	[31:0]	rh	Destination Address This bit field holds the actual 32-bit destination address of the active DMA channel that is used for write moves.

Direct Memory Access (DMA)

DMA Move Engine x Channel Source Address Register

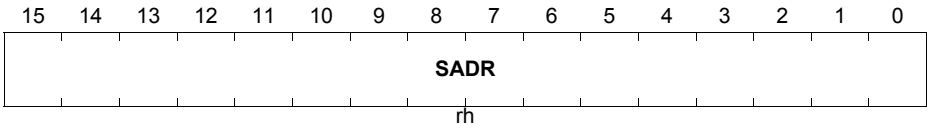
The read only DMA active channel Source Address Register holds the 32-bit source address. If a DMA channel is active, MExSADR is updated continuously (if programmed) and shows the actual source address that is used for read moves within DMA transfers.

DMA_MExSADR (x = 0-1)

DMA Move Engine x Channel Source Address Register

$$(0188_H + x * 1000_H)$$

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SADR	[31:0]	rh	<p>Source Start Address</p> <p>This bit field holds the actual 32-bit source address of the active DMA channel that is used for read moves.</p>

Direct Memory Access (DMA)

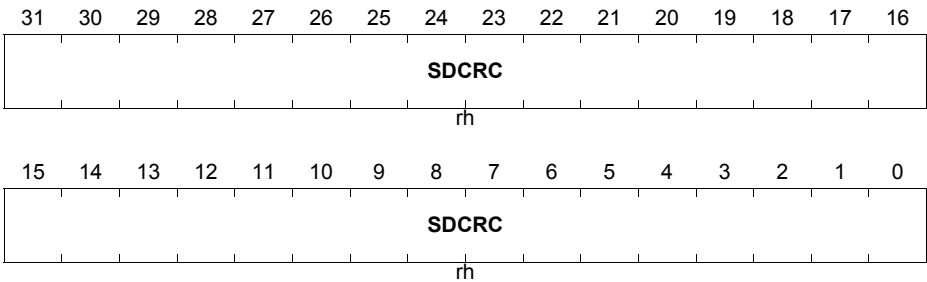
DMA Channel Source and Destination Address CRC Register

The read only DMA active channel Source and Destination Address CRC Register will store one polynomial calculation during each DMA read and each DMA write move provided there is no retry or error condition flagged.

DMA_MExSDCRC (x = 0-1)

DMA Move Engine x Channel Source and Destination Address CRC Register

(0184_H + x * 1000_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
SDCRC	[31:0]	rh	Source and Destination Address CRC This bit field contains the working CRC32 ethernet polynomial checksum for the source and destination address.

Direct Memory Access (DMA)

DMA Move Engine Channel Read Data CRC Register

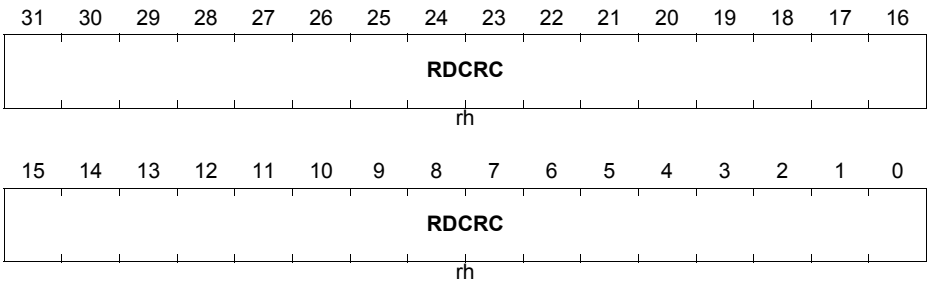
The read only DMA active channel Read Data CRC Register will store one polynomial calculation during each DMA read move provided that there is no retry or error condition flagged. In order to start a CRC32 sequence the MExRDCRC must be initialized (e.g. written with 00000000_H or with a desired start value) and an DMA transaction must be set up (start address, length, etc.).

DMA_MExRDCRC (x = 0-1)

DMA Move Engine x Channel Read Data CRC Register

$$(0180_H + x * 1000_H)$$

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RDCRC	[31:0]	rh	Read Data CRC This bit field contains the working CRC32 ethernet polynomial checksum for read data.

15.8.6 DMA OCDS Registers

DMA OCDS Trigger Set Select Register

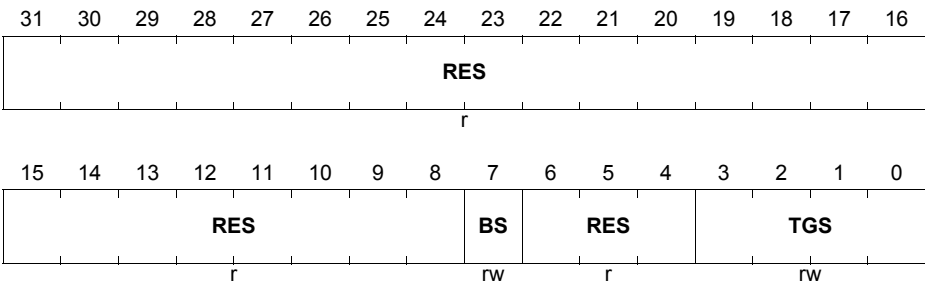
The selection of the OCDS Trigger Bus (OTGB) is controlled by the OTSS register.

The OTSS register is cleared by Debug Reset and it is cleared by each System Reset when OCDS is disabled. The OTSS Register is not touched by the System Reset when OCDS is enabled.

Write access requires OCDS to be enabled and Supervisor mode.

DMA_OTSS

DMA OCDS Trigger Set Select (1200_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
TGS	[3:0]	rw	Trigger Set (Table 15-16) for OTGB0/1 0 _D No Trigger Set selected 1 _D Trigger Set 1 2 _D Trigger Set 2 8 _D Trigger Set 8 9 _D Trigger Set 9 10 _D Trigger Set 10 others reserved
BS	7	rw	OTGB0/1 Bus Select 0 _B Trigger Set is output on OTGB0 1 _B Trigger Set is output on OTGB1
RES	[6:4], [31:8]	r	Reserved Read as 0; must be written with 0.

Direct Memory Access (DMA)

OCDS Trigger Bus Interface

Table 15-16 lists the Trigger Sets, which are selected with the DMA_OTSS register for the 16 bit OCDS Trigger Bus (OTGB0/1) interface. Only one Trigger Set can be output at a time either on OTGB0 or on OTGB1.

The OCDS Trigger Bus has no dependency on the OCDS enabled state.

Table 15-16 DMA Trigger Sets

Index	Description	Details
0	No Trigger Set selected	
1	Channels (TS16_PF) Active channels	Table 15-17
2	Channels (TS16_ERR) Error channel number and flags	Table 15-18
8	Transaction Request State (TS16_C15) Transaction Request State Channel [15:0]	
9	Transaction Request State (TS16_C31) Transaction Request State Channel [31:16]	
10	Transaction Request State (TS16_C47) Transaction Request State Channel [47:32]	
Other	Reserved. No Trigger Set selected	

Performance Trigger Set
Table 15-17 TS16_PF Trigger Set Channels

Bits	Name	Description
[6:0]	CH0	Channel number active in move engine 0
7	ME0	Move engine 0 idle/active 0_B Move engine 0 is idle 1_B Move engine 0 is active
[14:8]	CH1	Channel number active in move engine 1
15	ME1	Move engine 1 active 0_B Move engine 1 is idle 1_B Move engine 1 is active

The performance trigger set continuously monitors the activity in the move engines. While the move engine is servicing a DMA request from DMA channel z the move engine idle/active bit TS16_PF.MEx is set and the TS16_PF.CHx bit field is set to the DMA channel number z. When the move engine is idle the idle/active bit TS16_PF.MEx goes low. The TS16_PF.CHx bit field retains the value of the last active DMA channel.

Error Trigger Set
Table 15-18 TS16_ERR Trigger Set Channels

Bits	Name	Description
[6:0]	LEC	Last error channel number
[11:7]		Reserved
12	ME0SE	Source Error
13	ME0DE	Destination Error
14	ME1SE	Source Error
15	ME1DE	Destination Error

If a move engine x source or destination error occurs then the respective CH.MExSE or TS16_CH.MExDE gets set. The TS16_CH.LEC returns the last error channel. If both move engines generate an error condition in the same clock cycle then the TS16_CH.LEC will record the last error channel for move engine 0.

The TS16_ERR trigger set retains its value until there is a new move engine source or destination error. The MCDS must be able to sample the source or destination occurrence and the DMA error channel.

Direct Memory Access (DMA)

DMA Error Interrupt Set Register

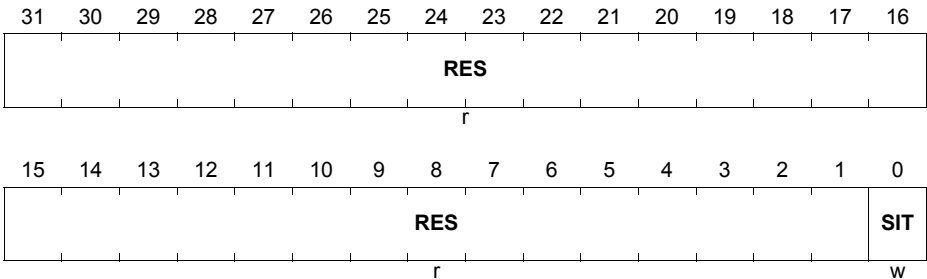
The Error Interrupt Set Register allows the DMA error interrupt service request to be activated by software.

DMA_ERRINTR

DMA Error Interrupt Set Register

(1204_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SIT	0	w	Set Error Interrupt Service Request 0 _B No action 1 _B DMA error interrupt service request will be activated. Reading this bit returns a 0
RES	[31:1]	r	Reserved Read as 0; must be written with 0.

Direct Memory Access (DMA)

15.8.7 DMA Pattern Detection Registers

Pattern Read Register 0

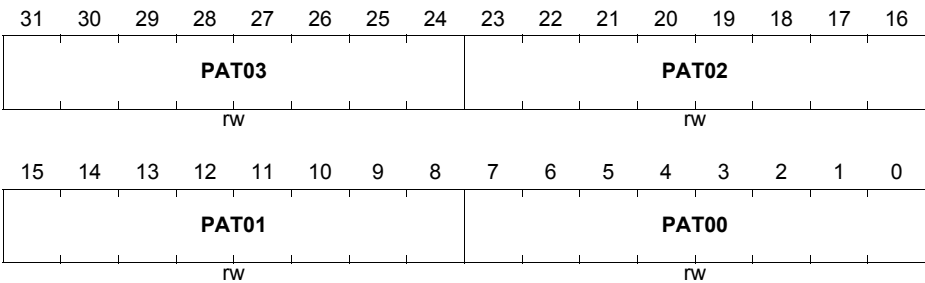
The Pattern Read Register 0 contains the patterns (mask and/or compare bits) to be processed by the Move Engine pattern detection logic.

DMA_PRR0

Pattern Read Register 0

(1208_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
PAT00, PAT01, PAT02, PAT03	[7:0], [15:8], [23:16], [31:24]	rw	Pattern for Move Engine Determines up to four 8-bit compare patterns/mask patterns to be processed by the pattern detection logic in the Move Engine. Depending on the pattern detection configuration (MEXCHCR.PATSEL) and channel data width (MEXCHCR.CHDW), the patterns are processed as bytes or half-words.

Direct Memory Access (DMA)

Pattern Read Register 1

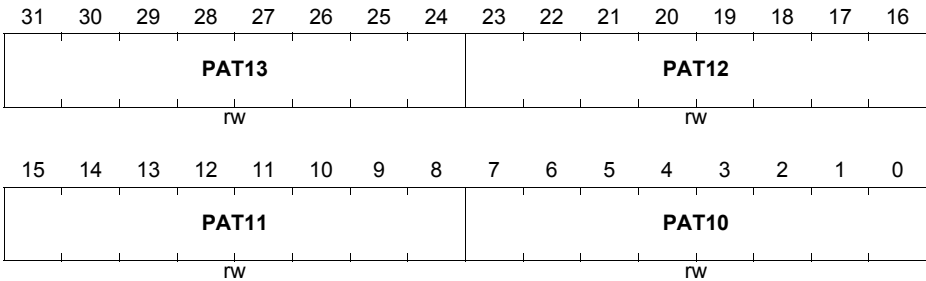
The Pattern Read Register 1 contains the patterns (mask and/or compare bits) to be processed by the Move Engine pattern detection logic.

DMA_PRR1

Pattern Read Register 1

(120C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
PAT10, PAT11, PAT12, PAT13	[7:0], [15:8], [23:16], [31:24]	rw	Pattern for Move Engine Determines up to four 8-bit compare patterns/mask patterns to be processed by the pattern detection logic in the Move Engine. Depending on the pattern detection configuration (MEXCHCR.PATSEL) and channel data width (MEXCHCR.CHDW), the patterns are processed as bytes or half-words.

15.8.8 DMA Flow Control Registers

Time Register

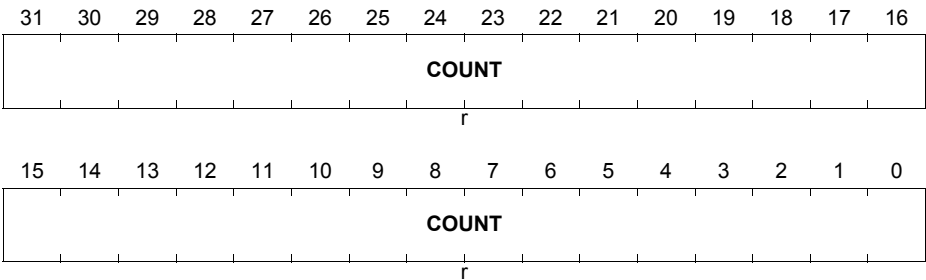
The Time Register contains the 32-bit count value used for the appendage of DMA timestamps.

DMA_TIME

Time Register

(1210_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
COUNT	[31:0]	r	Timestamp Count This bit field provides the count value used for the appendage of DMA timestamps.

Direct Memory Access (DMA)

15.8.9 DMA Channel Hardware Resource Registers

DMA Mode Registers

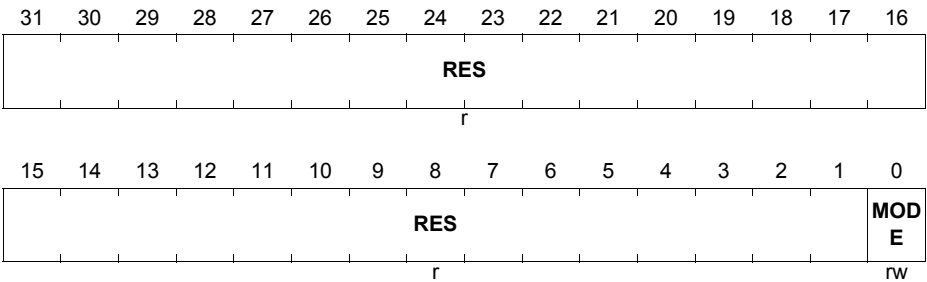
The DMA Mode Registers control whether hardware resources access the on chip buses in supervisor mode or user mode.

DMA_MODEy (y = 0-3)

DMA Mode Register y

$$(1300_H + y * 4_H)$$

Reset Value: 0000 0001_H



Field	Bits	Type	Description
MODE	0	rw	Hardware Resource Supervisor Mode Bit determines if a hardware resource 0 makes bus accesses in supervisor mode or user mode. 0 _B Bus master is in user mode. 1 _B Bus master is in supervisor mode.
RES	[31:1]	r	Reserved Read as 0; must be written with 0.

Direct Memory Access (DMA)
DMA Channel Hardware Resource Register z

The Hardware Resource Register assigns DMA channels to one of the four hardware resource partitions.

Initially all DMA channels are assigned to Hardware Resource 0 and read/write accesses are controlled by the ACCEN00 and ACCEN01 registers. If a DMA channel z is assigned to one of the other Hardware Resource Partitions then access control will change to the corresponding pair of access enable registers.

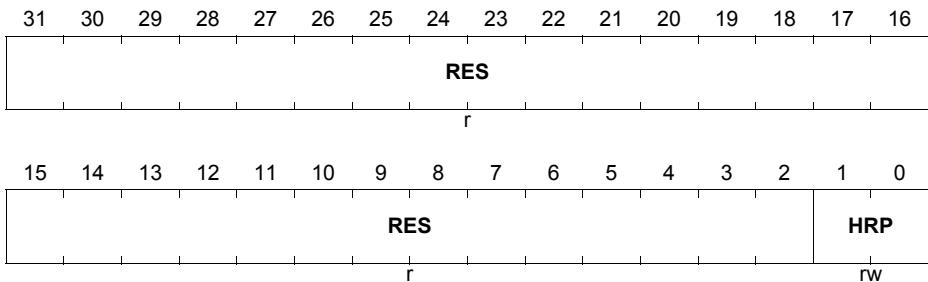
Each hardware resource partition supports a unique bus Master Tag Identifier (TAGID).

DMA_HRRz (z = 000-047)

DMA Channel Hardware Resource Register z

($1800_H + z * 0004_H$)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
HRP	[1:0]	rw	Hardware Resource Partition y 00 _B Hardware Resource Partition 0. 01 _B Hardware Resource Partition 1. 10 _B Hardware Resource Partition 2. 11 _B Hardware Resource Partition 3.
RES	[31:2]	r	Reserved Read as 0; must be written with 0.

15.8.10 DMA Channel Suspend Registers

DMA Channel Suspend Enable Register z

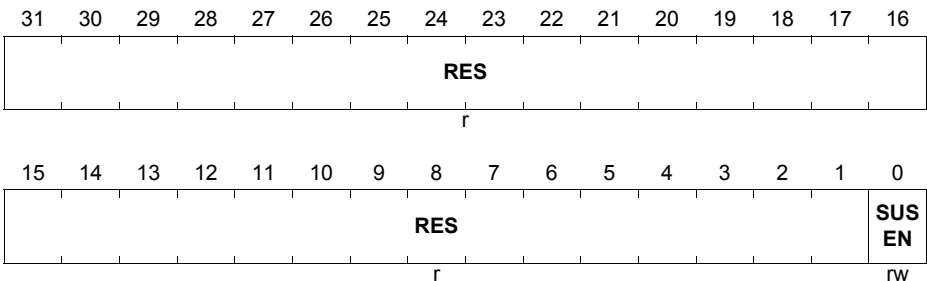
The Suspend Enable Register enables/disables soft suspend mode capability.

DMA_SUSENRz (z = 000-047)

DMA Suspend Enable Register z

(1A00_H + z * 0004_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SUSEN	0	rw	<p>Channel Suspend Enable for DMA Channel z</p> <p>This bit enables the channel suspend capability individually for each DMA channel z.</p> <p>0_B DMA channel z is disabled for channel suspend mode. The DMA channel z does not react on an active suspend request signal SUSREQ</p> <p>1_B DMA channel z is enabled for Soft-suspend Mode. If the suspend request signal SUSREQ becomes active, a DMA transaction of DMA channel z is stopped after the current DMA transfer has been finished</p> <p>Channel suspend mode can be terminated when SUSENz is written with 0.</p>
RES	[31:1]	r	<p>Reserved</p> <p>Read as 0; must be written with 0.</p>

Note: Register is only reset by the Debug Reset.

Direct Memory Access (DMA)
DMA Suspend Acknowledge Register z

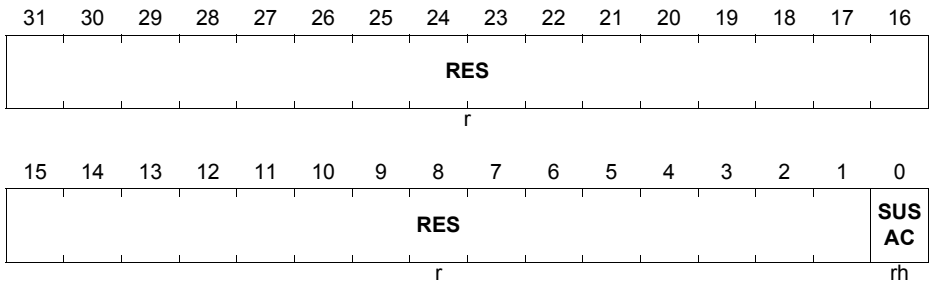
The Suspend Acknowledge Register indicates the DMA Channel soft suspend status.

DMA_SUSACRz (z = 000-047)

DMA Suspend Acknowledge Register z

($1C00_H + z * 0004_H$)

Reset Value: $0000\ 0000_H$



Field	Bits	Type	Description
SUSAC	0	rh	Channel Suspend Mode or Frozen State Active for DMA Channel z This status bit indicates whether or not DMA channel z is in channel suspend mode or in the frozen state. 0_B DMA channel z is not in channel suspend mode, frozen state or internal actions are not yet finished after the channel suspend mode or frozen state was requested. 1_B DMA channel z is in channel suspend mode or frozen state.
RES	[31:1]	r	Reserved Read as 0; must be written with 0.

Note: This register is only reset by the Debug Reset.

15.8.11 DMA Transaction State Registers

DMA Channel Transaction State Register z

The Transaction State Register supports:

- The enabling and assertion of DMA channel hardware requests.
- The recording of a transaction request lost event for the channel.
- The setting, reporting and clearing of a DMA channel halt.
- The assertion of a DMA channel reset.

DMA_TSRz (z = 000-047)

DMA Transaction State Register z

(1E00_H + z * 0004_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES						HLT CLR	RES						CTL	DCH	ECH
r						w	r						w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES						HLT ACK	HLT REQ	RES				CH	TRL	HTR E	RST
r						rh	rwh	r				rh	rh	rh	rwh

Field	Bits	Type	Description
RST	0	rwh	<p>DMA Channel Reset</p> <p>These bits force the DMA channel z to stop its current DMA transaction. Once set by software, this bit will be automatically cleared when the channel has been reset. Writing a 0 to RST has no effect.</p> <p>0_B No action (write) or the requested channel reset has been reset (read).</p> <p>1_B DMA channel z is stopped. More details see Page 15-20.</p>

Direct Memory Access (DMA)

Field	Bits	Type	Description
HTRE	1	rh	<p>Hardware Transaction Request Enable State</p> <p>0_B Hardware transaction request for DMA Channel is disabled. An input DMA request will not trigger the channel.</p> <p>1_B Hardware transaction request for DMA Channel is enabled. The transfers of a DMA transaction are controlled by the corresponding channel request line of the DMA requesting source.</p> <p><i>Note: HTRE Operation</i></p> <p>4. <i>Single Mode: HTRE is set to 0 when MExCHSR.TCOUNT is decremented and MExCHSR.TCOUNT = 0.</i></p> <p>5. <i>HTRE can be enabled and disabled with TSRz.ECH and TSRz.DCH.</i></p> <p>6. <i>HTRE is reset when a pattern match is detected.</i></p>
TRL	2	rh	<p>Transaction/Transfer Request Lost of DMA Channel</p> <p>0_B No request lost event has been detected for channel z.</p> <p>1_B A new DMA request was detected while TSRz.CH=1 (request lost event).</p> <p>This bit is reset by software when writing a 1 to TSRz.CTL or by a channel reset (writing TSRz.RST = 1).</p>
CH	3	rh	<p>Transaction Request State</p> <p>0_B No DMA request is pending for channel.</p> <p>1_B A DMA request is pending for channel.</p> <p>CH is reset when a pattern match is detected.</p>
HLTREQ	8	rwh	<p>Halt Request</p> <p>The halt request is a status bit that remains active until it is cleared by TSRz.HALTCLR</p> <p>0_B No action.</p> <p>1_B Halt request.</p>
HLTACK	9	rh	<p>Halt Acknowledge</p> <p>Halt acknowledge status from DMA channel.</p>
ECH	16	w	<p>Enable Hardware Transfer Request</p> <p>see table below</p>

Direct Memory Access (DMA)

Field	Bits	Type	Description
DCH	17	w	Disable Hardware Transfer Request see table below
CTL	18	w	Clear Transaction Request Lost for DMA Channel z Software clear of the DMA channel transaction request lost interrupt flag TSRz.TRL 0 _B No action 1 _B Clear DMA channel transaction request lost flag TSRz.TRL Reading this bit returns a 0.
HLTCLR	24	w	Clear Halt Request and Acknowledge Software write only active high clear of halt request TSRz.HALTREQ and halt acknowledge status bit TSRz.HALTACK 0 _B No action. 1 _B Halt clear. Reading this bit returns a 0.
RES	[7:4], [15:10], [23:19], [31:25]	r	Reserved Read as 0; must be written with 0.

Note: The DMA channel transaction/transfer request lost of a DMA channel is permanently available state that detects the loss of a DMA service request. If a transaction/transfer request lost is detected then an error interrupt trigger will be generated when the DMA channel z becomes active in a DMA sub-block x and if DMARAM bit ADICRz.ETRL enable bit is high.

DMA Channel Hardware Transaction/Transfer Request Conditions
Table 15-19 Conditions to Set/Reset the Bits TSRz.HTRE

TSRz.ECH	TSRz.DCH	Transaction Finishes ¹⁾ for Channel z	Modification of TSRz.HTRE
0	0	0	unchanged
1	0	0	set
X	1	X	reset
X	X	1	reset

Direct Memory Access (DMA)

1) In Single Mode only. In Continuous Mode, the end of a transaction has no impact.

15.8.12 DMA Transaction Control Set

The composition of a DMA transaction for each DMA channel is defined the transaction control set stored in the DMARAM. Each transaction control set consists of 8 X 32-bit words:

- DMA_RDCRCRz, DMA_SDCRCRz¹⁾, DMA_SADRz, DMA_DADRz and DMA_SHADRz store 32-bit CRC or address values.
- DMA_ADICRz and DMA_CHCFGRz store DMA channel configuration control bits.
- DMA_CHCSRz stores active channel status and write only active triggers.

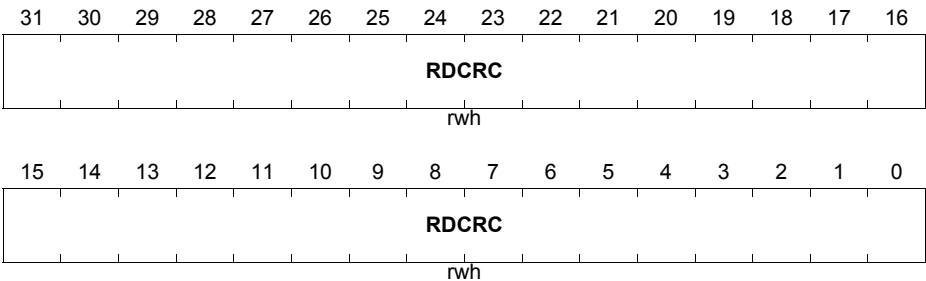
DMA Channel Read Data CRC Register

DMA_RDCRCRz (z = 000-047)

DMA Channel Read Data CRC Register z

($2000_H + z * 20_H$)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RDCRC	[31:0]	rwh	Read Data CRC This bit field stores the CRC32 ethernet polynomial checksum for read data of an inactive DMA channel.

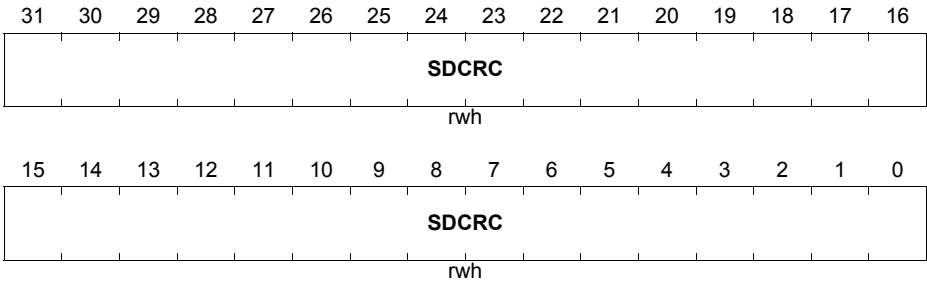
1) The Source and Destination Address CRC can detect failures in the DMA fetch configuration information and updates to the dynamic channel information.

Direct Memory Access (DMA)

DMA Channel Source and Destination Address CRC Register

DMA_SDCRCRz (z = 000-047)

DMA Channel Source and Destination Address CRC Register z
 (2004_H + z * 20_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
SDCRC	[31:0]	rwh	Source and Destination Address CRC This bit field stores the CRC32 ethernet polynomial checksum for the source and destination address of an inactive DMA channel.

Direct Memory Access (DMA)

DMA Channel Source Address Register

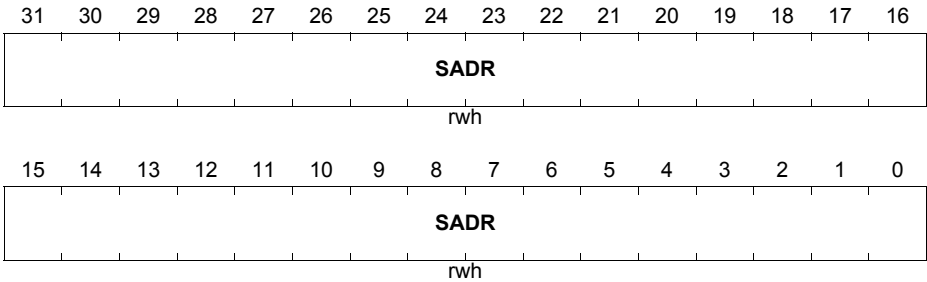
The Source Address Register stores the source address of an inactive DMA channel.

DMA_SADRz (z = 000-047)

DMA Channel Source Address Register z

$$(2008_H + z * 20_H)$$

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SADR	[31:0]	rwh	Source Address This bit field holds the actual 32-bit source address of an inactive DMA channel.

Note: If a DMA channel is configured to execute 16-bit DMA moves then the source address register should be aligned to a 16-bit start source address.

Direct Memory Access (DMA)

DMA Channel Destination Address Register

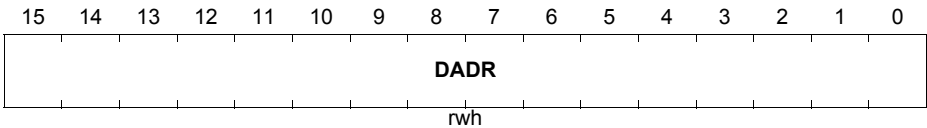
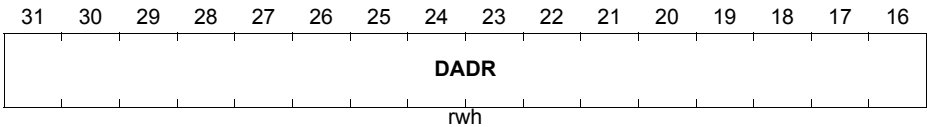
The Destination Address Register stores the destination address of an inactive DMA channel.

DMA_DADRz (z = 000-047)

DMA Channel Destination Address Register x

$$(200C_H + z * 20_H)$$

Reset Value: 0000 0000_H



Field	Bits	Type	Description
DADR	[31:0]	rwh	Destination Address This bit field holds the actual 32-bit destination address of an inactive DMA channel.

Note: If a DMA channel is configured to execute 16-bit DMA moves then the destination address register should be aligned to a 16-bit start destination address.

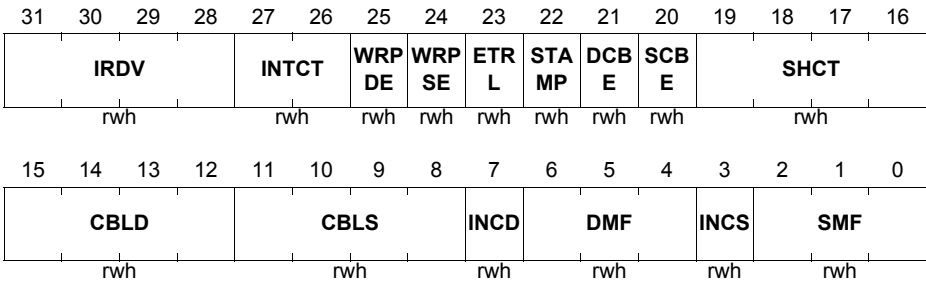
Direct Memory Access (DMA)
DMA Channel Address and Interrupt Control Register

The Address and Interrupt Control Register stores bits to update source and destination addresses and the generation of DMA channel traffic management interrupt triggers.

DMA_ADICRz (z = 000-047)

DMA Channel Address and Interrupt Control Register x

$$(2010_H + z * 20_H)$$

Reset Value: 0000 0000_H


Field	Bits	Type	Description
SMF	[2:0]	rwh	Source Address Modification Factor This bit field stores the Source Address Modification Factor of an inactive DMA channel.
INCS	3	rwh	Increment of Source Address This inactive DMA channel control bit determines if the source address is decremented or incremented.
DMF	[6:4]	rwh	Destination Address Modification Factor This bit field stores the Destination Address Modification Factor of an inactive DMA channel.
INCD	7	rwh	Increment of Destination Address This bit field stores the Destination Address Modification Factor for an inactive DMA channel.
CBLS	[11:8]	rwh	Circular Buffer Length Source This bit field stores circular buffer address update control of an inactive DMA channel.
CBLD	[15:12]	rwh	Circular Buffer Length Destination This bit field stores circular buffer address update control of an inactive DMA channel.
SHCT	[19:16]	rwh	Shadow Control Inactive DMA channel shadow control.

Direct Memory Access (DMA)

Field	Bits	Type	Description
SCBE	20	rwh	Source Circular Buffer Enable Inactive DMA channel source circular buffer enable.
DCBE	21	rwh	Destination Circular Buffer Enable Inactive DMA channel destination circular buffer enable.
STAMP	22	rwh	Time Stamp Enable appendage of time stamp.
ETRL	23	rwh	Enable Transaction Request Lost Interrupt Inactive DMA channel bit to enable the generation of an error interrupt service request.
WRPSE	24	rwh	Wrap Source Enable Inactive DMA channel source buffer interrupt trigger enable/disable.
WRPDE	25	rwh	Wrap Destination Enable Inactive DMA channel destination buffer interrupt trigger enable/disable.
INTCT	[27:26]	rwh	Interrupt Control Inactive DMA channel interrupt control.
IRDV	[31:28]	rwh	Interrupt Raise Detect Value Inactive DMA channel interrupt threshold value.

Shadow Address Register Read Only Mode

If $ADICRz.SHCT = 0001_B$ or $ADICRz.SHCT = 0010_B$ then source or destination address buffering is selected and SHADRz is written when a transaction is running. The SHADRz is automatically set to $0000\ 0000_H$ when the shadow transfer takes place. The user can read the shadow register in order to detect if the shadow transfer has already taken place. If the value in SHADRz is $0000\ 0000_H$, no shadow transfer can take place and the corresponding address register is modified according to the circular buffer rules.

Shadow Address Register Write Enable Mode

If $ADICRz.SHCT = 0101_B$ or $ADICRz.SHCT = 0110_B$ then the shadow register SHADRz can be directly written. The value stored in the MExSHADR is not modified when the shadow transfer takes place, the shadow mechanism remains active and the shadow transfer will be repeated until Channel z is reset or until the value in SHADRz is $0000\ 0000_H$, is written into the shadow register.

Direct Memory Access (DMA)
DMA Channel Configuration Register

The Channel Configuration Register stores control and configuration data.

DMA_CHCFGRz (z = 000-047)

DMA Channel Configuration Register z

($2014_H + z * 20_H$)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAprio		RES	PRSEL	RES	PATSEL			CHDW			CHMODE	RROAT	BLKM		
rwh		r	rwh	r	rwh			rwh			rwh	rwh	rwh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES		TREL													
r		rwh													

Field	Bits	Type	Description
TREL	[13:0]	rwh	Transfer Reload Value Inactive DMA channel transfer reload value.
BLKM	[18:16]	rwh	Block Mode Inactive DMA channel block mode control bit field.
RROAT	19	rwh	Reset Request Only After Transaction Inactive DMA channel control bit to reset the request state flag.
CHMODE	20	rwh	Channel Operation Mode Inactive DMA channel control bit to reset condition of the hardware transaction request enable bit.
CHDW	[23:21]	rwh	Channel Data Width Inactive DMA channel data width bit field.
PATSEL	[26:24]	rwh	Pattern Select Inactive DMA channel pattern select bit field.
PRSEL	28	rwh	Peripheral Request Select Inactive DMA channel hardware request trigger control.
DMAprio	[31:30]	rwh	DMA Priority Inactive DMA channel priority bit field.

Direct Memory Access (DMA)

Field	Bits	Type	Description
RES	[15:14], 27, 29	r	Reserved Read as 0; must be written with 0.

Direct Memory Access (DMA)

DMA Channel Shadow Address Register

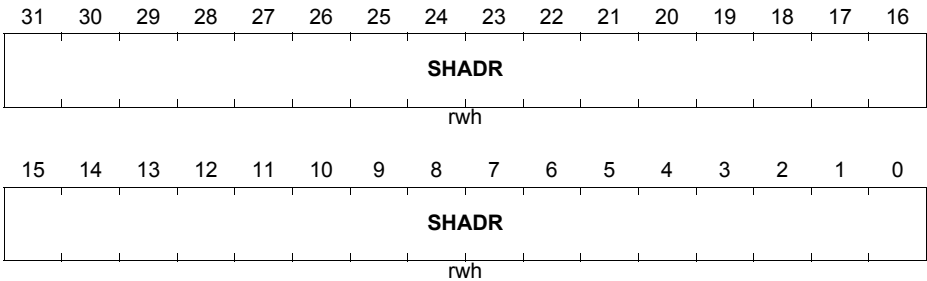
The Shadow Address Register stores the 32-bit shadow address of an inactive DMA channel.

DMA_SHADR_z (z = 000-047)

DMA Channel Shadow Address Register z

$$(2018_H + z * 20_H)$$

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SHADR	[31:0]	rwh	Shadowed Address This bit field stores the 32-bit shadow address of an inactive DMA channel.

Direct Memory Access (DMA)
DMA Channel Control and Status Register

The Channel Control and Status Register contains the current transfer count, channel reset, pattern detection compare result and the status of traffic management interrupt triggers.

DMA_CHCSRz (z = 000-047)
DMARAM Channel Control and Status Register z
(201C_H + z * 20_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SCH	RES		SIT	CICH	CWRP	SWB	FROZEN	BUFFER	RES		IPM	ICH	WRPD	WRPS	
w	r		w	w	w	w	rwh	rh	r		rh	rh	rh	rh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LXO	RES	TCOUNT													
rh	r	rh													

Field	Bits	Type	Description
TCOUNT	[13:0]	rh	Transfer Count Status DMA transfer count for an inactive DMA channel.
LXO	15	rh	Old Value of Pattern Detection Inactive DMA channel compare result.
WRPS	16	rh	Wrap Source Buffer Inactive DMA channel wrap-around of source buffer interrupt trigger flag.
WRPD	17	rh	Wrap Destination Buffer Inactive DMA channel wrap-around of destination buffer interrupt trigger flag.
ICH	18	rh	Interrupt from Channel Inactive DMA channel interrupt trigger flag.
IPM	19	rh	Pattern Detection from Channel Inactive DMA channel pattern detection flag.
BUFFER	22	rh	DMA Double Buffering Active Buffer This bit is active during DMA double buffering and indicates which buffer is read or filled. 0 _B Buffer 0 read or filled by DMA. 1 _B Buffer 1 read or filled by DMA.

Direct Memory Access (DMA)

Field	Bits	Type	Description
FROZEN	23	rwh	<p>DMA Double Buffering Frozen Buffer</p> <p>This bit is active during DMA double buffering and indicates that one of the double buffers is frozen and available for a cyclic software task.</p> <p>0_B Buffer is not frozen. 1_B Buffer is frozen.</p> <p><i>Note: FROZEN bit must only be cleared by software.</i></p>
SWB	24	w	<p>DMA Double Buffering Switch Buffer</p> <p>When double buffering is selected by programming ADICRz.SHCT then the control bit is used to re-direct data from one buffer to the other buffer.</p> <p>0_B No action. 1_B Switch from buffer.</p> <p><i>Note: If a Linked List mode (ADICRz.SHCT[3:2] = 11) is configured then SWB must be 0.</i></p>
CWRP	25	w	<p>Clear Wrap Buffer Interrupt z</p> <p>Software clear of both the DMA channel source and destination wrap buffer interrupts stored in DMARAM at locations CHSRz.WRPS and CHSRz.WRPD</p> <p>0_B No action. 1_B Bits CHSRz.WRPS and CHSRz.WRPD are reset.</p> <p><i>Note: If DMA channel z is active in Move Engine x then clear bit fields MExCHSR.WRPS and MExCHSR.WRPD</i></p> <p>Reading this bit returns a 0.</p>
CICH	26	w	<p>Clear Interrupt for DMA Channel z</p> <p>Software clear of the DMA channel interrupt flags stored in DMARAM locations CHSRz.ICH and CHSRz.IPM</p> <p>0_B No action. 1_B Bits MExCHSR.ICH and MExCHSR.IPM are reset.</p> <p><i>Note: If DMA channel z is active in Move Engine x then clear bit fields MExCHSR.ICH and MExCHSR.IPM</i></p> <p>Reading this bit returns a 0.</p>

Direct Memory Access (DMA)

Field	Bits	Type	Description
SIT	27	w	Set Interrupt Trigger for DMA Channel z 0 _B No action. 1 _B Channel z interrupt trigger will be activated. Reading this bit returns a 0. <i>Note: If a Linked List mode (ADICRz.SHCT[3:2] = 11) is configured then SIT must be 0.</i>
SCH	31	w	Set Transaction Request for DMA Channel 0 _B No action. 1 _B A transaction for DMA channel z is requested. When setting SCH, TSRz.CH becomes set to indicate that a DMA request is pending for DMA channel z.
RES	14, [21:20], [30:28]	r	Reserved Read as 0; must be written with 0.

15.9 Use Cases

This section provides a code example for the DMA module to give an overview about the core functionality. The example realizes a one word (32 bit-length) single data transfer from a data source location to a data destination location without intervention of the CPU. The transfer gets triggered by software in this example. The module also supports hardware triggering, please see [Chapter 15.4.3.3](#) for more details.

There will be no interrupt or further functions created in this example, please see the respective registers for more details on that. The examples uses DMA channel 000 (z=000).

Step description to initialize and trigger a data transfer via DMA:

(Line 1) The 32-bit data source address (DMA_SADR000_ADD) gets stored in the source address register SADR000 (see register definition [DMA_SADR](#)).

(Line 2) The 32-bit data destination address (DMA_DADR000_ADD) gets stored in the destination address register DADR000 (see register definition [DMA_DADR](#)).

(Line 3) The channel data width is defined in the channel configuration register CHCR000 (see register definition [DMA_CHCFGR](#)). Setting DMA_CHCFGR000.[23:21] = 010_B sets the channel data width to 32 bit.

(Line 4) DMA hardware transfer requests are disabled in the transaction state register TSR000 (see register definition [DMA_TSR](#)). Writing DMA_TSR000.[17] = 1 disables the hardware transfer requests.

(Line 5) This line starts the data transfer between the source and destination memory address.

Note: the declaration of DMA_SADR000_ADD and DMA_DADR000_ADD must be done by the user.

Initialization Code for 1 ms timer interrupt

```
(1) DMA_SADR000 = DMA_SADR000_ADD; //data source add
(2) DMA_DADR000 = DMA_DADR000_ADD; //data destination add
(3) DMA_CHCFGR000.U |= (010 << 21);
(4) DMA_TSR000.U |= (1 << 17); //disable hardware requests
(5) DMA_CHCSR000.U |= (1 << 31); //start transfer
```

Note: The transfer can be started everywhere in the program.

16 Flexible CRC Engine (FCE)

This document describes the Flexible CRC Engine (FCE) module. The FCE provides a parallel implementation of Cyclic Redundancy Code (CRC) algorithms. The current FCE version for the TC26x B-Step microcontroller implements the IEEE 802.3 ethernet CRC32, the CCITT CRC16 and the SAE J1850 CRC8 polynomials. FCE's generic structure enables it to be extended with multiple independent CRC polynomials. The primary target of FCE is to be used as an hardware acceleration engine for software applications or operating systems services (compatible with Autosar CRC "specification of CRC Routines") using CRC signatures. CRC algorithms are commonly used to calculate message signatures that can be used to check message integrity during transport over communication channels like internal busses or interfaces between micro-controllers. CRC signatures are also suitable to sign blocks of data residing in variable or invariable storage elements.

The FCE operates as a standard peripheral bus slave and is fully controlled through a set of configuration and control registers. The different CRC algorithms are independent from each other, they can be used concurrently by different software tasks.

This chapter is structured as follows:

- **"Features" on Page 16-3**
- **"Operational overview" on Page 16-4**
- **"FCE Functional Description" on Page 16-5**
- **"FCE Module Registers" on Page 16-13**
- **"Interfaces of the FCE Module" on Page 16-12**
- **"Programming Guide" on Page 16-37**
- **"Properties of CRC code" on Page 16-40**

*Note: The FCE kernel register names described in **"FCE Module Registers" on Page 16-13** are referenced in a product User's Manual by the module name prefix "FCE_".*

16.1 Related documentation

Input documents

- [D1] A painless guide to CRC Error Detection Algorithms, Ross N. Williams
- [D2] Autosar R3.1 Rev 0001, Specification of CRC Routines V3.0.2
- [D3] 32-Bit Cyclic Redundancy Codes for Internet Applications, Philip Koopman, International Conference on Dependable Systems and Networks (DSN), 2002

Related standards and norms

- [S1] IEEE 802.3 Ethernet 32-bits CRC

16.2 Features

The FCE provides the following features:

- The FCE implements the following CRC polynomials:
 - IEEE 802.3 CRC32 ethernet polynomial: $0x04C11DB7^{1)}$ (crc kernel 0 and 1)
 - CCITT CRC16 polynomial: $0x1021$ (crc kernel 2)
 - SAE J1850 CRC8 polynomial: $0x1D$ (crc kernel 3)
- Parallel CRC implementation
 - Data blocks to be computed by FCE shall be a multiple of the polynomial degree
 - Start address of Data blocks to be computed by FCE shall be aligned to the polynomial degree
- Register Interface compliant with Autosar specification for CRC routines:
 - Input Register
 - CRC Register
 - Configuration Registers enabling to control the CRC operation and perform automatic checksum checks at the end of a message.
 - Extended register interface to control reliability of FCE execution in safety applications.
- FCE can be reset independently by a module reset controlled by software. The reset affects all CRC engines.
- Error notification scheme via dedicated interrupt node for:
 - Transient error detection: error interrupt generation (maskable) with local status register (cleared by software)
 - Checksum failure: error interrupt generation (maskable) with local status register (cleared by software)
- FCE implements provides one interrupt line to the interrupt system. Each CRC engine has its own set of flag registers.

1) The polynomial hexadecimal representation covers the coefficients degree - 1 down to 0.

16.3 Operational overview

The FCE is a standard peripheral slave module. The FCE is fully synchronous with the peripheral bus clock and runs with a 1:1 clock ratio.

The FCE operation is controlled over a set of memory mapped registers. The main purpose is to serve as hardware acceleration for software applications requiring CRC checksum computation. The register set has been designed to enable the FCE. Depending on the hardware configuration the FCE may implement more crc kernels with different CRC polynomials. The specific configuration for a product will be described into the product customizing chapter. Every crc kernel will present the same hardware and software architecture. The rest of this document will focus only on the description of the generic CRC kernel architecture. The **Figure 16-1 “FCE system integration” on Page 16-4** shows the kernel configuration for the TC26x B-Step microcontroller.

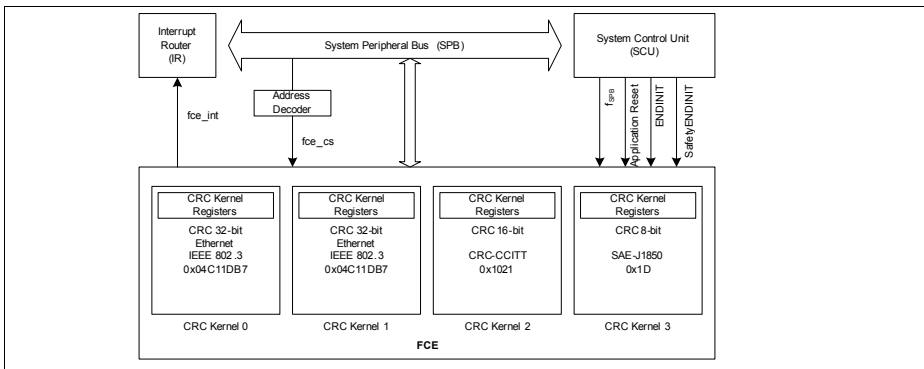


Figure 16-1 FCE system integration

In a multi-kernel implementation the interrupt lines are ored together, the FCE only present a single interrupt node to the system. Each crc kernel implements a status register that enables the software to identify which interrupt source is active. Please refer to the **STSm (m = 0-3)** register for a detailed description of the status and interrupt handling.

16.4 FCE Functional Description

16.4.1 Overview

The generic CRC kernel structure is presented in **Figure 16-2 “CRC kernel architecture”** on Page 16-5.

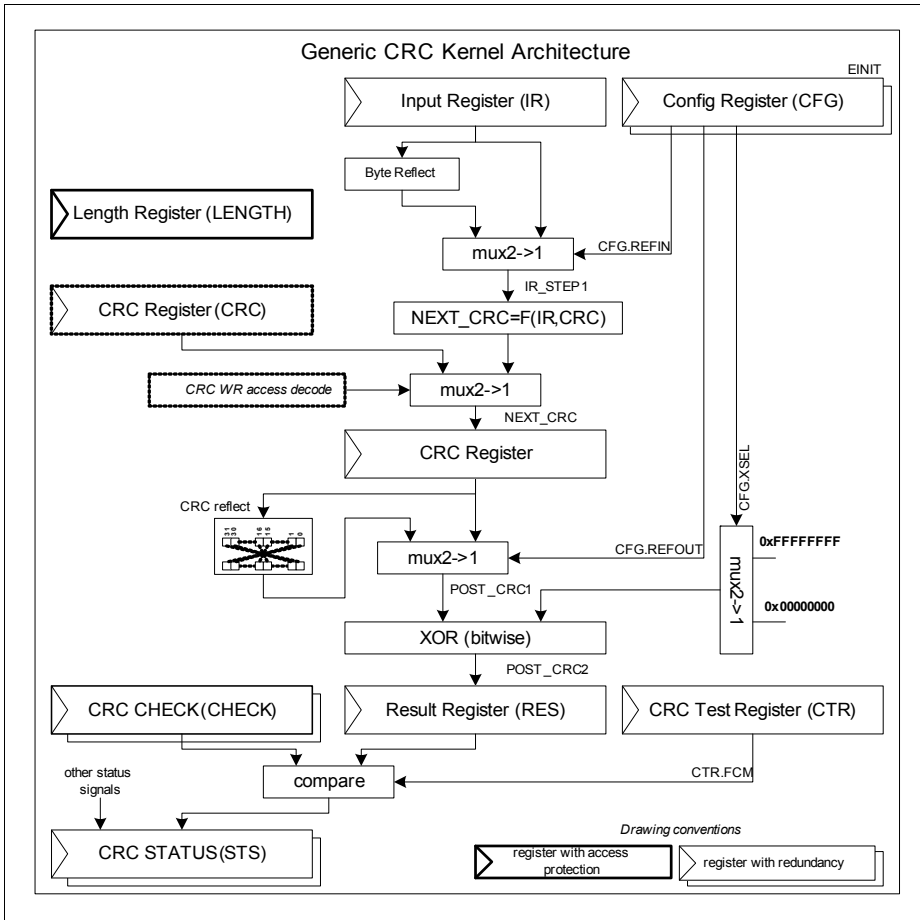


Figure 16-2 CRC kernel architecture

A checksum algorithm based on CRC polynomial division is characterized by the following properties:

Flexible CRC Engine (FCE)

- [1] polynomial degree (e.g. 32, that represents the highest power of two of the polynomial)
- [2] polynomial (e.g. 0x04C11DB7: the 33rd bit is omitted because always equal to 1)
- [3] init value: the initial value of the CRC register
- [4] input data reflected: indicates if each byte of the input parallel data is reflected before being used to compute the CRC
- [4] result data reflected: indicates if the final CRC value is reflected or not
- [5] XOR value: indicates if a final XOR operation is done before returning the CRC result

All the properties are static once a polynomial has been chosen. However the FCE provides the capability to control the two reflection steps and the final xor as depicted in [Figure 16-2 “CRC kernel architecture” on Page 16-5](#) through the CFG register. The reset values are compatible with the implemented algorithm. The final xor control enables to select either 0xFFFFFFFF or 0x00000000 to be xored with the POST_CRC1 (see [Figure 16-2 “CRC kernel architecture” on Page 16-5](#)) value. These two values are those used by the most common CRC polynomials.

Note: The reflection steps and final XOR do not modify the properties of the CRC algorithm in terms of error detection, only the CRC final signature is affected.

16.4.2 CRC Operation

Software must first ensure that the CRC kernel is properly configured, especially the initial CRC register value written via the **CRC** register. If the software wishes to use the automatic signature check at the end of a message, the **LENGTH** register and **CHECK** registers must be configured with respectively the length as number of words of the message and the expected signature (**CHECK**). The word length is defined by the degree of the polynomial used. The **CHECK** value takes into account the final CRC reflection and XOR operation. The self check is enable by the **CFG.CCE** bit field.

CRC Kernel access rules:

Depending on the CRC kernel accesses by software the following rules apply:

- When accessing a CRC kernel of degree $\langle N \rangle$ only the bits $N-1$ down to 0 are used by the CRC kernel. The upper bits are ignored on write. When reading from a CRC kernel register the non-used upper bits are set to 0.

Property:

If the input message $M1$ consists of a message $M0$ appended with the CRC signature of $M0$, then the CRC signature of $M1$ shall be 0.

The software writes as many times as necessary into the **IR** register according to the length of the message. If **CFG.CCE** bit field is set, every time the **IR** register is written, the **LENGTH** register is decremented by one. In the case the Automatic Length Reload feature is not enabled (see **CFGm (m = 0-3)** register), if **LENGTH** is already at zero but software still writes to **IR** (by mistake) every bit of the **LENGTH** should be set to 1 and hold this value until software initializes it again for the processing of a new message. In such case the **STS.LEF** (Length Error Flag) should be set and an interrupt generated if the **CFG.LEI** (Length Error Interrupt) is set.

The hardware monitors the transition of the **LENGTH** register from 1 to 0 to detect the end of the message and proceed with the comparison of the **POST_CRC2** (see **Figure 16-2**) value with the **CHECK** register value. If the Automatic Length Reload feature is enabled the **LENGTH** register is reinitialized with the previously configured value. This feature is especially suited when the FCE is used in combination with a DMA engine.

The next two figures provides an overview of the control and status features of a CRC kernel.

Flexible CRC Engine (FCE)

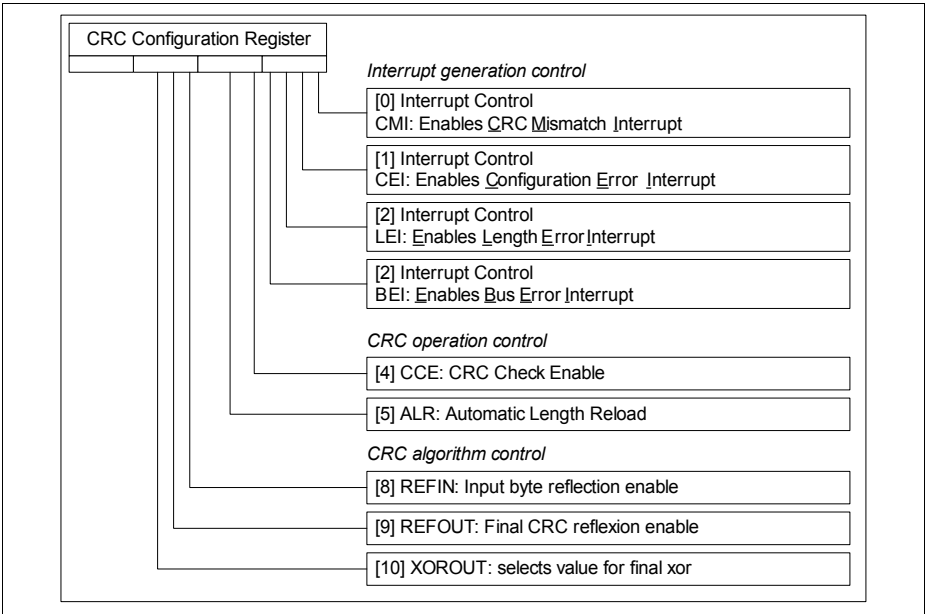


Figure 16-3 CRC kernel configuration register

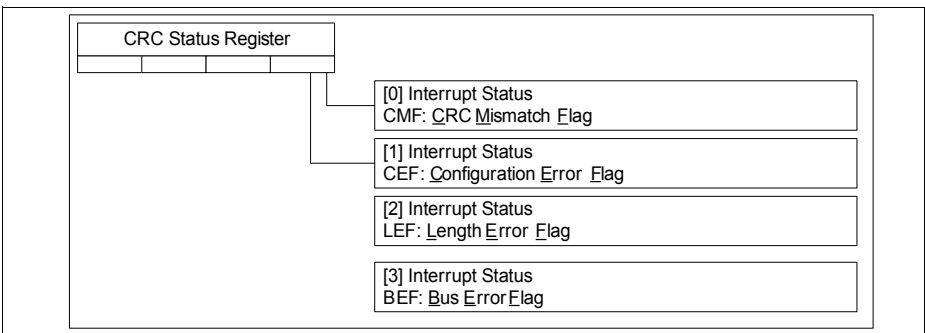


Figure 16-4 CRC kernel status register

16.4.3 Register protection and monitoring methods

Register Monitoring: applied to CFG and CHECK registers

Because CFG and CHECK registers are critical to the CRC operation, some mechanisms to detect and log transient errors are provided. Early detection of transient failures enables to improve the failure detection time and assess the severity of the failure. The monitoring mechanisms are implemented using two redundant instances as presented in [Figure 16-5](#).

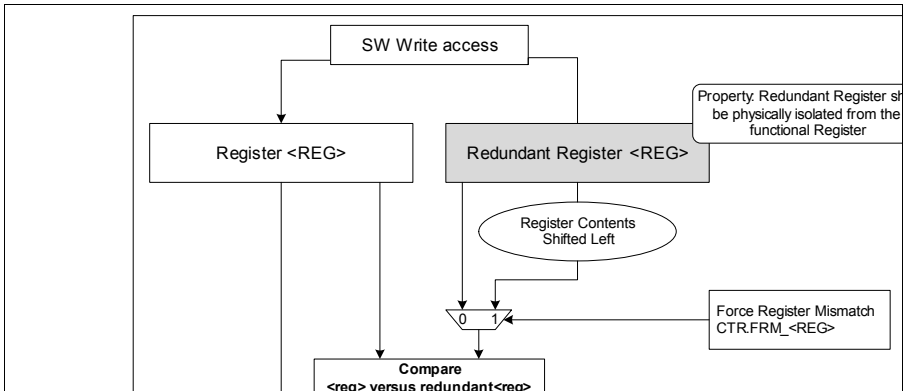


Figure 16-5 Register monitoring scheme

Let <REG> designate either CFG or CHECK registers. When a write to <REG> takes place a copy of the redundant register is also updated. **Redundant registers are not visible to software.** Bits of <REG> reserved have no storage and are not used for redundancy. A compare logic continuously compares the two stored values and provides a signal that indicates if the compare is successful or not. The result of all compare blocks are ored together to provide a single flag information. If a mismatch is detected the **STS.CEF** (Configuration Error Flag) bit is set. For run-time validation of the compare logic a Force Register Mismatch bit field (**CTR.FRM_<REG>**) is provided. When set to 1 by software the contents of the redundant register is shifted left by one bit position (redundant bit 0 position is always replaced by a logical 0 value) and is given to the compare logic instead of the redundant register value. This enables to check the compare logic is functional. Using a walking bit pattern, the software can completely check the full operation of the compare logic. Software needs to clear the **CTR.FRM_<REG>** bit to '0' to be able to trigger again a new comparison error interrupt.

Register Access Protection: applies to LENGTH and CHECK registers

In order to reduce the probability of a mis-configuration of the CHECK and LENGTH registers (in the case the automatic check is used), the write access to the CHECK and LENGTH registers must follow the procedure depicted in [Figure 16-6](#):

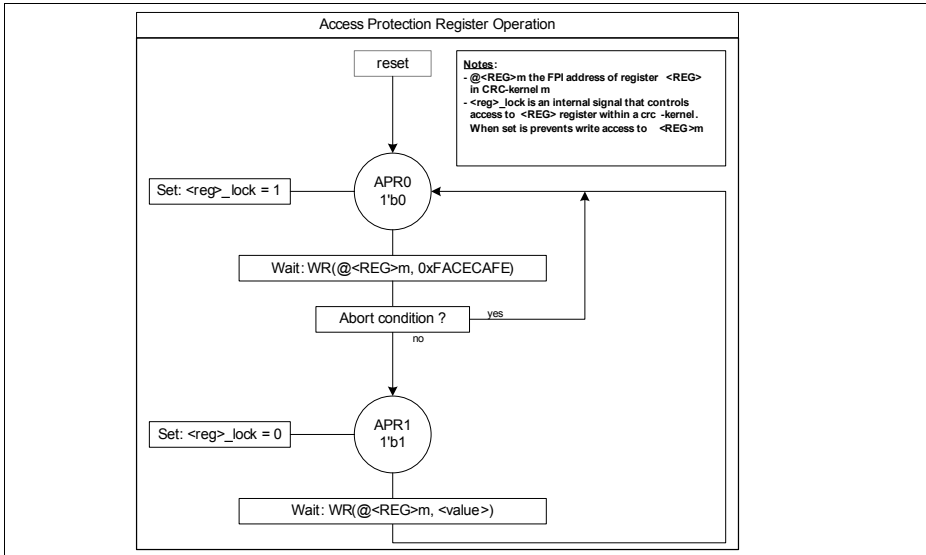


Figure 16-6 Access control to CHECK register

Let <REG> designate **CHECK** or **LENGTH** registers. Before being able to configure a new <value> value into the <REG> register of a crc kernel, software must first write the 0xFACECAFE value to the <REG> address. The 0xFACECAFE is not written into the CHECK register. The next write access will proceed as a normal bus write access. The write accesses shall use full 32-bit access only. This procedure will then be repeated every time software wants to configure a new <REG> value. If software reads the CHECK register just after writing 0xFACECAFE it returns the current <REG> contents and not 0xFACECAFE. **A read access to <REG> has no effect on the protection mechanism.**

16.4.4 FCE interrupts

Each FCE crc kernel provides one internal interrupt source. The interrupt lines from each crc kernel are ored together to be sent to interrupt system. The system interrupt is an active high pulse with the duration of one cycle (of the peripheral clock). The FCE interrupt handler can use the status information located within the **STS** status register of each crc kernel.

Flexible CRC Engine (FCE)

Each crc kernel provides the following interrupt sources:

- CRC Mismatch Interrupt controlled by **CFG.CMI** bit field and observable via the status bit field **STS.CMF** (CRC Mismatch Flag).
- Configuration Error Interrupt controlled by **CFG.CEI** bit field and observable via the status bit field **STS.CEF** (Configuration Error Flag).
- Length Error Interrupt controlled by **CFG.LEI** bit field and observable via the status bit field **STS.LEF** (Length Error Flag).
- Bus Error Interrupt controlled by **CFG.BEI** bit field and observable via the status bit field **STS.BEF** (Bus Error Flag).

Interrupt generation rules

- A status flag shall be cleared by software by writing a **1** to the corresponding bit position.
- If an status flag is set and a new hardware condition occurs, no new interrupt is generated by the kernel: the STS.<FLAG> bit field masks the generation of a new interrupt from the same source. If a SW access to clear the interrupt status bit takes place and in the same cycle the hardware wants to set the bit, the hardware condition wins the arbitration.

As all the interrupts are caused by an error condition, the interrupt shall be handled by a Error Management software layer. The software services using the FCE as acceleration engine may not directly deal with error conditions but let the upper layer using the service to deal with the error handling.

16.5 Interfaces of the FCE Module

The interfaces of the FCE module shall be described in the module design specification.

16.6 FCE Module Registers

Figure 16-7 and **Table 16-2** show all registers associated with a FCE crc-kernel. All FCE kernel register names are described in this section. They should get the prefix “FCE_” when used in the context of a product specification.

The registers are numbered by one index to indicate the related FCE CRC Kernel ($m = 0-3$). Some kernel registers are adapted to the degree of the polynomial implemented by the kernel.

FCE Registers Overview

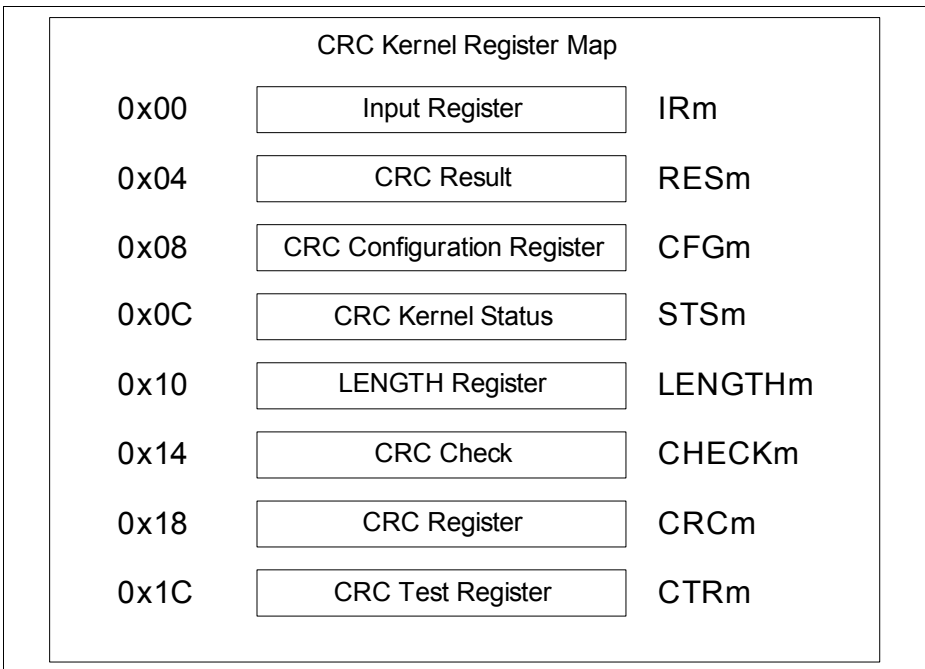


Figure 16-7 FCE Kernel Registers

Figure 16-8 shows the FCE module register map.

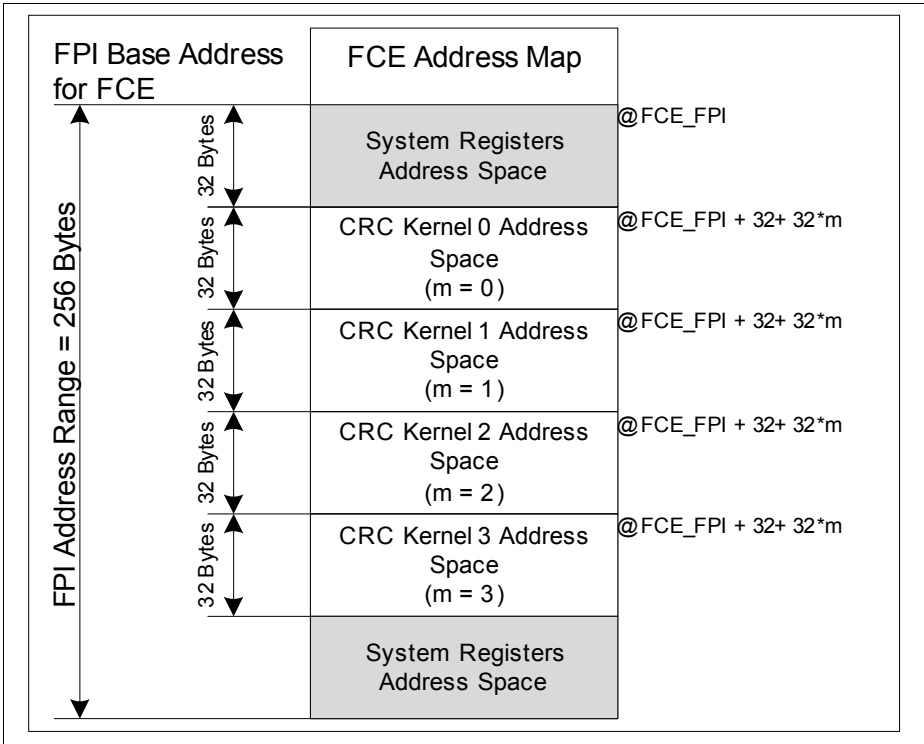


Figure 16-8 FCE Register Map

Table 16-1 Registers Address Space - FCE Module

Module	Base Address	End Address	Note
FCE	F000 3F00 _H	F000 3FFF _H	

Table 16-2 Registers Overview - CRC Kernel Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
System Registers						
CLC	Clock Control Register	00 _H	U, SV	E, SV	3	Page 16-17

Table 16-2 Registers Overview - CRC Kernel Registers

Short Name	Description	Offset Addr. ¹⁾	Access Mode		Reset Class	Description See
			Read	Write		
ID	Module Identification Register	08 _H	U, SV	BE	3	Page 16-18
Generic CRC Engine Registers						
IRm	Input Register m	20 _H + m*20 _H	U, SV	P,U, SV	3	Page 16-24
RESm	CRC Result Register m	24 _H + m*20 _H	U, SV	BE	3	Page 16-26
CFGm	CRC Configuration Register m	28 _H + m*20 _H	U, SV	P,E,SV	3	Page 16-28
STSm	CRC Status Register m	2C _H + m*20 _H	U, SV	P,U, SV	3	Page 16-30
LENGTHm	CRC Length Register m	30 _H + m*20 _H	U, SV	P,U, SV	3	Page 16-31
CHECKm	CRC Check Register m	34 _H + m*20 _H	U, SV	P,U,SV	3	Page 16-32
CRCm	CRC Register m	38 _H + m*20 _H	U, SV	P,U,SV	3	Page 16-34
CTRm	CRC Test Register m	3C _H + m*20 _H	U, SV	P,U,SV	3	Page 16-36
System Registers (Cont'd)						
KRSTCLR	Reset Status Clear Register	EC _H	U, SV	SV, E, P	3	Page 16-19
KRST1	Reset Control Register 1	F0 _H	U, SV	SV, E, P	3	Page 16-20
KRST0	Reset Control Register 0	F4 _H	U, SV	SV, E, P	3	Page 16-21
ACCEN1	Access Enable Register 1	F8 _H	U, SV	SV, SE	3	Page 16-22
ACCEN0	Access Enable Register 0	FC _H	U, SV	SV, SE	3	Page 16-23

1) The absolute register byte address for each crc kernel m is calculated as follows:
 CRC kernel register base Address ([Table 16-1](#)) + m*20H, m = 0-3

Access Mode Rules

The [Table 16-2 “Registers Overview - CRC Kernel Registers” on Page 16-14](#) uses the standard access mode conventions.

- E indicates that an access is only possible if the **end of initialization** signal is active. In this case Supervisor Mode (SV) is also mandatory.
- SE indicates that an access is only possible if the **safety end of initialization** signal is active. In this case Supervisor Mode (SV) is also mandatory.
- When U, SV are both listed it means that a read or write access can be done either in user mode (U) or supervisor mode (SV).
- BE stands for Bus Error, NSC stands for No Special Condition.
- P indicates that the register is protected by ACCEN0/1.

Disabling the FCE

The FCE module can be disabled using the **CLC** register.

When the disable state is requested all pending transactions running on the bus slave interface must be completed before the disabled state is entered. The CLC Register Module Disable Bit Status CLC.DISS indicates whether the module is currently disabled (DISS == 1). Any attempt to write any of the BPI writable registers with the exception of the CLC Register will generate a bus error. A read operation of BPI registers is allowed and does not generate a bus error.

Resetting the FCE

The FCE module can be reset using the **KRST0**, **KRST1**, and **KRSTCLR** registers. This action affects all the CRC kernels.

To reset the FCE it is necessary to set the RST bits by writing with '1' in both Reset Registers **KRST0**, **KRST1**. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Reset Register 0 includes a reset status bit that is set to '1' by the BPI in the same clock cycle the RST bit is re-set by the BPI. This bit can be used to detect that a reset was processed. The bit can be re-set to '0' by writing to it with '1'.

16.6.1 System Registers description

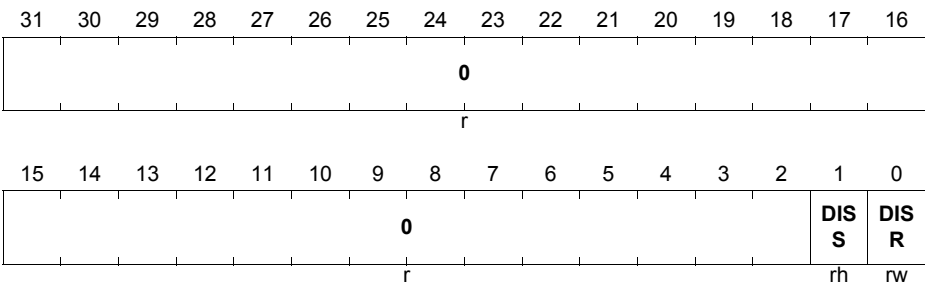
This section describes the registers related to the product system architecture.

Clock Control Register (CLC)

The Clock Control Register allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

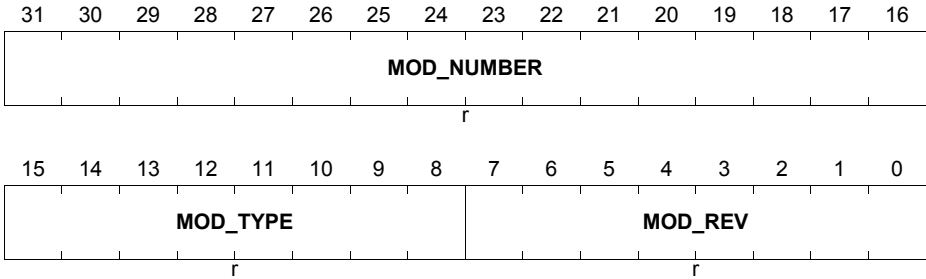
CLC

Clock Control Register (00_H) **Reset Value: 0000 0003_H**



Field	Bits	Type	Description
DIRS	0	rw	Module Disable Request Bit Used for enable/disable control of the module.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Note: The features related to the pre-defined FDIS, EDIS and RMC fields are not supported by the FCE. Therefore the corresponding fields have been removed from the CLC register.

Module Identification Register
ID
Module Identification Register (08_H) Reset Value: 00CA C002_H


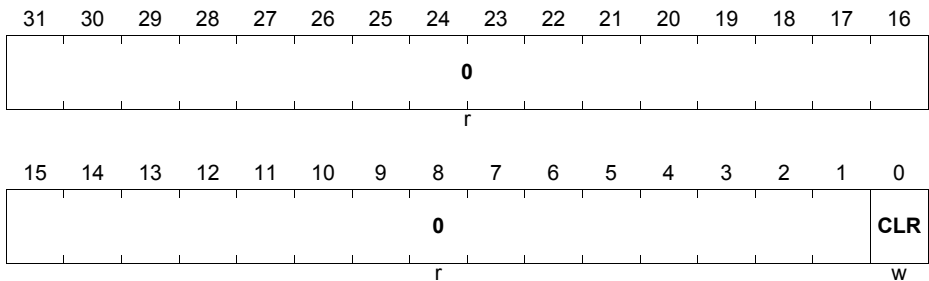
Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01 _H (first revision). The current revision number is 02 _H .
MOD_TYPE	[15:8]	r	Module Type The bit field is set to C0 _H which defines the module as a 32-bit module.
MOD_NUMBER	[31:16]	r	Module Number Value This bit field defines a module identification number. The value for the FCE module is 00CA _H .

Kernel Reset Clear Register (KRSTCLR)

Refer to **“Resetting the FCE” on Page 16-16** for the usage of the register.

KRSTCLR

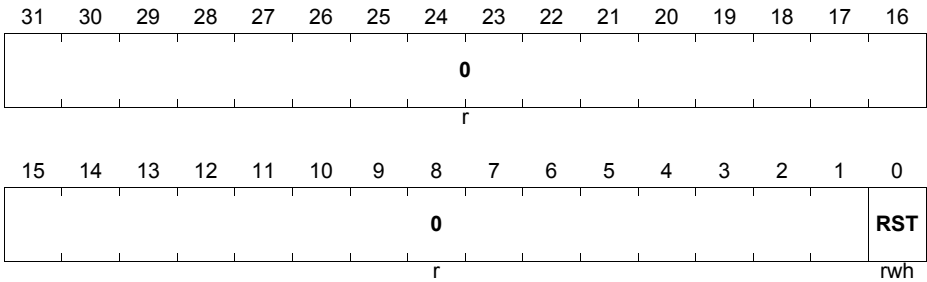
Kernel Reset Status Clear Register (EC_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

 Refer to **“Resetting the FCE” on Page 16-16** for the usage of the register.

KRST1
Kernel Reset Register 1
(F0_H)
Reset Value: 0000 0000_H


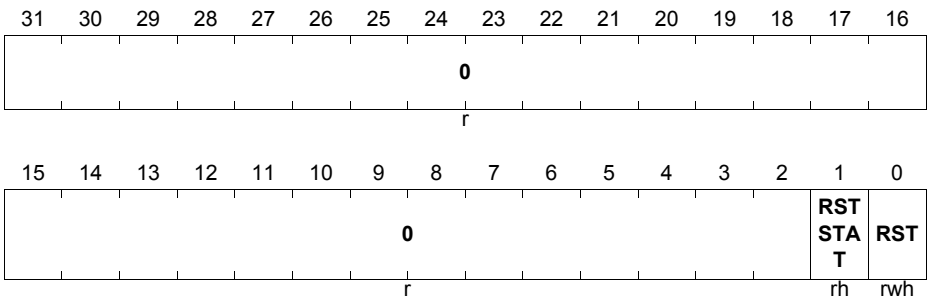
Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 0 (KRST0)

Refer to **“Resetting the FCE” on Page 16-16** for the usage of the register.

KRST0

Kernel Reset Register 0 (F4_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set.</p> <p>0_B No kernel reset was requested 1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p>
RSTSTAT	1	rw	<p>Kernel Reset Status</p> <p>This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits.</p> <p>0_B No kernel reset was executed 1_B Kernel reset was executed</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.</p>
0	[31:2]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Access Enable Register 1 (ACCEN1)

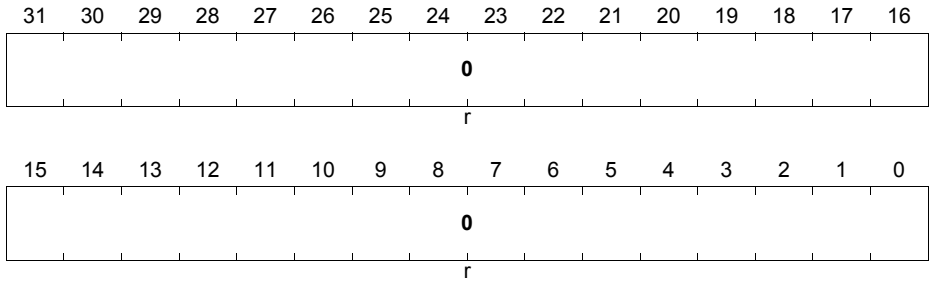
The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000_B to 111111_B.

ACCEN1

Access Enable Register 1

(F8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Access Enable Register 0 (ACCEN0)

The Access Enable Register 0 controls write access for transactions with the on chip bus master TAG ID 00000_B to 011111_B.

ACCEN0
Access Enable Register 0
(FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

16.6.2 CRC Kernel Control/Status Registers

CRC Engine Input Register

IR_m (m = 0-1)

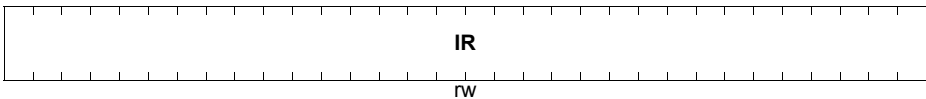
Input Register m

(20_H + m*20_H)

Reset Value: 0000 0000_H

31

0



Field	Bits	Type	Description
IR	[31:0]	rw	Input Register This bit field holds the 32-bit data to be computed

A write to IR_m triggers the CRC kernel to update the message checksum according to the IR contents and to the current CRC register contents. Only 32-bit write transactions are allowed to this IR_m registers, any other bus write transaction will lead to a Bus Error.

CRC Engine Input Register

IR_m (m = 2-2)

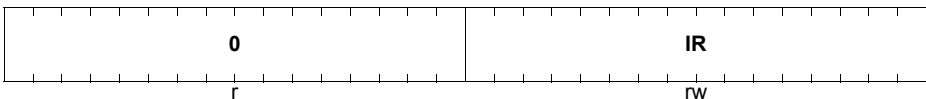
Input Register m

(20_H + m*20_H)

Reset Value: 0000 0000_H

31

0



Field	Bits	Type	Description
IR	[15:0]	rw	Input Register This bit field holds the 16-bit data to be computed
0	[31:16]	r	Reserved Read as 0; should be written with 0.

A write to IR_m triggers the CRC kernel to update the message checksum according to the IR contents and to the current CRC register contents. Only 32-bit or 16-bit write

Flexible CRC Engine (FCE)

transactions are allowed to this IRm register, any other bus write transaction will lead to a Bus Error. Only the lower 16-bit of the write transactions will be used.

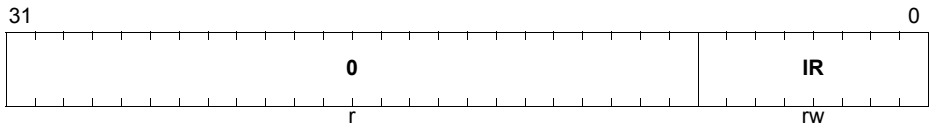
CRC Engine Input Register

IRm (m = 3-3)

Input Register m

(20_H + m*20_H)

Reset Value: 0000 0000_H



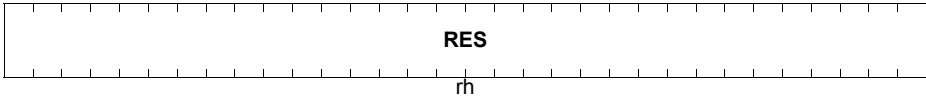
Field	Bits	Type	Description
IR	[7:0]	rw	Input Register This bit field holds the 8-bit data to be computed
0	[31:8]	r	Reserved Read as 0; should be written with 0.

A write to IRm triggers the CRC kernel to update the message checksum according to the IR contents and to the current CRC register contents. Any write transaction is allowed to this IRm register. Only the lower 8-bit of the write transactions will be used.

CRC Engine Result Register
RES_m (m = 0-1)
CRC Result Register m
(24_H + m*20_H)
Reset Value: FFFF FFFF_H

31

0

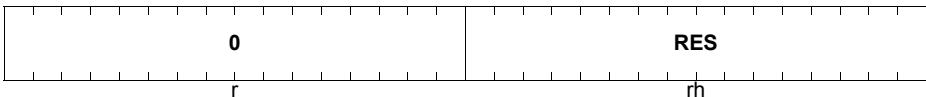


Field	Bits	Type	Description
RES	[31:0]	rh	Result Register Returns the final CRC value including CRC reflection and final XOR according to the CFG register configuration. Writing to this register has no effect.

CRC Engine Result Register
RES_m (m = 2-2)
CRC Result Register m
(24_H + m*20_H)
Reset Value: 0000 FFFF_H

31

0



Field	Bits	Type	Description
RES	[15:0]	rh	Result Register Returns the final CRC value including CRC reflection and final XOR according to the CFG register configuration. Writing to this register has no effect.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

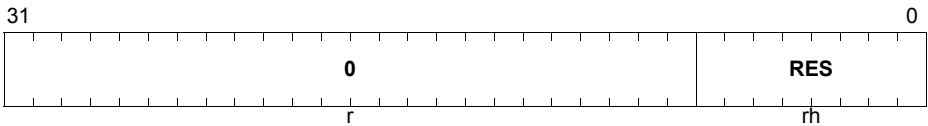
CRC Engine Result Register

RESm (m = 3-3)

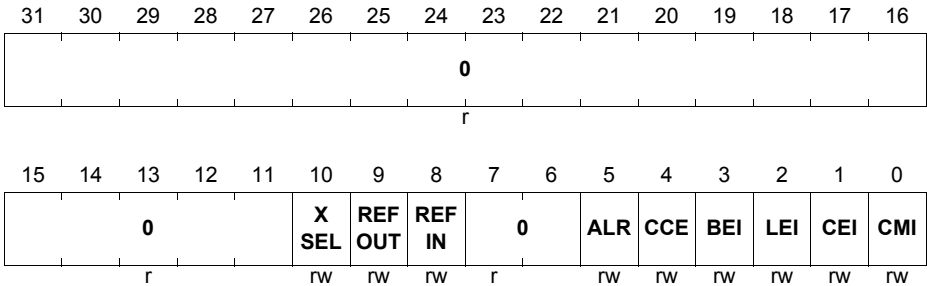
CRC Result Register m

(24_H + m*20_H)

Reset Value: 0000 00FF_H



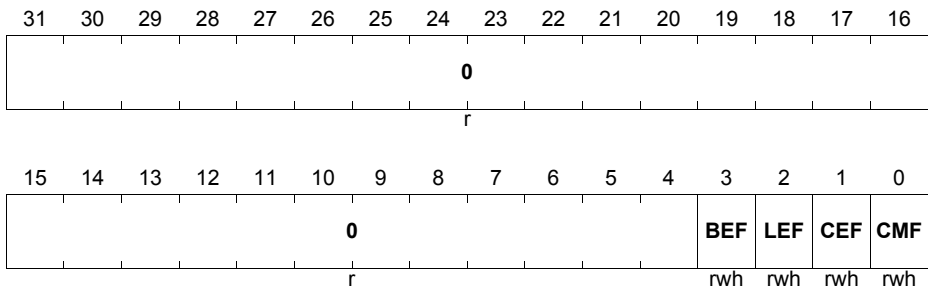
Field	Bits	Type	Description
RES	[7:0]	rh	Result Register Returns the final CRC value including CRC reflection and final XOR according to the CFG register configuration. Writing to this register has no effect.
0	[31:8]	r	Reserved Read as 0; should be written with 0.

CRC Engine Configuration Register
CFGm (m = 0-3)
CRC Configuration Register m
(28_H + m*20_H)
Reset Value: 0000 0700_H


Field	Bits	Type	Description
CMI	0	rw	CRC Mismatch Interrupt 0 _B CRC Mismatch Interrupt is disabled 1 _B CRC Mismatch Interrupt is enabled
CEI	1	rw	Configuration Error Interrupt When enabled, a Configuration Error Interrupt is generated whenever a mismatch is detected in the CFG and CHECK redundant registers. 0 _B Configuration Error Interrupt is disabled 1 _B Configuration Error Interrupt is enabled
LEI	2	rw	Length Error Interrupt When enabled, a Length Error Interrupt is generated if software writes to IR register with LENGTH equal to 0 and CFG.CCE is set to 1. 0 _B Length Error Interrupt is disabled 1 _B Length Error Interrupt is enabled
BEI	3	rw	Bus Error Interrupt When enabled, an interrupt is generated if a bus write transaction with an access width smaller than the kernel width is issued to the input register. 0 _B Bus Error Interrupt is disabled 1 _B Bus Error Interrupt is enabled

Flexible CRC Engine (FCE)

Field	Bits	Type	Description
CCE	4	rw	CRC Check Comparison 0 _B CRC check comparison at the end of a message is disabled 1 _B CRC check comparison at the end of a message is enabled
ALR	5	rw	Automatic Length Reload 0 _B Disables automatic reload of the LENGTH field. 1 _B Enables automatic reload of the LENGTH field at the end of a message.
REFIN	8	rw	IR Byte Wise Reflection 0 _B IR Byte Wise Reflection is disabled 1 _B IR Byte Wise Reflection is enabled
REFOUT	9	rw	CRC 32-Bit Wise Reflection 0 _B CRC 32-bit wise is disabled 1 _B CRC 32-bit wise is enabled
XSEL	10	rw	Selects the value to be xored with the final CRC 0 _B 0x00000000 1 _B 0xFFFFFFFF
0	[7:6], [31:11]	r	Reserved Read as 0; should be written with 0.

CRC Engine Status Register
STSm (m = 0-3)
CRC Status Register m ($2C_H + m \cdot 20_H$) **Reset Value: 0000 0000_H**


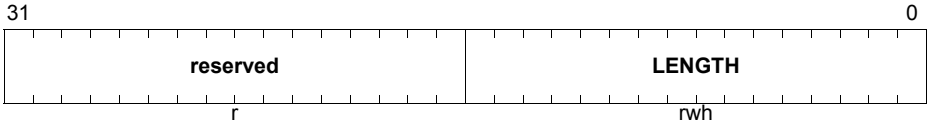
Field	Bits	Type	Description
CMF	0	rwh	CRC Mismatch Flag This bit is set per hardware only. To clear this bit, software must write a 1 to this bit field location. Writing 0 per software has no effect.
CEF	1	rwh	Configuration Error Flag This bit is set per hardware only. To clear this bit, software must write a 1 to this bit field location. Writing 0 per software has no effect.
LEF	2	rwh	Length Error Flag This bit is set per hardware only. To clear this bit, software must write a 1 to this bit field location. Writing 0 per software has no effect.
BEF	3	rwh	Bus Error Flag This bit is set per hardware only. To clear this bit, software must write a 1 to this bit field location. Writing 0 per software has no effect.
0	[31:4]	r	Reserved Read as 0; should be written with 0.

Flexible CRC Engine (FCE)

CRC Engine Length Register

LENGTH_m (m = 0-3)

CRC Length Register m (30_H + m*20_H) Reset Value: 0000 0000_H



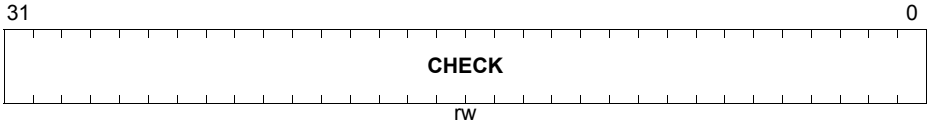
Field	Bits	Type	Description
LENGTH	[15:0]	rwh	Message Length Register Number of 32-bits words building the message over which the CRC checksum is calculated. This bit field is modified by the hardware: every write to the IR register decrements the value of the LENGTH bit field. If the CFG.ALR field is set to 1, the LENGTH field shall be reloaded with its configuration value at the end of the cycle where LENGTH reaches 0.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

Flexible CRC Engine (FCE)

CRC Engine Check Register

CHECK_m (m = 0-1)

CRC Check Register m (34_H + m*20_H) Reset Value: 0000 0000_H

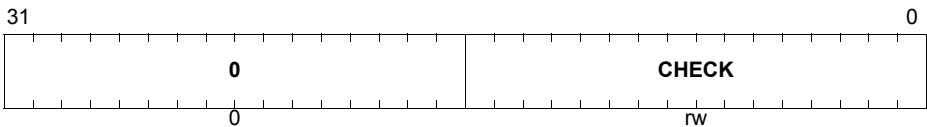


Field	Bits	Type	Description
CHECK	[31:0]	rw	CHECK Register Expected CRC value to be checked by the hardware upon detection of a 1 to 0 transition of the LENGTH register. The comparison is enabled by the CFG.CCE bit field

CRC Engine Check Register

CHECK_m (m = 2-2)

CRC Check Register m (34_H + m*20_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
CHECK	[15:0]	rw	CHECK Register Expected CRC value to be checked by the hardware upon detection of a 1 to 0 transition of the LENGTH register. The comparison is enabled by the CFG.CCE bit field
0	[31:16]	r	Reserved Read as 0; should be written with 0.

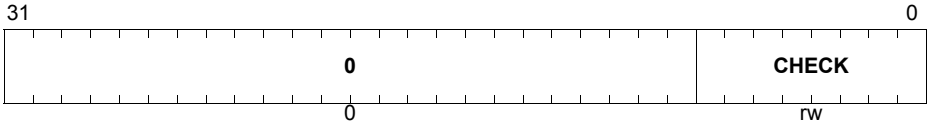
CRC Engine Check Register

CHECK_m (m = 3-3)

CRC Check Register m

(34_H + m*20_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CHECK	[7:0]	rw	CHECK Register Expected CRC value to be checked by the hardware upon detection of a 1 to 0 transition of the LENGTH register. The comparison is enabled by the CFG.CCE bit field
0	[31:8]	r	Reserved Read as 0; should be written with 0.

Flexible CRC Engine (FCE)

CRC Engine Initialization Register

CRCm (m = 0-1)

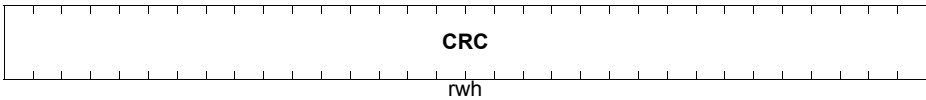
CRC Register m

 $(38_H + m*20_H)$

 Reset Value: 0000 0000_H

31

0



Field	Bits	Type	Description
CRC	[31:0]	rwh	CRC Register This register enables to directly access the internal CRC register

CRC Engine Initialization Register

CRCm (m = 2-2)

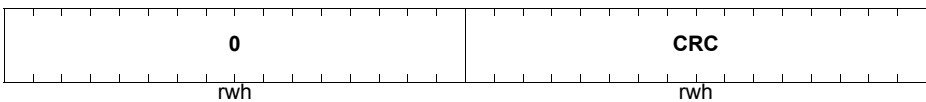
CRC Register m

 $(38_H + m*20_H)$

 Reset Value: 0000 0000_H

31

0



Field	Bits	Type	Description
CRC	[15:0]	rwh	CRC Register This register enables to directly access the internal CRC register
0	[31:16]	r	Reserved Read as 0; should be written with 0.

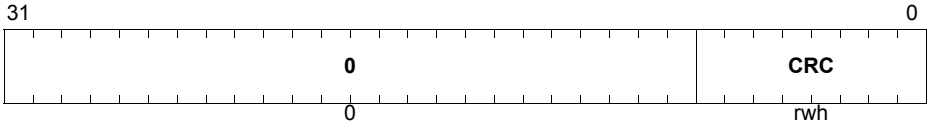
CRC Engine Initialization Register

CRCm (m = 3-3)

CRC Register m

(38_H + m*20_H)

Reset Value: 0000 0000_H

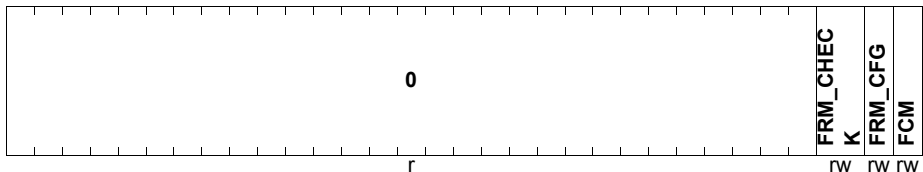


Field	Bits	Type	Description
CRC	[7:0]	rwh	CRC Register This register enables to directly access the internal CRC register
0	[31:8]	r	Reserved Read as 0; should be written with 0.

Flexible CRC Engine (FCE)
CRC Test Register
CTRM (m = 0-3)
CRC Test Register m
 $(3C_H + m*20_H)$
Reset Value: 0000 0000_H

31

0



Field	Bits	Type	Description
FCM	0	rw	Force CRC Mismatch Forces the CRC compare logic to issue an error regardless of the CHECK and CRC values. The hardware detects a 0 to 1 transition of this bit field and triggers a CRC Mismatch interrupt
FRM_CFG	1	rw	Force CFG Register Mismatch This field is used to control the error injection mechanism used to check the compare logic of the redundant CFG registers. This is a one shot operation. When the hardware detects a 0 to 1 transition of this bit field it triggers a Configuration Mismatch interrupt (if enabled by the corresponding CFGm register).
FRM_CHECK	2	rw	Force Check Register Mismatch This field is used to control the error injection mechanism used to check the compare logic of the redundant CHECK registers. This is a one shot operation. The hardware detects a 0 to 1 transition of this bit field and triggers a Check Register Mismatch interrupt (if enabled by the corresponding CFGm register).
0	[31:3]	r	Reserved Read as 0; should be written with 0.

16.7 Programming Guide

This section provides some guidelines showing how the FCE configuration features can be mapped to the AUTOSAR API for CRC32 routines.

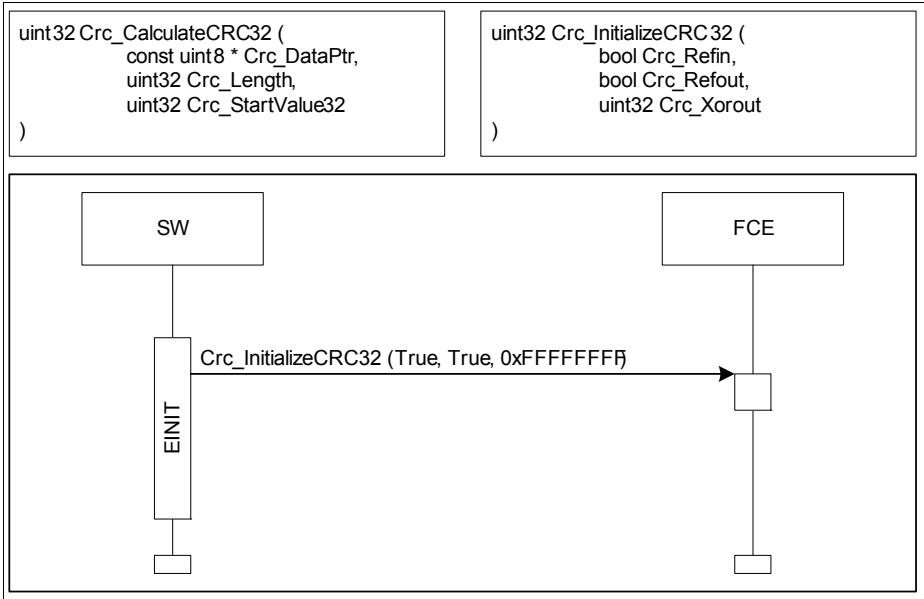


Figure 16-9 Autosar initialization API

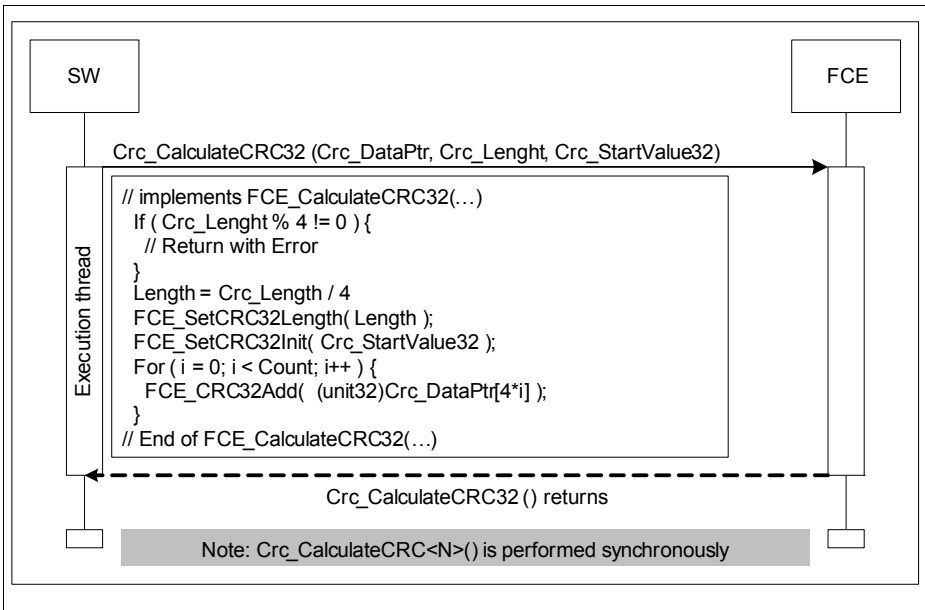


Figure 16-10 Autosar CRC_calculate API

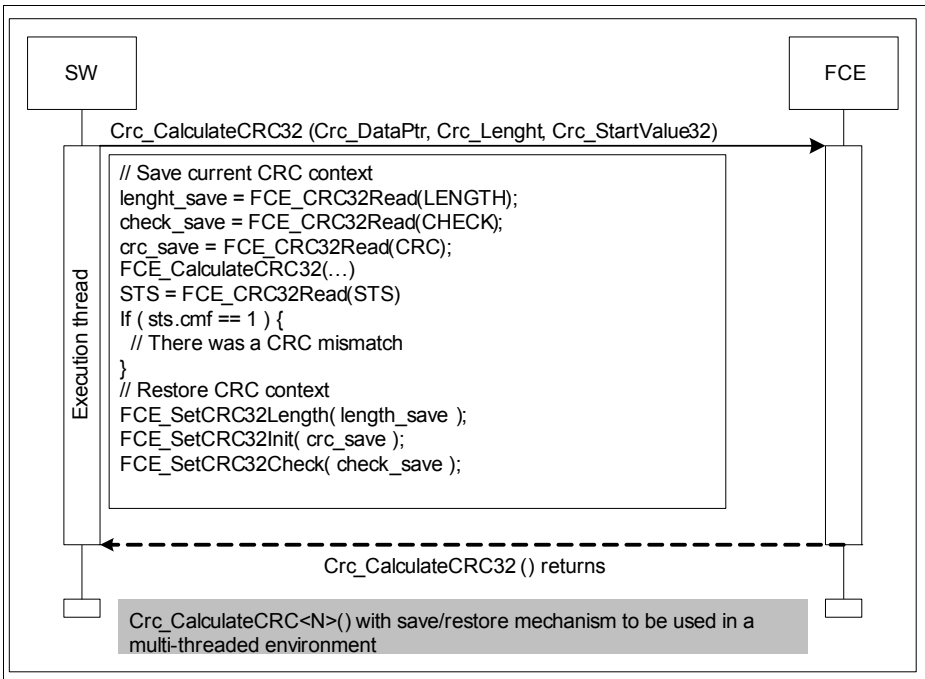


Figure 16-11 Autosar CRC_calculate API with save/restore

16.8 Properties of CRC code

Hamming Distance

The Hamming distance defines the error detection capability of a CRC polynomial. A cyclic code with a Hamming Distance of D can detect all D-1 bit errors. [Table 16-3 “Hamming Distance as a function of message length \(bits\)” on Page 16-40](#) shows the dependency of the Hamming Distance with the length of the message.

Table 16-3 Hamming Distance as a function of message length (bits)¹⁾

Hamming Distance	IEEE-802.3 CRC32	CCITT CRC16	J1850 CRC8
15	8 - 10	Information not available	Information not available
14	8 - 10		
13	8 - 10		
12	11 - 12		
11	13 - 21		
10	22 - 34		
9	35 - 57		
8	58 - 91		
7	92 - 171		
6	172 - 268		
5	269 - 2974		
4	2973 - 91607		
3	91607 - 131072		

1) Data from technical paper “32-Bit Cyclic Redundancy Codes for Internet Applications” by Philip Koopman, Carnegie Mellon University, 2002

17 Interrupt Router (IR)

The chapter describes the Interrupt Router (IR) module that schedules on TC26x B-Step Interrupts (here called service requests) from external resources, internal resources and SW to the TriCore CPU and to the DMA modules (here called service provider).

Topics covered include the architecture of the interrupt system, interrupt system configuration, and the interrupt operations of the TC26x B-Step peripherals.

17.1 Overview

An interrupt request can be serviced either by one of the CPUs or by the DMA module. Interrupt requests are called “service requests” rather than “interrupt requests” in this document because they can be serviced by either one of the service providers.

The interrupt system in the TC26x B-Step is realized in the Interrupt Router module which includes the Service Request Nodes (SRNs), the Interrupt Control Units (ICUs) and additional functionality for SW development support.

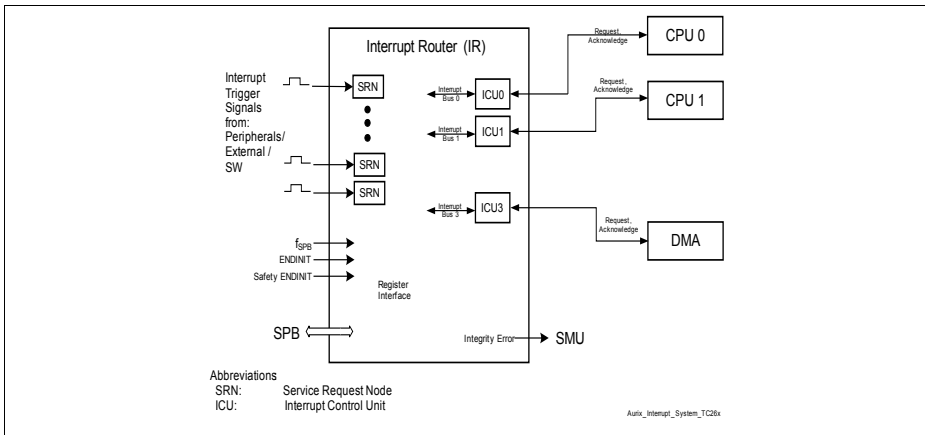
As shown in [Figure 17-1](#), each TC26x B-Step module that can generate service requests is connected to one or more Service Request Nodes (SRNs) in the central Interrupt Router module. The Interrupt Router module includes also several general purpose Service Request Nodes (SRNs) that can be used for software (SW) triggered service requests.

Each SRN contains a Service Request Control Register (SRC) to configure the service request regarding e.g. priority, mapping to one of the available service providers.

Each SRN is connected to all ICUs in the interrupt router module where the SRNs control register setting defines the target service provider and the priority of the service request.

Each ICU handles the interrupt arbitration among competing service requests from SRNs that are mapped to the ICU.

Each ICU is connected to one service provider (CPU or DMA module) where the ICU offers the valid winning service request/SRN of an arbitration round and the service provider signals back to the ICU when and which service request it is processing.

Interrupt Router (IR)

Figure 17-1 Block Diagram of the TC26x B-Step Interrupt System

17.2 Features

- Interrupt System with support of up to 512 / 1024 service requests
- Support of up to 255 service request priority levels per ICU¹⁾ / Service Provider
- Support of up to 16 ICU / service provider
- Each CPU and DMA module with an dedicated ICU
- ICU service request arbitration independent from other ICUs
- Low latency arbitration - three / four clocks²⁾ from receipt of an service request to sending it to the service provider
- Each Service Request with a dedicated Service Request Node (SRN)
- Each SRN with a programmable 8-bit priority vector¹⁾
- Each SRN can be mapped to one of the implemented ICUs / Service Providers
- SRNs are cleared automatically by hardware on interrupt acknowledge by the configured service provider
- Interrupt System Integrity support
- Four General Purpose Service Requests (GPSR) per CPU that can be used as Software Interrupts (not assigned to peripherals or external interrupts)
- Mechanism to signal General Purpose Service Requests (Software Interrupts) simultaneously to multiple Service Providers (Service Request Broadcast Registers, SRB)

1) Max. 255 of the implemented service requests can be mapped to one CPU

2) Depends on the complexity and the clock frequency of the Interrupt Router. Details for the implementation are described in the chapter Module Implementation

Interrupt Router (IR)

- Priority dependent masking of service requests (for CPUs, related control registers included in the CPUs)
- External Interrupts with filter modes and trigger modes (to e.g. falling edge, rising edge, high or low level). Modes can be configured during runtime¹⁾
- CPU wake up support (service request to CPUx is signalled to SCU to wake up CPUx in case CPUx is in IDLE state)

17.3 Service Request Nodes (SRN)

Each Service Request node (SRN) inside the Interrupt Router module contains a Service Request Control (SRC) Register and interface logic that connects it to the triggering unit outside the Interrupt Router module and to the interrupt arbitration buses inside the Interrupt Router (see also: [Figure 17-1](#)).

17.3.1 Service Request Control Registers

All Service Request Control Registers in the Interrupt Router module have the same format. In general, these registers contain:

- Enable/disable information
- Service Request Set bit and Service Request Clear bit
- Service Request Priority Level vector (8-bit)
- Service provider destination
- Service request status bit
- Software-initiated service request set and reset bits
- Integrity errors signalled to the Safety Monitor Unit (SMU)
- Interrupt Sticky and Overflow bits

Besides being activated by the associated triggering unit through hardware, each SRN can also be set or reset by software via two software-initiated service request control bits.

The description given in this chapter characterizes all Service Request Control Registers of the TC26x B-Step. Information on peripheral module interrupt functions such as enable or request flags is provided in the corresponding sections of the module chapters.

17.3.1.1 General Service Request Control Register Format

Service Request Control Register (SRC)

The description given in this chapter characterizes all Service Request Control Registers of the TC26x B-Step. Information on peripheral module interrupt functions such as enable or request flags is provided in the corresponding sections of the module chapters.

1) External Interrupt logic and related control registers are described in the System Control Unit (SCU) chapter, External Request Unit (ERU)

Interrupt Router (IR)
SRC
Service Request Control Register
(xx_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOV CLR	IOV	SET R	CLR R	SRR	0					ECC		
rh	w	rh	w	rh	w	w	rh	r					rwh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0		TOS		SRE	0									
	r		rw		rw	r									

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority ... FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i> <i>Note: For DMA, SRPN must be =< the highest implemented DMA channel number</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated Other bit combinations are reserved.

Interrupt Router (IR)

Field	Bits	Type	Description
ECC	[21:16]	rwh	<p>ECC</p> <p>The SRC.ECC bit field will be updated by the SRN under the following conditions:</p> <ul style="list-style-type: none"> • write or Read-Modify-Write to SRC[31:0] • write to SRC[15:0] (16-bit write) • write to SRC[15:8] or write to SRC[7:0] (byte write) <p>In case of a 32 bit write to SRC, the ECC bit field will be updated with the calculated ECC, the data written to ECC bit field will be ignored.</p> <p>ECC encoding covers new SRPN, TOS, SRE values and the internal index number of the written SRN. There is no permanent ECC check. ECC check will be done whenever the SRN with a pending interrupt was accepted by the selected (TOS) service provider as next service request to be processed.</p> <p>For a check of the error detection mechanisms ECC errors can be inserted (ECC bit field modified) by: writing directly to the ECC bit field.</p> <ul style="list-style-type: none"> • writing to SRC[23:16] (8-bit write) • writing to SRC[31:16] (16-bit write) <p><i>Note: In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.</i></p>
SRR	24	rh	<p>Service Request Flag</p> <p>0_B No service request is pending 1_B A service request is pending</p>
CLRR	25	w	<p>Request Clear Bit</p> <p>CLRR is required to reset SRR.</p> <p>0_B No action 1_B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.</p>
SETR	26	w	<p>Request Set Bit</p> <p>SETR is required to set SRR.</p> <p>0_B No action 1_B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.</p>

Interrupt Router (IR)

Field	Bits	Type	Description
IOV	27	rh	Interrupt Trigger Overflow Bit The IOV bit is set by HW if a new service request was triggered via interrupt trigger or SETR bit while the SRN has still an pending service request. 0_B No Interrupt Trigger Overflow detected 1_B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit IOVCLR is required to reset IOV. 0_B No action 1_B Clear IOV; bit value is not stored; read always returns 0.
SWS	29	rh	SW Sticky Bit The Software Sticky Bit is set when the SRR bit has been set via the SETR (Request Set Bit). 0_B No interrupt was initiated via SETR 1_B Interrupt was initiated via SETR This bit can be cleared by writing with 1. Writing with 0 has no effect.
SWSCLR	30	w	SW Sticky Clear Bit SWSCLR is required to reset SWS. 0_B No action 1_B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

17.3.1.2 Changing the SRN configuration

All Service Request Nodes are disabled per default. To use a Service Request Node, it has to be configured and enabled by setting the SRC.SRE bit='1'.

The Service Request Nodes can be configured regarding the interrupt service provider target (SRC.TOS) and regarding the service request priority number (SRC.SRPN).

Once an SRN is enabled, the TOS and/or SRPN bit fields can be changed by using the following sequence:

- disable the SRN (SRC.SRE='0') (write to SRC.SRE)
- check that the SRN is disabled (read back SRC.SRE and check for SRE='0')
- check the register LWSRx (read/poll LWSRx, see note below):
- if LWSRx.STAT='1' and LWSRx.SRPN and LWSRx.ECC are equal to the (old) SRC value check again
- if LWSRx.STAT='0' or LWSRx.SRPN or LWSRx.ECC are not equal to the old SRC values anymore go on with the next step (change SRC value)
- change the SRC.TOS and/or SRC.SRPN bit field
- enable the SRN (SRC.SRE='1') (write to SRC.SRE)

The read/poll of LWSRx is mandatory before changing SRC.TOS and/or SRC.SRPN. Otherwise the last interrupt before the SRN disabling could still be taken (enter) for execution in the CPU. It could be acknowledged later by the CPU with the side effect that the first interrupt of the just re-configured SRN can get lost. The time between enter and acknowledge is not deterministic if the ISP is a TriCore. If the ISP is the DMA module, enter and acknowledge are signalled in the same cycle. The 'x' in LWSRx means that the LWSR register related to the TOS setting has to be checked (TOS=0 -> LWSR0, TOS=1 -> LWSR1, ...).

17.3.1.3 Protection of the SRC Registers

The SRC register is write protected via an On Chip Bus Master TAG-ID protection (see [Chapter 17.6.1](#)). This protection is controlled via the Interrupt Router control registers ACCEN10 and ACCEN00.

- SRC[31:16] is write protected by ACCEN00
- SRC[15:0] is write protected by ACCEN10

This implementation allows to define a first group of TAG ID(s) that are allowed to modify the configuration of the Service Request Nodes (TOS, SRPN, SRE), and a second group of TAG IDs that are allowed to modify the Service Request Nodes control bits (set/clear of SW Interrupts, clear of Sticky and Overflow bits)

Write access to SRC[31:16] with access protection violation

When an SRC register is written with a 32 bit data access, only the SRC register parts without ACCEN protection violation will be updated:

Interrupt Router (IR)

- Violation for SRC[31:16] (ACCEN00) and SRC[15:0] (ACCEN10)
- no update of the SRC register, Alarm send to SMU
- Violation for SRC[31:16] (ACCEN00)
- no update of SRC[31:16], SRC[15:0] updated, Alarm send to SMU
- Violation for SRC[31:16] (ACCEN10)
- SRC[31:16] updated, no updated of SRC[15:0], Alarm send to SMU

17.3.1.4 Request Set and Clear Bits (SETR, CLRR)

The SETR and CLRR bits allow software to set or clear the service request bit SRR.

- Writing a 1 to SETR causes bit SRR to be set to 1
- Writing a 1 to CLRR causes bit SRR to be cleared to 0
- Writing a 1 to SETR and CLRR at the same time, SRR is not changed
- The value written to SETR or CLRR is not stored
- Writing a 0 to these bits has no effect
- These bits always return 0 when read

17.3.1.5 Enable Bit (SRE)

The SRE bit enables an interrupt to take part in the arbitration for the selected service provider. It does not enable or disable the setting of the request flag SRR; the request flag can be set by hardware or by software (via SETR) independent of the state of the SRE bit. This allows service requests to be handled automatically by hardware or through software polling.

If SRE = 1, pending service requests are passed on to the designated service provider for interrupt arbitration. The SRR bit is automatically set to 0 by hardware when the service request is acknowledged by the Interrupt Service Provider. It is recommended that in this case, software should not modify the SRR bit to avoid unexpected behavior due to the hardware controlling this bit.

If SRE = 0, pending service requests are not passed on to service providers. Software can poll the SRR bit to check whether a service request is pending. To acknowledge the service request, the SRR bit must then be reset by software by writing a 1 to CLRR.

Note: In this document, 'active source' means an SRN whose Service Request Control Register has its request enable bit SRE set to 1 to allow its service requests to participate in interrupt arbitration.

17.3.1.6 Service Request Flag (SRR)

When set, the SRR flag indicates that a service request is pending. It can be set or reset directly by hardware or indirectly through software using the SETR and CLRR bits. Writing directly to this bit via software has no effect.

Interrupt Router (IR)

SRR can be set or cleared either by hardware or by software regardless of the state of the enable bit SRE. However, the request is only forwarded for service if the enable bit is set. If SRE = 1, a pending service request takes part in the interrupt arbitration of the service provider selected by the device's TOS bit field. If SRE = 0, a pending service request is excluded from interrupt arbitrations.

SRR is automatically reset by hardware when the service request is acknowledged and serviced. Software can poll SRR to check for a pending service request. SRR must be reset by software in this case by writing a 1 to CLRR.

Note: Clearing a pending service request flag SRR and enabling the corresponding service request node (SRN) should be done in two steps / two writes: first clear the SRR flag (SRC.CLRR), then enable the (SRC.SRE).

17.3.1.7 Type-Of-Service Control (TOS)

There are multiple service providers for service requests in the TC26x B-Step, e.g. two CPU and the DMA module. The TOS bit field is used to select to which of the available interrupt service provider an service request has to be forwarded.

Note: Before modifying an TOS or an SRPN bit field, the corresponding SRN must be disabled (SRE = 0).

17.3.1.8 Service Request Priority Number (SRPN)

The 8-bit Service Request Priority Number (SRPN) indicates the priority of a service request with respect to other sources requesting service from the same service provider, and with respect to the priority of the service provider itself.

SRPN Rules:

- Each active source selecting the same service provider must have a unique SRPN value to differentiate its priority
- Active sources selecting different service provider could use same SRPN values
- If a source is not active – meaning its SRE bit is 0 – no restrictions are applied to the service request priority number
- The SRPN is used by service providers to select an Interrupt Service Routine (ISR) or DMA channel to service the request

Service Provider is a CPU:

ISRs are associated with Service Request Priority Numbers by an Interrupt Vector Table located in each CPU. This means that the CPU Interrupt Vector Table is ordered by priority number. This is unlike traditional interrupt CPU architectures in which their interrupt vector tables are ordered by the source of the interrupt. The CPU Interrupt Vector Tables allow a single peripheral to have multiple priorities for different purposes.

Interrupt Router (IR)

Note: For a CPU, the SRPN value of 00_H is a special value and must not be used for service requests mapped to a CPU.

Note: TC1.6E and TC1.6P CPUs are providing a flexible interrupt table alignment with a configurable vector spacing of 8 byte or 32 byte. Pls. see also CPU chapter.

Service Provider is the DMA:

The complete SRPN number is used for the arbitration of service requests the DMA module. Within the module, the DMA channels are associated with the Service Request Priority Number by the SRPN LSBs of the SRPN.

Only the SRPN numbers 0 ... max_channel_number will result in a trigger of the related DMA channel. SRPN numbers > max_channel_number do not result in a DMA channel trigger nor any other signalling.

Examples:

- In case of an 16 channel DMA module the SRPN number 00H will trigger the channel 0, 07H will trigger the channel 7. All SRPN > 0FH will not result in a channel trigger.
- In case of an 64 channel DMA module the SRPN number 00H will trigger the channel 0, 17H will trigger the channel 23 an 3FH. will trigger channel 64. All SRPN > 3FH will not result in a channel trigger.

17.3.1.9 ECC Encoding (ECC)

The SRC.ECC bit field will be updated by the SRN under the following conditions:

- write or Read-Modify-Write to SRC[31:0]
- write to SRC[15:0] (16-bit write)
- write to SRC[15:8] or write to SRC[7:0] (byte write)

In case of a 32-bit write or a Read-Modify-Write to the SRC, the ECC bit field will be updated with the calculated ECC, the data written to ECC bit field will be ignored.

ECC encoding covers the new values of SRC.SRPN, SRC.TOS, SRC.SRE and the internal 10 bit index number of the written SRN.

There is no permanent ECC check. ECC check will be checked whenever the SRN with an pending service request was accepted by the selected (TOS) service provider as next service request to be processed.

For a check of the error detection mechanisms ECC errors can be inserted (ECC bit field modified) by:

- writing to SRC[23:16] (byte write)
- writing to SRC[31:16] (16-bit write)

Note: In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.

ECC Code

The ECC code used for the IR Error Detection mechanism is a Hsiao 22_5 code with DED (double error detection) capability:

```
GEN_ENC22_5 : if (word_width_g/(nb_mems_g*ecc_granularity_g)) =
22 and (nb_check_bits_g = 5) generate
    cmr22_5: for i in 0 to nb_check_bits_g - 1 generate
        CODE_MATRIX_ROWS(i)    <= code_rows_22_5(i);
    end generate;
end generate;
```

```
type rows_22_5_t is array (4 downto 0) of std_ulogic_vector(21
downto 0);
constant code_rows_22_5 : rows_22_5_t :=
("0001001011001011011011",
"0010010101010101101101",
"0100100110100110110110",
"1000111000111000111111",
"1111000000111111000111");
```

17.3.1.10 Interrupt Trigger Overflow Bit (IOV)

The IOV bit is set by HW if both conditions are true:

- service request is pending
- a new service request is triggered via interrupt trigger or SETR bit

17.3.1.11 Interrupt Trigger Overflow Clear Bit (IOVCLR)

The Interrupt Trigger Overflow Clear Bit can be used to clear the IOV bit. The IOV bit is cleared by writing a '1' to the IOVCLR bit.

17.3.1.12 SW Sticky Bit (SWS)

The Software Sticky Bit is set when the SETR (Request Set Bit) is written with 1.

17.3.1.13 SW Sticky Clear Bit (SWSCLR)

The Software Sticky Clear Bit can be used to clear the SWS bit. The SWS bit is cleared by writing a '1' to the SWSCLR bit.

17.4 Interrupt Control Unit (ICU)

The Interrupt Router module includes one ICU per service provider (CPUs and DMA module) where each ICU is related to one service provider. SRNs can be mapped to one of the ICUs via the SRNs SRCx.TOS register bit field (see also: [Figure 17-1](#)).

The Interrupt Control Units (ICU):

- Manages the arbitration among competing service requests from SRNs that are mapped to the ICU
- Provides the winner of the arbitration round to the service provider
- Receives the information from the service provider which service request was accepted
- Checks the accepted service request information (ECC check)
- Signals integrity errors to the Safety Monitor Unit (SMU)
- Manages the clearing of acknowledged service requests in the related SRNs

Note: In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.

17.4.1 ICU Control Registers

This section describes the Interrupt Control Unit (ICU) registers. Each ICU includes two control registers:

- Latest Winning Service Request register (LWSR) provides information about the winner of the last service request arbitration round
- Last Acknowledged Service Request register (LASR) provides information about the last service request that was accepted by the service provider

17.4.1.1 Latest Winning Service Request Register (LWSR)

Latest Winning Service Request (LWSR)

The Latest Winning Service Request register provides information about the winner of the last arbitration round. The register bit fields are representing what is provided by the ICU to the Interrupt Service Provider.

LWSR

Latest Winning Service Request (xx_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STAT		0				ID									
r		r				r									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC						0		PN							
r						r		r							

Field	Bits	Type	Description
PN	[7:0]	r	Latest Winner Priority Number This bit field shows the Priority Number of the service request that won the last arbitration round. This bit field is only valid if STAT is set to 1
ECC	[15:10]	r	Latest Winner ECC This bit field shows the ECC field (SRN.ECC) that was transferred from the last winning SRN to the ICU. This bit field is only valid if STAT is set to 1. <i>Note: In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.</i>
ID	[25:16]	r	Latest Winner ID This bit field shows the ID number of the last winning SRN. This bit field is only valid if STAT is set to 1

Interrupt Router (IR)

Field	Bits	Type	Description
STAT	31	r	<p>LWSR Register Status</p> <p>The STAT register indicates if the PN, ECC and ID bit fields are still valid. They are still valid if the interrupt from the SRN identified by ID is still pending. If the ICU does not have a winner because no interrupt is pending or not yet arbitrated than it clears the STAT bit.</p> <p>0_B LWSR bit fields are not valid 1_B LWSR bit fields are valid</p>
0	[30:26] , [9:8]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

17.4.1.2 Last Acknowledged Service Request Register (LASR)

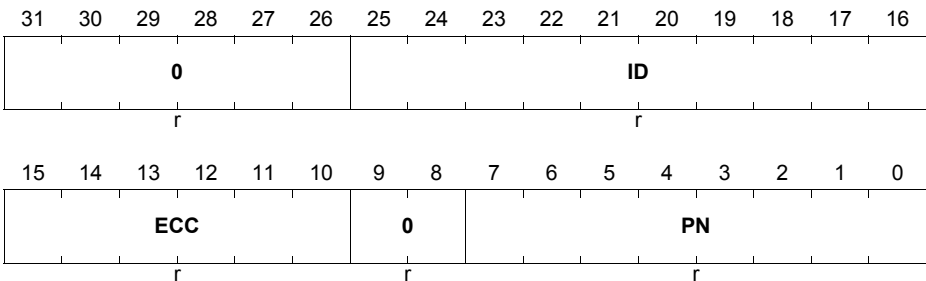
Last Acknowledged Service Request (LASR)

The Last Acknowledged Service Request register is representing the information that was sent by the Interrupt Service Provider together with the last acknowledge.

LASR

Last Acknowledged Service Request (xx_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
PN	[7:0]	r	Last Acknowledged Service Request Priority Number This bit field shows the Priority Number of the last acknowledged service request
ECC	[15:10]	r	Last Acknowledged Interrupt ECC This bit field shows the ECC value of the last acknowledged service request, as send by the service provider with acknowledge <i>Note: In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.</i>
ID	[25:16]	r	Last Acknowledged Interrupt SRN ID This bit field shows the ID number of the last acknowledged service request, as send by the service provider with acknowledge
0	[31:26], [9:8]	r	Reserved Read as 0; should be written with 0.

17.4.1.3 Error Capture Register (ECR)

Error Capture Register (ECR)

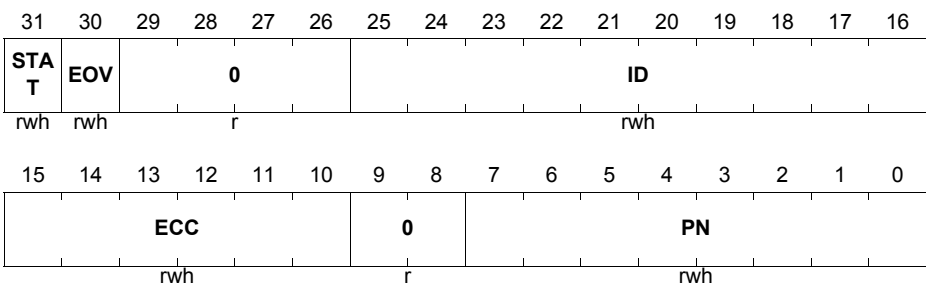
The Error Capture Register captures the Last Acknowledged Service Request (LASR) register contents when an ECC error was detected by the ICU. The ECR shows always the last ECR contents where an ECC error was detected. Software can clear the ECR contents by writing to the ECR. Error Status (STAT) and Error Overflow (EOV) bits can be used as error handling mechanism and indication that error information where lost.

If ECR.EOV is cleared by SW, ECR.EOV must be cleared together with ECR.STAT. If a new error is detected in parallel to the ECR.EOV clear than ECR.EOV is set again by hardware while ECR.STAT is cleared.

Note: In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.

ECR

Error Capture Register

 (xx_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
PN	[7:0]	rwh	Service Request Priority Number This bit field shows the priority number of the last service request where an error was detected. Bit field can be modified by writing to it.
ECC	[15:10]	rwh	Service Request ECC This bit field shows the ECC of the last service request where an error was detected. Bit field can be modified by writing to it. This bit field can be modified by: <ul style="list-style-type: none"> • writing to SRC[23:16] (byte write) • writing to SRC[31:16] (16-bit write)

Interrupt Router (IR)

Field	Bits	Type	Description
ID	[25:16]	rwh	Service Request Node ID This bit field shows the ID of the last service request where an error was detected. Bit field can be modified by writing to it
EOV	30	rwh	Error Overflow Bit The bit is set if an ECC error was detected by the ICU while ECR.STAT='1' (Error Overflow situation). 0 _B No Error Overflow situation detected 1 _B Error Overflow situation detected This bit can be cleared by writing with 1. Writing with 0 has no effect. If this bit is cleared, it must be cleared together with the STAT bit.
STAT	31	rwh	Error Status Bit The Error Status Bit is set whenever an ECC was detected by the ICU. 0 _B No ECC error detected 1 _B ECC error detected This bit can be cleared by writing with 1. Writing with 0 has no effect.
0	[29:26] , [9:8]	r	Reserved Read as 0; should be written with 0.

17.5 General Purpose Service Requests, Service Request Broadcast

The Interrupt Router module provides multiple groups of General Purpose Service Requests (GPSR) and a mechanism to trigger multiple Service Requests of a GPSR group in parallel, by software. The GPSR can be used as Software Interrupt.

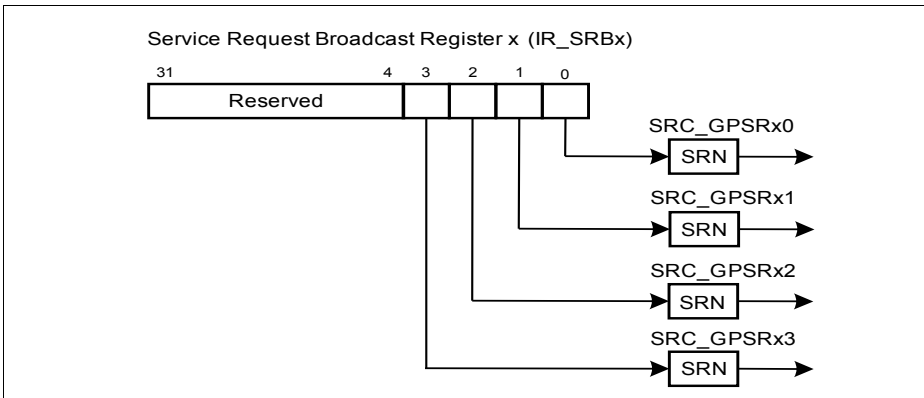


Figure 17-2 Structure of a General Purpose Service Request Group and the related Broadcast register (Example for SRC_GPSRx0, SRB0)

17.5.1 General Purpose Service Requests (GPSRxy)

The Interrupt Router module provides multiple groups of General Purpose Service Requests:

- Each General Purpose Service Request Group consists of four Service Request Nodes. Nodes that can be used as Software Interrupt
- The General Purpose Service Requests can be configured and controlled via the related Service Request Control registers SRC_GPSRxy¹⁾
- The GPSR are not mapped to module service request triggers so they can only be used a SW trigger
- A General Purpose Service Request xy can only be triggered by writing '1' to the related SRC_GPSRxy.SETR¹⁾ bit or by writing a '1' to the related Service Request Broadcast register bit SRBx[y]

17.5.2 Service Request Broadcast Registers (SRBx)

There is one Service Request Broadcast register (SRBx) implemented for each General Purpose Service Request Group (GPSRxy¹⁾).

The Service Request Broadcast register x can be used to trigger multiple Service Requests within the SRC_GPSRxy¹⁾ group in parallel.

- SRBx is always read as 0
- Writing '1' to SRBx[y] triggers the service request GPXRxy (y=3:0)
- Writing '1' to SRBx[31:4] has no effect.

1) SRC_GPSRxy: x = group number; y= number of interrupt within the group, y=0:3

17.6 System Registers

The Interrupt Router module includes the standard set of TC26x B-Step system registers. **Figure 17-3** shows these system registers which include registers for:

- Register access protection
- Module Clock Control¹⁾
- Module Kernel Reset¹⁾
- OCDS Control and Status Register

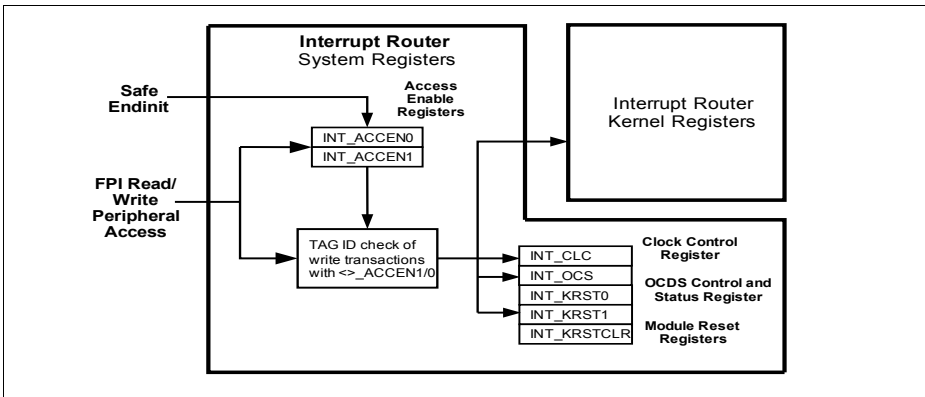


Figure 17-3 Interrupt Router System Registers

17.6.1 Register Access Protection (ACCEN1/0)

The Interrupt Router module provides a master TAG ID based write access protection as part of the AURIX safety concept. Each on chip resource with direct or indirect bus master capability has a unique master TAG ID that can be used to identify the master of an on chip bus transaction (see also chapter On Chip Bus Systems).

The SRC register is write protected via an On Chip Bus Master TAG-ID protection (see **Chapter 17.6.1**). This protection is controlled via the Interrupt Router control registers ACCEN10 and ACCEN00.

TAG ID based protection means that the support of write transactions to the Interrupt Router control registers can be enabled / disabled for each master TAG ID individually. For a disabled master TAG ID, write access will be disconnected with error acknowledge, read access will be processed (see also **Figure 17-3**).

¹⁾ The Interrupt Router module does not support the features of this system register

Interrupt Router (IR)

The register access protection is controlled via the registers INT_ACCEN1 and INT_ACCEN0 where each bit is related to one encoding of the 6 bit On Chip Master TAG ID.

The INT_ACCEN1/0 access protection controls the write access to all Interrupt Router control and system registers with the exception of INT_ACCEN1/0. INT_ACCEN1/0 are only Safety Endinit protected.

After reset, all access enable bits and access control bits are enabled. INT_ACCEN1/0 have to be configured and checked to bring the system into a safe state.

17.6.2 Kernel Reset Registers (KRST1/0, KRSTCLR)

The Interrupt Router module does not include the kernel reset registers (KRST1, KRST0, KRSTCLR).

Note: The Interrupt Router module does not support a module kernel reset.

17.6.3 Clock Control Register (CLC)

The Interrupt Router module does not include the module clock control (CLC).

Note: The Interrupt Router module does not support the Clock Control register functionality which means that the Interrupt Router module clock can not be disabled by the CLC register.

17.6.4 OCDS Control and Status Register (OCS)

The Interrupt Router module does not include OCDS Control and Status (OCS) register.

Note: The Interrupt Router module does not support the OCS register functionality.

17.7 Arbitration Process

Each ICU in the interrupt module has its own interrupt bus. Each Service Request Node (SRN) can be directed to one service provider by mapping it via the SRC.TOS bit field setting to the related ICU / Interrupt Bus.

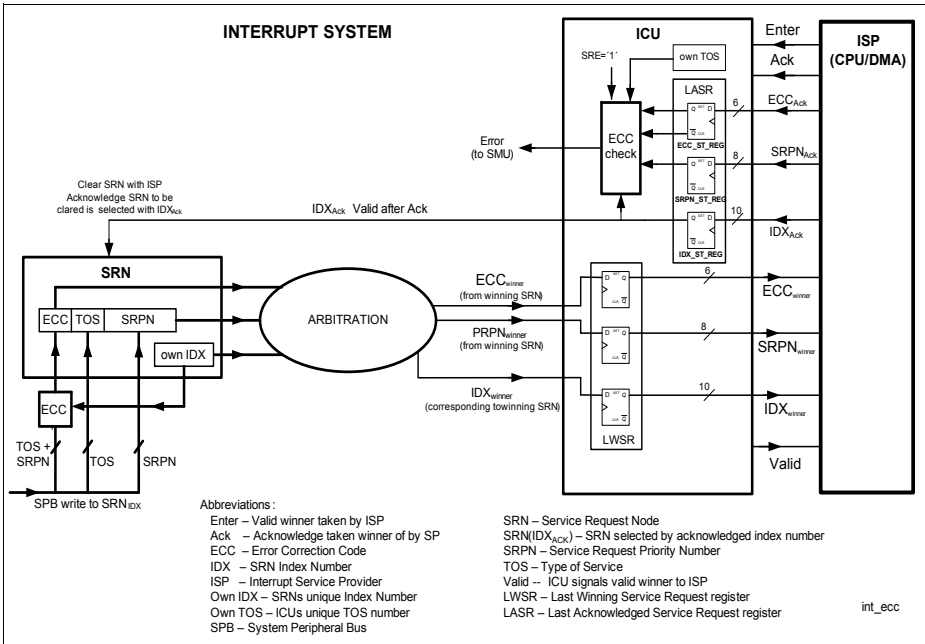


Figure 17-4 Interrupt System Arbitration Scheme Overview

With a first pending service request, an interrupt bus is starting a first arbitration process. The arbitration process on one interrupt bus is repeated as long at least one service request is pending. The related Interrupt Control Unit provides the service request that won the last arbitration process.

This means: when the ICU provides a service request to the service provider, and a new service request with a higher priority comes in, the ICU will offer the new service request after the next arbitration process.

The arbitration process implemented in the TC26x B-Step uses 3 system peripheral bus clock cycles to determine the pending service request with the highest priority number, SRPN. The exact number of the implementation is described in the Module Implementation chapter.

In an arbitration process, the interrupt bus compares the SRC.SRPN bit fields of all pending Service Request Nodes, mapped to this interrupt bus via SRC.TOS setting.

Interrupt Router (IR)

During the arbitration process, the pending service request with the highest priority number is identified as winner and the related SRN Service Request Control register bit field values SRPN, ECC and the Index of the SRN are provided to the ICU. The ICU provides these (SRPN, ECC, SRN Index) to the service provider. The ICU does an ECC check when it gets these information back from the service provider with an acknowledge. The ECC check is done with the received values: ECC, SRPN, SRN Index Number, SRE bit assumed to be '1' (SRN enabled) and the TOS number of the ICU.

The Interrupt Router module signals detected errors to the Safety Management Unit (one bit in the SMU, covers errors from all SRN and ICUs).

Note: In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.

17.7.1 Number of Clock Cycles per Arbitration Process

The Interrupt Router implementation is can be configured regarding the number of:

- supported service requests (Service Request Nodes, SRN, up to 1024)
- supported service providers (Interrupt Control Units, ICUs)
- clock cycles per service request arbitration (3 SPB clock cycles)

The characteristics of the Interrupt Router module implemented in an AURIX product is described in the Module Implementation sub-chapter.

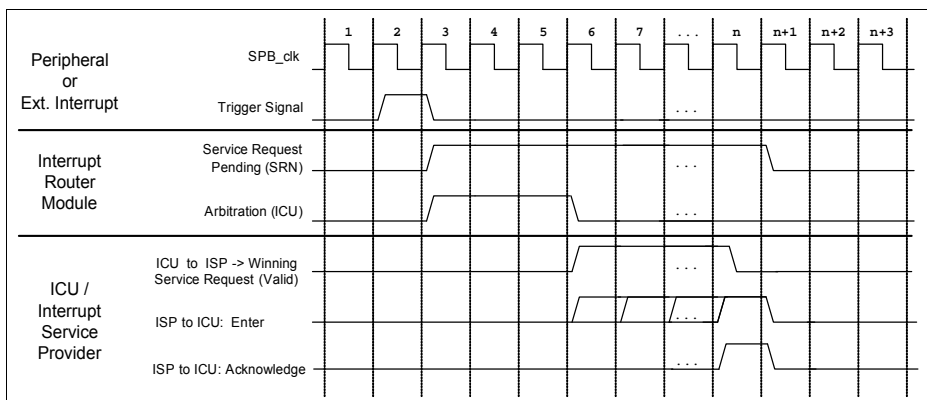


Figure 17-5 Interrupt System Timing (Schematic Overview, 3 Cycle Arbitration)

Figure 17-5 shows the interrupt timing of an Interrupt Router implementation with 3 cycle arbitration.

Cycle 1: No service request for the ICU is pending (therefore no arbitration round)

Cycle 2: One module is triggering a service request by sending a pulse to the related SRN in the Interrupt Router module.

Interrupt Router (IR)

Cycle 3-5: Arbitration among all pending service requests to the ICU

Cycle 6: ICU provides winning service request to the service provider (SRPN, ECC, SRN Index)

Cycle 7 - n-1: ICU re-arbitrates whenever a new service request from another SRN is pending, provides new winning service request to ICU if there is a new pending one with a higher SRPN number (higher priority)

Cycle 6- n-1: Interrupt Service Provider takes the information of the latest winning SRN (Enter)

Cycle n: Service provider acknowledges service request (provides SRPN, ECC, SRN Index informations of the acknowledged service request). ICU changes signals to ICU to 'no valid service request available' in the same clock cycle.

Cycle n+1: ICU does an ECC check of the acknowledge information (SRPN, ECC, SRN Index, SRE='1', TOS number of the ICU. If mismatch -> Integrity Error signalled to SMU/ SRPN, ECC and Index are captured in the ECR)

Cycle n+2: ICU selects the acknowledged SRN (selected by SRN Index). Selected SRN checks acknowledged SRPN and ECC with own SRC SRPN/ECC values (if mismatch -> Integrity Error signalled to SMU)

Cycle n+3: If at least one service request to the ICU is pending: new arbitration among all pending service request to the ICU

17.7.2 Service Request Acknowledge

When a Service Provider starts with the execution of a service request that was provided by the ICU, the Service Provider sends an acknowledge to the ICU. In parallel to the acknowledge, the Service Provider sends the informations about the executed service request back to the ICU (SRPN, ECC, SRN Index Number).

In the same clock cycle, the Service Provider sends an acknowledge, the ICU changes to 'no service request available': the ICU does not provide an arbitration winner to the service provider. This behavior of the ICU ensures that a just acknowledged service request is not provided again before the SRN was re-set (see [Figure 17-5](#)).

17.7.3 Handling of detected ECC Errors

The ICU does an ECC check of the informations it receives with the acknowledge from the Service Provider and the TOS number of the ICU itself. Assumption for the SRE bit is '1' (SRN was enabled). ECC in the SRN is covering the SRC bit field values: SRC.SRPN, SRC.SRN Index, SRC.SRE and SRC.TOS. (see [Figure 17-5](#))

Note: In the current implementation the ECC code is only used for error detection. Detected errors are reported to the SMU but not corrected.

If the ICUx detects an ECC error:

Interrupt Router (IR)

- ICUx captures the ECC, SRPN and the Index that showed an ECC error in the Error capture registers (ECRx), sets the ECRx.STAT bit and the ECRx.EOV bit if the STAT bit is still set.
- ICU signals the error via the IR error signal to the TC26x B-Step Safety Management Unit (SMU). SMU forwards this information to a CPU (if enabled).
- ICUx clears the service request in the SRN (selected by the Index)
- The CPU that received the 'ECC error detected in Interrupt Router' information via SMU can identify the ICU via the Error Status bits (ECRx.STAT='1'), read out the related service request information and clear the ECRx.STAT bit by writing with '1'. The CPU can also identify if one or multiple ECC errors were detected by this ICU via the Error Overflow bit (ECRx.EOV='1').

17.8 Usage of the TC26x B-Step Interrupt System

The following sections provide a short description of the Service Provider interfaces to the Interrupt Router ICUs.

Note: All ICU sub-modules in the Interrupt Router have the same functionality.

17.8.1 CPU to ICU Interface

Each CPU has one Interface is connected to one ICU of the Interrupt Router module. The CPU ICU interface consists of a register set where it takes over the informations of a service request provided by the ICU (SRPN, SRN Index, ECC). The informations will be send back to the ICU when the CPU acknowledges the provided service request.

The CPU ICU interface contains an Interrupt Control Register (ICR) that holds the current CPU priority number (CCPN), the global interrupt enable/disable bit (IE) and the pending interrupt priority number (PIPn). Further details of the CPU ICU interface and the CPU handling of interrupts can be found in the CPU chapter.

17.8.2 DMA to ICU Interface

The DMA module has one interface where it is connected to one ICU of the Interrupt Router module.

The DMA takes over the service request informations for the ICU, triggers internally the addressed channel and acknowledge it immediately to the ICU where the related SRN is cleared.

The DMA to ICU interface consists of a register set where it takes over the information of a service request provided by the ICU (SRPN, SRN Index, ECC), The DMA sends them back to the ICU in the next clock cycle with an acknowledge.

The DMA channel priority scheme is identical to the scheme of the interrupt system: higher SRPN number -> higher service request priority, higher DMA channel number -> higher priority of the DMA channels where channel 0 has the lowest priority within the DMA.

17.8.3 Software-Initiated Interrupts

Software can set the service request bit (SRR) in any SRN by writing to its Service Request Control Register. Thus, software can initiate service requests that are handled by the same mechanism as hardware initiated service requests.

After the SRR bit is set in an SRN, there is no way to distinguish between a software initiated service request and a hardware initiated service request. For this reason, software should only use SRNs and interrupt priority numbers that are not being used for hardware initiated service requests.

Interrupt Router (IR)

The TC26x B-Step contains groups of General Purpose Service Request SRNs per CPU that support software-initiated interrupts. One group per implemented TriCore CPU, each group including four SRNs. These SRNs are not connected to internal or external hardware trigger signals and can only be used as software interrupts / software initiated service requests. These SRNs are called General Purpose Service Requests Nodes (SRC_GPSRxy, x=group number, y=0-3).

Additionally, any otherwise unused SRN can be employed to generate software interrupts.

17.8.4 External Interrupts

Four SRNs (Int_SCUSRC[3:0]) are reserved to handle external interrupts. The setup for external GPIO port input signals (edge/level triggering, gating etc.) that are able to generate an interrupt request is controlled in the External Request Unit (ERU). The ERU functionality is described in detail in the SCU chapter.

17.9 Use Case Examples

This section shows a use case for the interrupt system and a use case for the OTGS.

17.9.1 Use Case Example Interrupt Handler

This section explains how to organize the TriCore interrupt vector table. When an interrupt is accepted by the TriCore, the entry address into the interrupt vector table is calculated by the base interrupt vector table pointer TriCore register BIV and the priority number of that interrupt (PIPn). The TriCore TC1.6P and TC1.6E architecture offers the possibility to configure the vector spacing per entry to either 32 Byte (see Figure below a (Figure 17-6) or 8 Byte (see Figure below b (Figure 17-6)). As a third option the vector table can be reduced to a single entry by masking the PIPn (see Figure below c (Figure 17-6)).

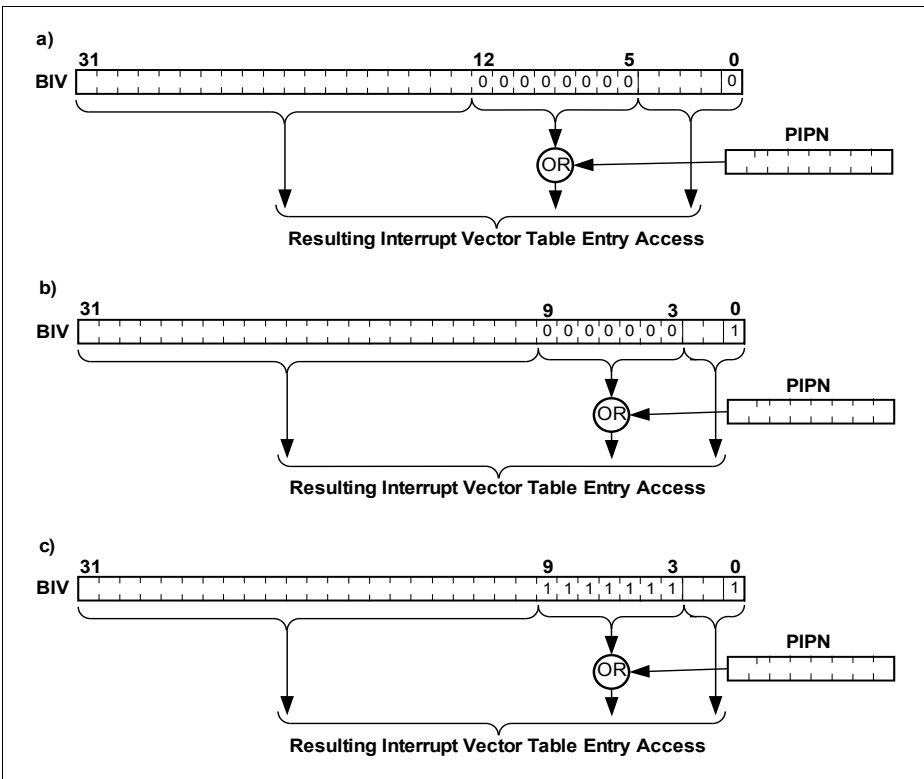


Figure 17-6 Interrupt Vector Table Address Calculation for a) 32 Byte b) 8 Byte vector entry or c) single entry

By using the 32 Byte configuration small interrupt routines can be implemented directly into the vector table. They can even span multiple vector entries (see also TriCore Architecture Manual). This type of fast interrupt handling is useful, if the vector table can

Interrupt Router (IR)

be located into the TriCore program side memory. The 8 Byte configuration reduces the vector table size. Each vector entry contains only a jump instruction or a call and return as 16-bit opcode instruction. The TriCore compiler supports this kind of interrupt vector table generation by keywords or functions. A minimum vector table can be configured if the BIV mask the PIPN so that any interrupt address calculation results in the same address. E.g.

```
__mtcr(BIV,0x80000001 | 0xFF<<3); // move to core register BIV
```

This configures the BIV register to use a common, single entry where a function interruptHandler is located to branch to the specific interrupt routine by using an array of function pointers. If a pointer to the array is used the array could be switched quickly.

Step description to initialize and install interrupts:

(Line 1) define ISR pointer array. Max. 255 interrupts possible.

(Line 2) define pointer which points to the start of the isr_pointer_array.

(Line 3) start of function interruptHandlerInstall. This function installs the interrupts in the array. Necessary information are the interrupt priority and ISR entry address.

(Line 4) This line stores the ISR entry address in the array.

(Line 5 and 6) This function branches to the specific interrupt routine and gets called immediately after an interrupt occurs.

(Line 7) This line gives the return command, after the ISR has been processed.

C Code Example to initialize and install interrupts:

```
(1) void (*isr_pointer_array[256])(void);
(2) void (**isr)(void) = isr_pointer_array;
(3) void interruptHandlerInstall(long int SRprio, long int addr){
(4) *isr_pointer_array[SRprio]=addr;
}
(5) void interruptHandler(void){
(6) isr[__mtcr(ICR) & 0xFF]();
(7) asm (" rfe"); // return from event
}
```

The interrupt entry addresses are stored in a data array instead of encoding the values into the instructions. The function interruptHandlerInstall organizes the installation of the interrupts in that array (see the application of this interrupt handler in a module e.g. use case example in STM chapter). This kind of vector table generation offers sometimes more flexibly than the 8 Byte configuration and does not require any specific compiler support for interrupts.

Note: Before an interrupt is able to occur, the interrupt system has to be globally enabled. The Interrupt Control Register (ICR) holds the global interrupt enable bit

Interrupt Router (IR)

(ICR.IE) which enables the CPU service request system. Most compiler support the attribute (or similar):

`__enable();`

to set this bit. (See also Architecture Manual for more details)

17.10 Module Implementation

17.10.1 Characteristics of TC26x B-Step Interrupt Router Module

The Interrupt Router module is implemented with the following characteristics:

- Number of clock cycles per arbitration process: 3
- Number of Interrupt Control Units (ICU): 3
- Number of General Purpose Service Request groups: 2

17.10.2 Mapping of TC26x B-Step Module Service Request Triggers to SRNs

All TC26x B-Step module service requests are mapped to one Service Request Node in the Interrupt Router.

Each SRN has one unique SRN Index Number within the Interrupt Router module.

The Interrupt Router module has one 511 / 1024 interrupt trigger input vector where the interrupt trigger input vector bit [x] is related to the SRN with the SRN Index Number x¹⁾. This means that a trigger pulse on interrupt trigger input vector bit [x] will trigger the SRN [x].

1) The Interrupt Router register overview table shows for all TC26x B-Step module service requests the SRC registers with its address offset and the related SRN Index number

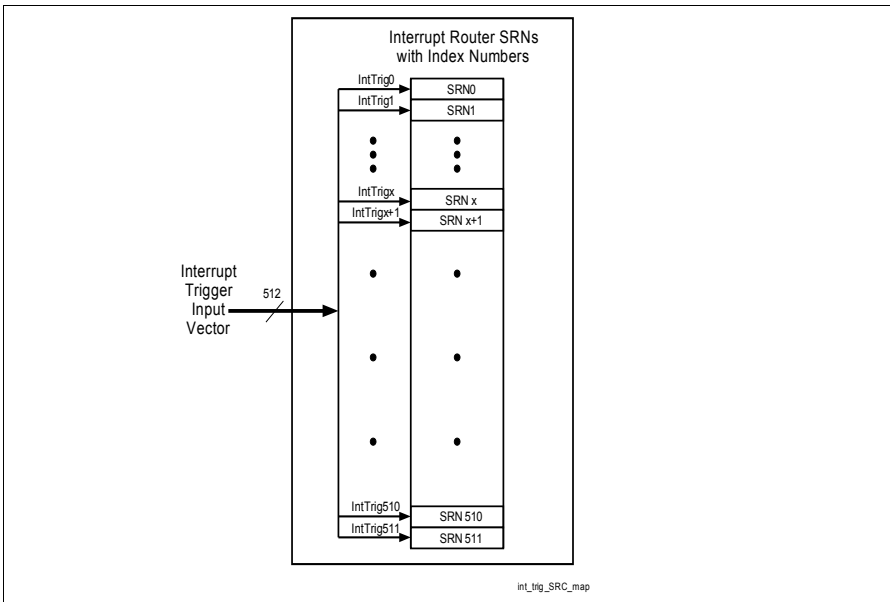


Figure 17-7 Mapping of Module Interrupt Trigger to SRN (Index Numbers)

17.10.2.1 Mapping of Service Request Control Registers

The address of SRC registers related to one module instance is identical over the whole Aurix family (e.g. interrupts of SPI0).

Each SRN has one unique SRN Index Number within the Interrupt Router module.

The number of the implemented SRNs for each Aurix device is adapted the devices feature set. All SRNs in an Interrupt Router module have consecutive Index Numbers (e.g.: if 256 SRNs are implemented, Index Numbers are 0 - 255). Each SRN has one unique SRN Index Number within the Interrupt Router module.

Example:

In a high end device with many modules/module instances SPI0 interrupts might be mapped to the SRNs with Index Numbers 180 to 185, related SRC can be accessed with address offset 0x2F0 to 0x2F5

In order to fulfill both requirements (same address for SPI0 related SRCs in all AURIX devices but reduced number of SRNs in smaller family devices with consecutive Index Numbers, the Interrupt Router module includes an address mapping shell (**Figure 17-8**). This address mapping shell re-maps the fixed SRC addresses of e.g. SPI0 of the above example to the SRN in the Interrupt Router module¹⁾.

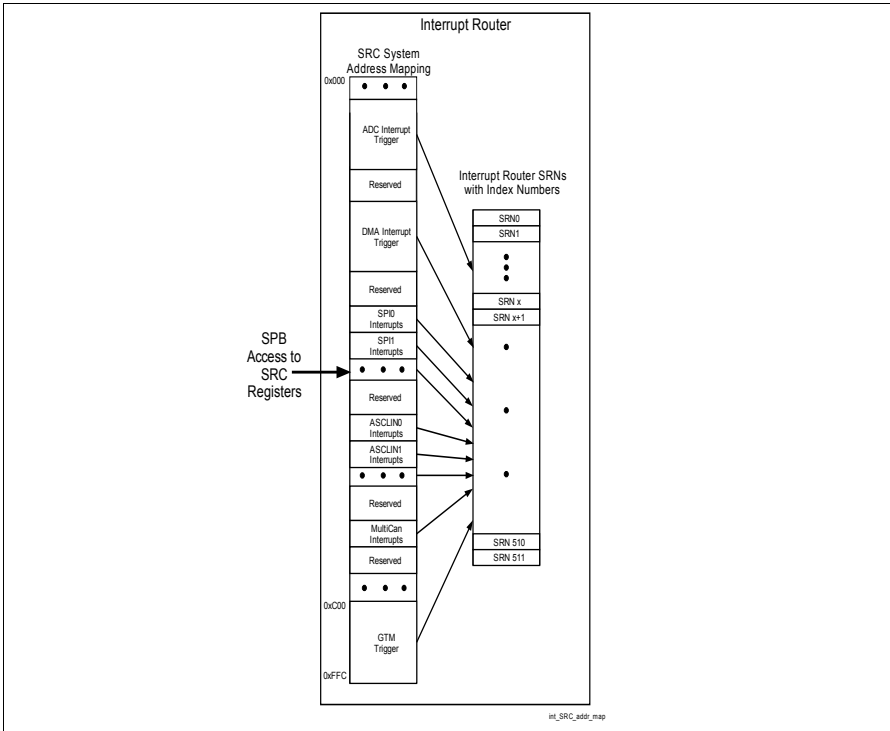


Figure 17-8 Mapping of SRC System Address to SRNs

17.10.2.2 Interrupts related to the Debug Reset

For software debug purposes the AURIX devices require some service request nodes related only to the Debug Reset. These SRNs keep its SRC register contents and the pending service request status in case of a non Debug Reset (e.g. an Application Reset). In combination with other debug reset related debug logic (e.g. breakpoint logic) this allows customer SW to debug situations that result in an application reset and after an application reset.

17.10.2.3 Timing characteristics of Service Request Trigger Signals

The Interrupt Router is clocked with the System Peripheral Bus (SPB) clock

Rules for the TC26x B-Step module interrupt / Service Request trigger signals to the IR:

- Trigger signals must be synchronous to SPB clock
- IR trigger inputs re edge sensitive (positive clock edge)

Interrupt Router (IR)

- Trigger signal pulse with min. high length of one SPB clock cycle, high pulse length can be > 1 SPB clock cycle
- Debug related trigger signal pulse should be kept high by the related TC26x B-Step module until the trigger was processed

Interrupt Router (IR)

17.11 Interrupt Router System and Module Registers

Figure 17-9 shows all registers associated with the Interrupt Router module in the TC26x B-Step device. The Interrupt Router allocates two address ranges:

- 2 * 256 byte address range covering the Interrupt Router system registers, ICU control registers and OTGM registers (Table 17-1)
- 8 KByte address range covering the Service Request Control registers (Table 17-3)

List of used Reset Class abbreviations:

- Reset Class 1 -> Debug Reset (see description in the chapter SCU / Reset Types)
- Reset Class 3 -> Application Reset (see description in the chapter SCU / Reset Types)

Note: A violation of the access protection will not be executed (e.g. a write to a 'Px'/ACCEN protected register from a disabled master). In this case an access protection error is signaled to the SMU. Beside this signaling to the SMU, no other error, interrupt or trap is generated.

Interrupt Router Module Registers

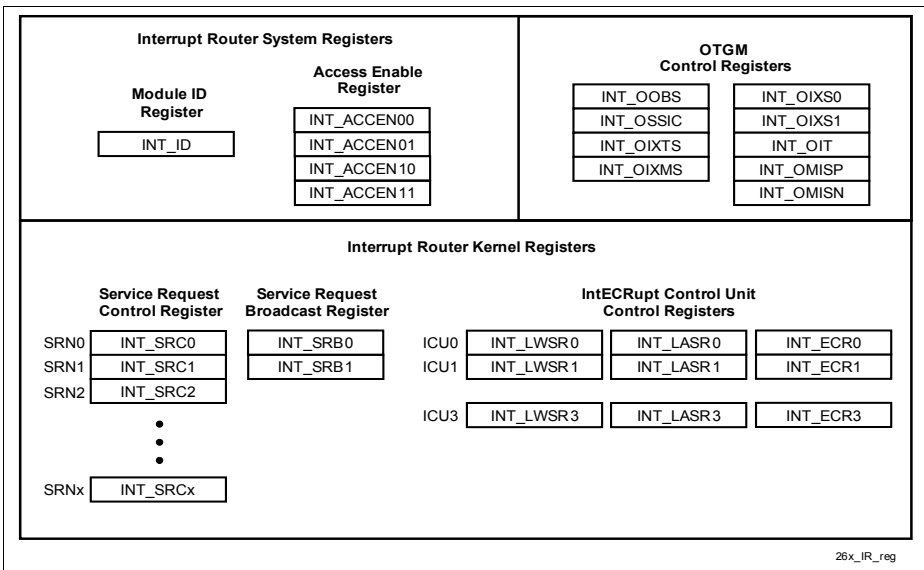


Figure 17-9 Interrupt Router module registers

Table 17-1 Registers Address Space - System, OTGM and ICU Control Registers

Module	Base Address	End Address	Note
INT	F003 7000 _H	F003 7FFF _H	

Table 17-2 Registers Overview - System, OTGM and ICU Control Registers

Short Name	Description	Offset Addr.	Access Mode		Reset Class	Description See
			Read	Write		
-	Reserved	000 _H -004 _H	U, SV	BE	-	-
INT_ID	Interrupt Router Module Identification Register	008 _H	U, SV	BE	-	Page 17-37
-	Reserved	00C _H	U, SV	BE	-	-
INT_SRB0	Service Request Broadcast Register 0	010 _H	U, SV	SV, P0	3	Page 17-38
INT_SRB1	Service Request Broadcast Register 1	014 _H	U, SV	SV, P0	3	Page 17-38
-	Reserved	018 _H -07F _H	U, SV	BE	-	-
INT_OOBS	OTGM OTBG0/1 Status Register	080 _H	U, SV	BE	1	Page 17-44
INT_OSSIC	OTGM SSI Control Register	084 _H	U, SV	SV	1	Page 17-45
INT_OIXTS	OTGM IRQ MUX Trigger Set Select Register	088 _H	U, SV	SV	1	Page 17-45
INT_OIXMS	OTGM IRQ MUX Missed IRQ Select Register	08C _H	U, SV	SV	1	Page 17-46
INT_OIXS0	OTGM IRQ MUX Select 0 Register	090 _H	U, SV	SV	1	Page 17-47
INT_OIXS1	OTGM IRQ MUX Select 1 Register	094 _H	U, SV	SV	1	Page 17-47
-	Reserved	098 _H -09C _H	U, SV	BE	-	-

Interrupt Router (IR)
Table 17-2 Registers Overview - System, OTGM and ICU Control Registers

Short Name	Description	Offset Addr.	Access Mode		Reset Class	Description See
			Read	Write		
INT_OIT	OTGM IRQ Trace Register	0A0 _H	U, SV	SV	1	Page 17-48
INT_OMISP	OTGM MCDS I/F Sensitivity Positive Edge Register	0A4 _H	U, SV	SV	1	Page 17-49
INT_OMISN	OTGM MCDS I/F Sensitivity Negative Edge Register	0A8 _H	U, SV	SV	1	Page 17-49
-	Reserved	0AC _H -0EF _H	U, SV	BE	-	-
INT_ACCEN01	Access Enable Register 1 (Access Mode, Write: P0)	0F0 _H	U, SV	SV, SE	3	Page 17-40
INT_ACCEN00	Access Enable Register 0 (Access Mode, Write: P0)	0F4 _H	U, SV	SV, SE	3	Page 17-39
INT_ACCEN11	Access Enable Register 1 (Access Mode, Write: P1)	0F8 _H	U, SV	SV, SE	3	Page 17-40
INT_ACCEN10	Access Enable Register 0 (Access Mode, Write: P1)	0FC _H	U, SV	SV, SE	3	Page 17-39
INT_LWSR0	ICU0 - Last Winning Service Request Register (CPU0)	100 _H	U, SV	BE	3	Page 17-40
INT_LASR0	ICU0 - Last Acknowledged Service Request Register (CPU0)	104 _H	U, SV	BE	3	Page 17-42
INT_ECR0	ICU0 - Error Capture Register (CPU0)	108 _H	U, SV	SV, P1	3	Page 17-43
-	Reserved	10C _H	U, SV	BE	-	-
INT_LWSR1	ICU1 - Last Winning Service Request Register (CPU1)	110 _H	U, SV	BE	3	Page 17-40

Interrupt Router (IR)
Table 17-2 Registers Overview - System, OTGM and ICU Control Registers

Short Name	Description	Offset Addr.	Access Mode		Reset Class	Description See
			Read	Write		
INT_LASR1	ICU1 - Last Acknowledged Service Request Register (CPU1)	114 _H	U, SV	BE	3	Page 17-42
INT_ECR1	ICU1 - Error Capture Register (CPU1)	118 _H	U, SV	SV, P1	3	Page 17-43
-	Reserved	11C _H - 12F _H	U, SV	BE	-	-
INT_LWSR3	ICU3 - Last Winning Service Request Register (DMA)	130 _H	U, SV	BE	3	Page 17-40
INT_LASR3	ICU3 - Last Acknowledged Service Request Register (DMA)	134 _H	U, SV	BE	3	Page 17-42
INT_ECR3	ICU3 - Error Capture Register (DMA)	138 _H	U, SV	SV, P1	3	Page 17-43
-	Reserved	13C _H - FFF _H	U, SV	BE	-	-

Note: Register bits marked "r" in the following register description are virtual registers and do not contain flip-flops. They are always read as 0.

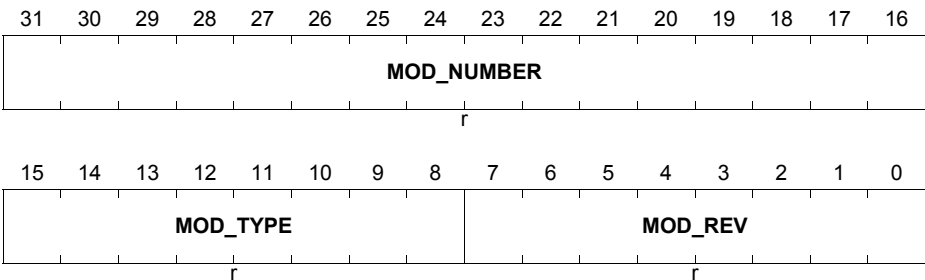
17.11.1 System and ICU Control Registers

Module Identification Register (ID)

Interrupt Router Module Identification Register.

INT_ID

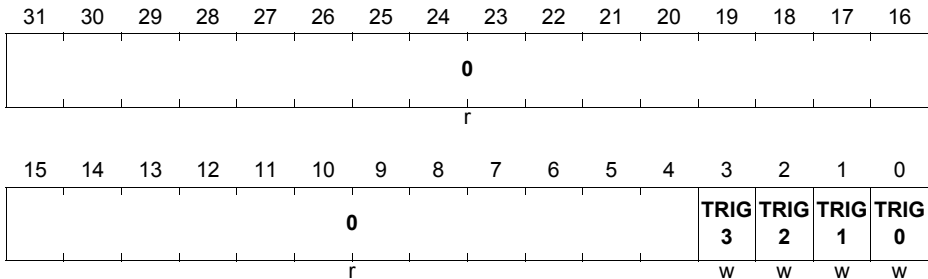
Module Identification Register (008_H) **Reset Value: 00B9 C0XX_H**



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01H (first revision).
MOD_TYPE	[15:8]	r	Module Type The bit field is set to C0H which defines the module as a 32-bit module.
MOD_NUMBER	[31:16]	r	Module Number Value This bit field defines a module identification number. The value for the Interrupt Router module is 0087H.

Service Request Broadcast Register (SRB0 / SRB1)

Interrupt Service Request Broadcast Registers can be used to trigger multiple Service Requests of a General Purpose Service Request Group in parallel.

Interrupt Router (IR)
INT_SRB0
Service Request Broadcast Register 0(010_H)
Reset Value: 0000 0000_H
INT_SRB1
Service Request Broadcast Register 1(014_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
TRIG0	0	w	General Purpose Service Request Trigger 0 Writing with '1' SRBx.TRIG0 triggers the General Purpose Service Request 0 in group x (GPSRx0). Writing with 0 has no effect. This bit is always read as 0.
TRIG1	1	w	General Purpose Service Request Trigger 1 Writing with '1' SRBx.TRIG1 triggers the General Purpose Service Request 0 in group x (GPSRx1). Writing with 0 has no effect. This bit is always read as 0.
TRIG2	2	w	General Purpose Service Request Trigger 2 Writing with '1' SRBx.TRIG2 triggers the General Purpose Service Request 0 in group x (GPSRx2). Writing with 0 has no effect. This bit is always read as 0.
TRIG3	3	w	General Purpose Service Request Trigger 3 Writing with '1' SRBx.TRIG3 triggers the General Purpose Service Request 0 in group x (GPSRx3). Writing with 0 has no effect. This bit is always read as 0.
0	[31:4]	r	Reserved Read as 0; should be written with 0.

Interrupt Router (IR)
Access Enable Register 0 (ACCEN0/10)

The Access Enable Register 0 controls write access for transactions with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The registers ACCEN00 / ACCEN01 are providing one enable bit for each 6-bit On Chip Bus Master TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... ,EN31 -> TAG ID 011111B.

INT_ACCEN00

Access Enable Register 0 (0F4_H) **Reset Value: FFFF FFFF_H**

INT_ACCEN10

Kernel 1 Access Enable Register 0 (0FC_H) **Reset Value: FFFF FFFF_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN3	EN3	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN1	EN1	EN1	EN1	EN1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN1	EN1	EN1	EN1	EN1	EN1	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
5	4	3	2	1	0										
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register 1 (ACCEN11/01)

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). ACCEN11/01 are not implemented with register bits as the related On Chip Bus Master TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000B, EN1 -> TAG ID 100001B, ... ,EN31 -> TAG ID 111111B.

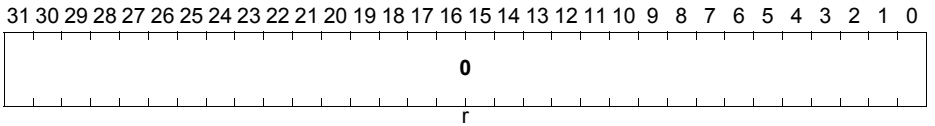
Interrupt Router (IR)

INT_ACCEN01

Kernel 0 Access Enable Register 1 (0F0_H) **Reset Value: 0000 0000_H**

INT_ACCEN11

Kernel 1 Access Enable Register 1 (0F8_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Latest Winning Service Request (LWSR)

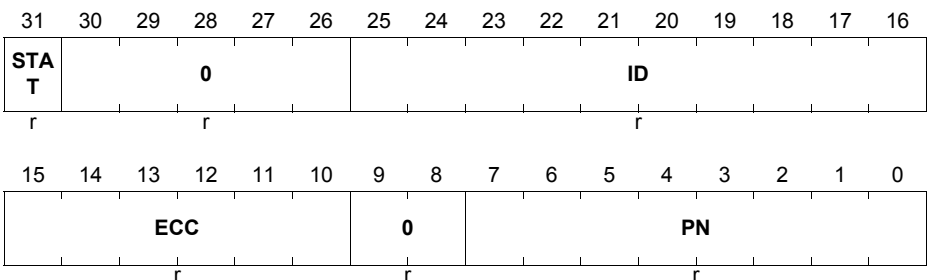
The Latest Winning Service Request register provides informations about the winner of the last arbitration round. The register bit fields are representing what is provided by the ICU to the Interrupt Service Provider.

INT_LWSRx (x = 0-1)

Latest Winning Service Request Register x (100_H+x*10_H) **Reset Value: 0000 0000_H**

INT_LWSR3

Latest Winning Service Request Register 3 (130_H) **Reset Value: 0000 0000_H**



Interrupt Router (IR)

Field	Bits	Type	Description
PN	[7:0]	r	Latest Winner Priority Number This bit field shows the Priority Number of a pending service request that won the last arbitration round. This bit field is only valid if STAT is set to 1
ECC	[15:10]	r	Latest Winner ECC This bit field shows the ECC field (SRN.ECC) that was transferred from the last winning SRN to the ICU. This bit field is only valid if STAT is set to 1.
ID	[25:16]	r	Latest Winner Index Number This bit field shows the index number of the last winning SRN. This bit field is only valid if STAT is set to 1
STAT	31	r	LWSR Register Status The STAT register indicates if the PN, ECC and ID bit fields are still valid. They are still valid if the interrupt from the SRN identified by ID is still pending. If the ICU does not have a winner because no interrupt is pending or not yet arbitrated than it clears the STAT bit. 0 _B LWSR bit fields are not valid 1 _B LWSR bit fields are valid
0	[30:26] , [9:8]	r	Reserved Read as 0; should be written with 0.

Last Acknowledged Service Request (LASR)

The Last Acknowledged Service Request register provides informations about the last service request that was acknowledged by the Interrupt Service Provider. The register bit fields are representing what was sent by the Interrupt Service Provider together with the latest acknowledge.

Interrupt Router (IR)

INT_LASRx (x = 0-1)

Last Acknowledged Service Request Register x ($104_H + x \cdot 10_H$)

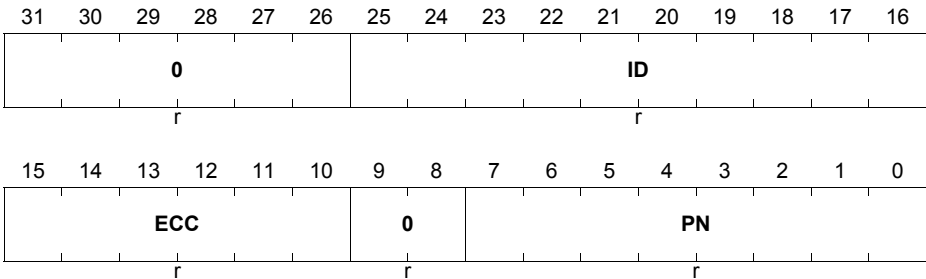
Reset Value:

0000 0000_H

INT_LASR3

Last Acknowledged Service Request Register 3 (134_H)

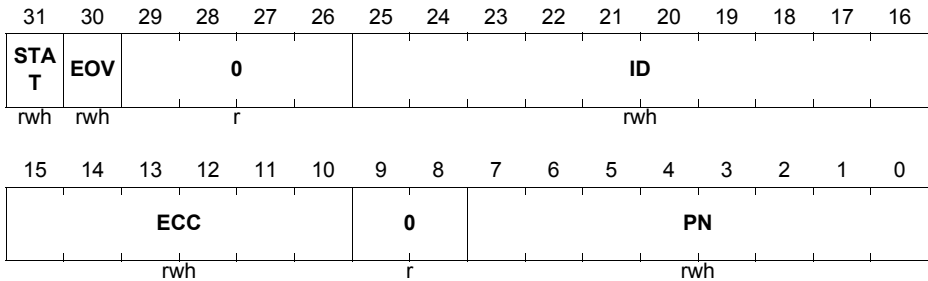
Reset Value: 0000 0000_H



Field	Bits	Type	Description
PN	[7:0]	r	Last Acknowledged Service Request Priority Number This bit field shows the Priority Number of the last acknowledged service request
ECC	[15:10]	r	Last Acknowledged Interrupt ECC This bit field shows the ECC value of the last acknowledged service request, as send by the service provider with acknowledge
ID	[25:16]	r	Last Acknowledged Interrupt SRN Index Number This bit field shows the index number of the SRN, related to the last acknowledged service request, as send by the service provider with acknowledge
0	[31:26], [9:8]	r	Reserved Read as 0; should be written with 0.

Error Capture Register (ECR)

The Error Capture Register captures the Last Acknowledged Service Request (LASR) register contents when an ECC error was detected by the ICU. The ECR shows always the last ECR contents where an ECC error was detected. Software can clear the ECR contents by writing to the ECR.

Interrupt Router (IR)
INT_ECR_x (x = 0-1)
Error Capture Register x
(108_H+x*10_H)
Reset Value: 0000 0000_H
INT_ECR3
Error Capture Register 3
(138_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
PN	[7:0]	rwh	Service Request Priority Number This bit field shows the priority number of the last service request where an error was detected. Bit field can be modified by writing to it.
ECC	[15:10]	rwh	Service Request ECC This bit field shows the ECC of the last service request where an error was detected. Bit field can be modified by writing to it.
ID	[25:16]	rwh	Service Request Node Index Number This bit field shows the index number of the last service request where an error was detected. Bit field can be modified by writing to it
EOV	30	rwh	Error Overflow Bit The bit is set if an ECC error was detected by the ICU while ECR.STAT='1' (Error Overflow situation). 0 _B No Error Overflow situation detected 1 _B Error Overflow situation detected This bit can be cleared by writing with 1. Writing with 0 has no effect. If this bit is cleared, it must be cleared together with the STAT bit.

Interrupt Router (IR)

Field	Bits	Type	Description
STAT	31	rwh	Error Status Bit The Error Status Bit is set whenever an ECC was detected by the ICU. 0 _B No ECC error detected 1 _B ECC error detected This bit can be cleared by writing with 1. Writing with 0 has no effect.
0	[29:26], [9:8]	r	Reserved Read as 0; should be written with 0.

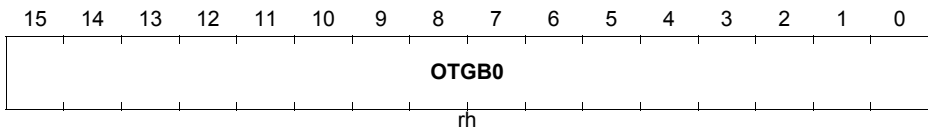
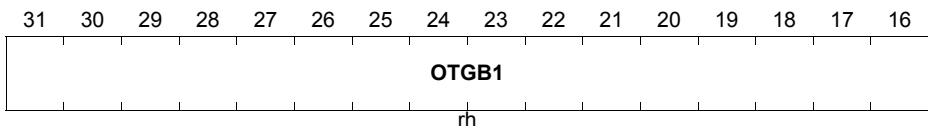
17.12 OTGM Registers

All OTGM registers are cleared by Debug Reset and by each System Reset when OCDS is disabled. They are not touched by System Reset when OCDS is enabled.

Write access is 32 bit wide only and requires Supervisor Mode and OCDS enabled.

17.12.1 Status and Control
INT_OOBS

OTGM OTGB0/1 Status (80_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
OTGB0	[15:0]	rh	Status of OTGB0
OTGB1	[31:16]	rh	Status of OTGB1

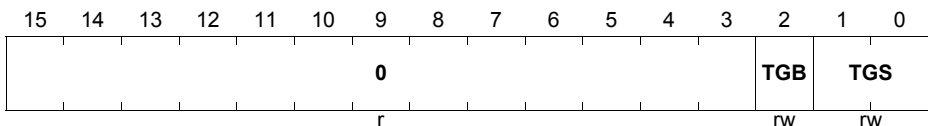
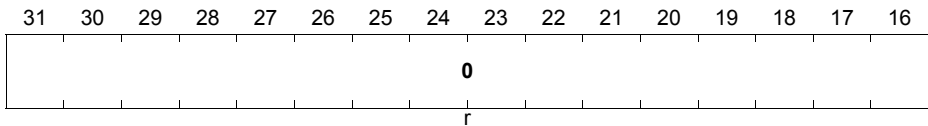
Note: OTGB0/1 value is sampled not captured. A capture register is available in OTGS.

Interrupt Router (IR)

INT_OSSIC

OTGM SSI Control

 (84_H)

 Reset Value: 0000 0000_H


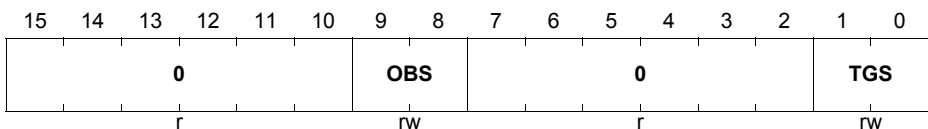
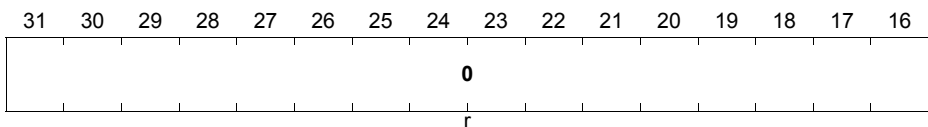
Field	Bits	Type	Description
TGS	[1:0]	rw	Trigger Set for OTGB0/1 0 _H No Trigger Set output 1 _H Trigger Set TS16_SSI (Table 1-1) others, reserved
TGB	2	rw	OTGB0/1 Bus Select 0 _B Trigger Set is output on OTGB0 1 _B Trigger Set is output on OTGB1
0	[31:3]	r	Reserved Read as 0; must be written with 0.

17.12.2 IRQ MUX Control

INT_OIXTS

OTGM IRQ MUX Trigger Set Select

 (88_H)

 Reset Value: 0000 0000_H


Interrupt Router (IR)

Field	Bits	Type	Description
TGS	[1:0]	rw	Trigger Set Select for OTGB0/1 Overlay 0 _D No overlay 1 _D Trigger Set TS8_IS (Table 1-2) 2 _D Trigger Set TS8_SPA (Table 1-3) 3 _D reserved
OBS	[9:8]	rw	Overlay Byte Select 0 _D OTGB0 [7:0] 1 _D OTGB0 [15:8] 2 _D OTGB1 [7:0] 3 _D OTGB1 [15:8]
0	[7:2], [31:10]	r	Reserved Read as 0; must be written with 0.

INT_OIXMS
OTGM IRQ MUX Missed IRQ Select (8C_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						MIRQ									
r						rw									

Field	Bits	Type	Description
MIRQ	[9:0]	rw	SRN Index for Missed Interrupt Trigger
0	[31:10]	r	Reserved Read as 0; must be written with 0.

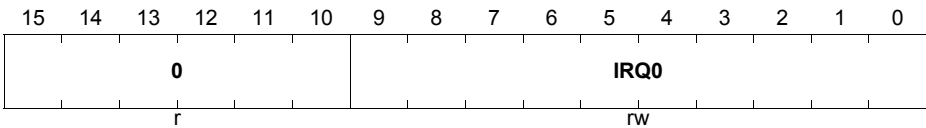
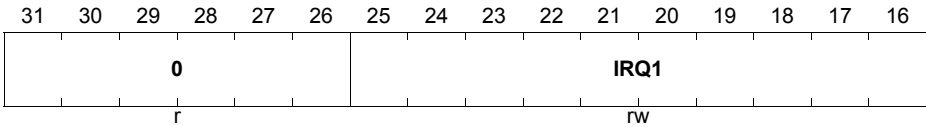
Interrupt Router (IR)

INT_OIXS0

OTGM IRQ MUX Select 0

(90_H)

Reset Value: 0000 0000_H



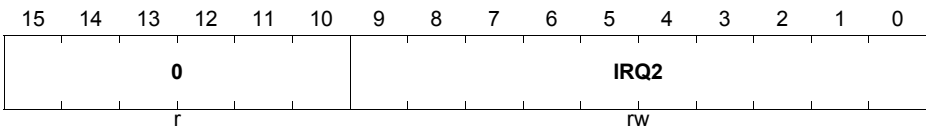
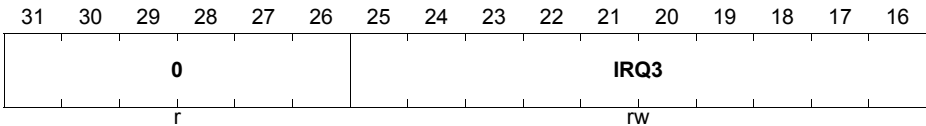
Field	Bits	Type	Description
IRQ0	[9:0]	rw	SRN Index for Interrupt Trigger 0
IRQ1	[25:16]	rw	SRN Index for Interrupt Trigger 1
0	[31:26], [15:10]	r	Reserved Read as 0; must be written with 0.

INT_OIXS1

OTGM IRQ MUX Select 1

(94_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
IRQ2	[9:0]	rw	SRN Index for Interrupt Trigger 2
IRQ3	[25:16]	rw	SRN Index for Interrupt Trigger 3
0	[31:26], [15:10]	r	Reserved Read as 0; must be written with 0.

17.12.3 Interrupt System Trace

INT_OIT

OTGM IRQ Trace

 (A0_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OE1	0			TOS1		OE0	0			TOS0					
rw	r			rw		rw	r			rw					

Field	Bits	Type	Description
TOS0	[1:0]	rw	Type of Service for Observation on OTGB0 Trigger Set TS16_SP (Table 1-4) 0 _D CPU0 service is observed 1 _D CPU1 service is observed 2 _D Reserved (00 _H is output for TS16_SP) 3 _D DMA service is observed
OE0	7	rw	Output Enable for OTGB0 0 _B Disabled 1 _B Enabled
TOS1	[9:8]	rw	Type of Service for Observation on OTGB1 Trigger Set TS16_SP (Table 1-4) 0 _D CPU0 service is observed 1 _D CPU1 service is observed 2 _D Reserved (00 _H is output for TS16_SP) 3 _D DMA service is observed
OE1	15	rw	Output Enable for OTGB1 0 _B Disabled 1 _B Enabled
0	[6:2], [14:10], [31:16]	r	Reserved Read as 0; must be written with 0.

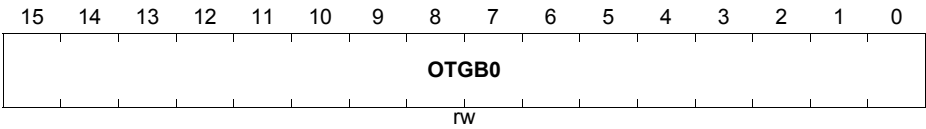
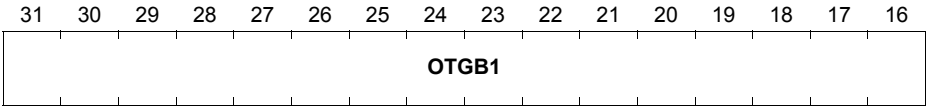
Interrupt Router (IR)

17.12.4 MCDS Interface

INT_OMISP

OTGM MCDS I/F Sensitivity Posedge (A4_H)

Reset Value: 0000 0000_H

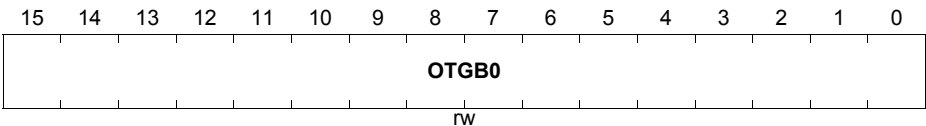
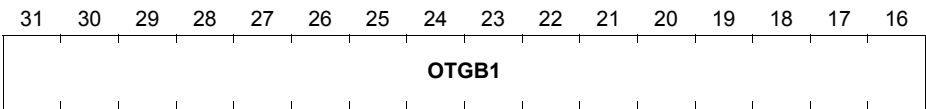


Field	Bits	Type	Description
OTGB0	[15:0]	rw	Bitwise Posedge Sensitivity for OTGB0 If a bit is set an OTGB value will be written to MCDS on a rising edge of the associated OTGB0 bit.
OTGB1	[31:16]	rw	Bitwise Posedge Sensitivity for OTGB1 If a bit is set an OTGB value will be written to MCDS on a rising edge of the associated OTGB1 bit.

INT_OMISN

OTGM MCDS I/F Sensitivity Negedge (A8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
OTGB0	[15:0]	rw	Bitwise Negedge Sensitivity for OTGB0 If a bit is set an OTGB value will be written to MCDS on a falling edge of the associated OTGB0 bit.

Interrupt Router (IR)

Field	Bits	Type	Description
OTGB1	[31:16]	rw	Bitwise Negedge Sensitivity for OTGB1 If a bit is set an OTGB value will be written to MCDS on a falling edge of the associated OTGB1 bit.

17.13 Interrupt Router SRC Registers

Figure 17-9 shows all registers associated with the Interrupt Router module in the TC26x B-Step device. This chapter describes the Service Request Control registers including:

- Mapping of Aurix module interrupt triggers to SRC
- SRC offsets
- SRN Index numbers

List of used Reset Class abbreviations:

- Reset Class 1 -> Debug Reset (see description in the chapter SCU / Reset Types)
- Reset Class 3 -> Application Reset (see description in the chapter SCU / Reset Types)

Note: A violation of the access protection will not be executed (e.g. a write to a 'Px'ACCEN protected register from a disabled master). In this case an access protection error is signaled to the SMU. Beside this signaling to the SMU, no other error, interrupt or trap is generated.

Table 17-3 Registers Address Space - Service Request Control Registers (SRC)

Module	Base Address	End Address	Note
SRC	F003 8000 _H	F003 9FFF _H	

Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Reset Class	Index NR.
				Read	Write		
SRC_CPU0 SBSRC	CPU0	CPU 0 Software Breakpoint Service Request	0000 _H	U, SV	SV, P0, P1	1	0
SRC_CPU1 SBSRC	CPU1	CPU 1 Software Breakpoint Service Request	0004 _H	U, SV	SV, P0, P1	1	1
-	-	Reserved	0008 _H - 001C _H	U, SV	BE	-	-

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Reset Classes	Index NR.
				Read	Write		
SRC_EMEM	EMEM	Emulation Memory Service Request (on ED devices only)	0020 _H	U, SV	SV, P0, P1	1	2
SRC_AGBT	AGBT	AGBT Service Request (on ED devices only)	0024 _H	U, SV	SV, P0, P1	1	3
-	-	Reserved	0028 _H -0034 _H	U, SV	BE	-	-
SRC_BCUS PBSBSRC	BCU	Bus Control Unit SPB Service Request	0040 _H	U, SV	SV, P0, P1	1	4
-	-	Reserved	000C _H -0044 _H	U, SV	BE	-	-
SRC_XBAR SRC	XBAR_SRI	XBAR_SRI Service Request	0048 _H	U, SV	SV, P0, P1	1	5
-	-	Reserved	004C _H	U, SV	BE	-	-
SRC_Cerberusm	CERBERUS	Cerberus Service Request m (m = 0-1)	0050 _H + (m × 4 _H)	U, SV	SV, P0, P1	1	6 + m
-	-	Reserved	0058 _H -007C _H	U, SV	BE	-	-
SRC_ASCLINmTX	ASCLINm	ASCLIN m Transmit Service Request (m = 0-3)	0080 _H + (m × C _H)	U, SV	SV, P0, P1	3	8 + m*3
SRC_ASCLINmRX	ASCLINm	ASCLIN m Receive Service Request (m = 0-3)	0084 _H + (m × C _H)	U, SV	SV, P0, P1	3	9 + m*3
SRC_ASCLINmEX	ASCLINm	ASCLIN m Error Service Request (m = 0-3)	0088 _H + (m × C _H)	U, SV	SV, P0, P1	3	10 + m*3

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Res et Class	Index NR.
				Read	Write		
-	-	Reserved	00B0 _H -018C _H	BE	BE	-	-
SRC_QSPI mTX	QSPI m	QSPI m Transmit Service Request (m = 0-3)	0190 _H + (m × 18 _H)	U, SV	SV, P0, P1	3	20 + m*6
SRC_QSPI mRX	QSPI m	QSPI m Receive Service Request (m = 0-3)	0194 _H + (m × 18 _H)	U, SV	SV, P0, P1	3	21 + m*6
SRC_QSPI mERR	QSPI m	QSPI m Error Service Request (m = 0-3)	0198 _H + (m × 18 _H)	U, SV	SV, P0, P1	3	22 + m*6
SRC_QSPI mPT	QSPI m	QSPI m Phase Transition Service Request (m = 0-3)	019C _H + (m × 18 _H)	U, SV	SV, P0, P1	3	23 + m*6
SRC_RESE RVED1m	RESE RVED	Reserved Service Request 1m (m = 0-1)	01A0 _H + (m × 18 _H)	U, SV	SV, P0, P1	3	24 + m*6
SRC_QSPI mHC	QSPI m	QSPI m High Speed Capture Service Request (m = 2-3)	01A0 _H + (m × 18 _H)	U, SV	SV, P0, P1	3	24 + m*6
SRC_QSPI mU	QSPI m	QSPI m User Defined Service Request (m = 0-3)	01A4 _H + (m × 18 _H)	U, SV	SV, P0, P1	3	25 + m*6
-	-	Reserved	0208 _H -028C _H	U, SV	BE	-	-
SRC_HSCT	HSCT	HSCT Service Request	0290 _H	U, SV	SV, P0, P1	3	44
-	-	Reserved	0294 _H -029C _H	U, SV	BE	-	-
SRC_HSSL COKm	HSSL	Channel m OK Service Request (m = 0-3)	02A0 _H + (m × 10 _H)	U, SV	SV, P0, P1	3	45 + m*4

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Res et Clas s	Inde x NR.
				Rea d	Write		
SRC_HSSL RDIm	HSSL	Channel m Read Data Service Request (m = 0-3)	02A4 _H + (m × 10 _H)	U, SV	SV, P0, P1	3	46 + m*4
SRC_HSSL ERRm	HSSL	Channel m Error Service Request (m = 0-3)	02A8 _H + (m × 10 _H)	U, SV	SV, P0, P1	3	47 + m*4
SRC_HSSL TRGm	HSSL	Channel m Trigger Interrupt Service Request m (m = 0-3)	02AC _H + (m × 10 _H)	U, SV	SV, P0, P1	3	48 + m*4
SRC_HSSL EXI	HSSL	HSSL Exception Service Request	02E0 _H	U, SV	SV, P0, P1	3	49
-	-	Reserved	02E4 _H - 02FC _H	U, SV	BE	-	-
SRC_I2C0B REQ	I2C	I2C 0 Burst Data Transfer Request	0300 _H	U, SV	SV, P0, P1	3	62
SRC_I2C0L BREQ	I2C	I2C 0 Last Burst Data Transfer Service Request	0304 _H	U, SV	SV, P0, P1	3	63
SRC_I2C0S REQ	I2C	I2C 0 Single Data Transfer Service Request	0308 _H	U, SV	SV, P0, P1	3	64
SRC_I2C0L SREQ	I2C	I2C 0 Last Single Data Transfer Service Request	030C _H	U, SV	SV, P0, P1	3	65
SRC_I2C0E RR	I2C	I2C 0 Error Service Request	0310 _H	U, SV	SV, P0, P1	3	66
SRC_I2C0P	I2C	I2C 0 Kernel Service Request	0314 _H	U, SV	SV, P0, P1	3	67

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Res et Class	Index NR.
				Read	Write		
-	-	Reserved	0318 _H -034C _H	BE	BE	-	-
SRC_SENT _m	SENT	SENT TRIG _m Service Request (m = 0-5)	0350 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	68 + m
-	-	Reserved	0378 _H -03DC _H	BE	BE	-	-
SRC_MSC _m SR0	MSC _m	MSC m Service Request 0 (m = 0-1)	03E0 _H + (m × 10 _H)	U, SV	SV, P0, P1	3	74 + m*5
SRC_MSC _m SR1	MSC _m	MSC m Service Request 1 (m = 0-1)	03E4 _H + (m × 10 _H)	U, SV	SV, P0, P1	3	75 + m*5
SRC_MSC _m SR2	MSC _m	MSC m Service Request 2 (m = 0-1)	03E8 _H + (m × 10 _H)	U, SV	SV, P0, P1	3	76 + m*5
SRC_MSC _m SR3	MSC _m	MSC m Service Request 3 (m = 0-1)	03EC _H + (m × 10 _H)	U, SV	SV, P0, P1	3	77 + m*5
SRC_MSC _m SR4	MSC _m	MSC m Service Request 4 (m = 0-1)	03F0 _H + (m × 10 _H)	U, SV	SV, P0, P1	3	78 + m*5
-	-	Reserved	0408 _H -041C _H	BE	BE	-	-
SRC_CCU6 _m SR0	CCU6 _m	CCU6 m Service Request 0 (m = 0-1)	0420 _H + (m × 10 _H)	U, SV	SV, P0, P1	3	84 + m*4
SRC_CCU6 _m SR1	CCU6 _m	CCU6 m Service Request 1 (m = 0-1)	0424 _H + (m × 10 _H)	U, SV	SV, P0, P1	3	85 + m*4
SRC_CCU6 _m SR2	CCU6 _m	CCU6 m Service Request 2 (m = 0-1)	0428 _H + (m × 10 _H)	U, SV	SV, P0, P1	3	86 + m*4

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Res et Class	Index NR.
				Read	Write		
SRC_CCU6 mSR3	CCU6 m	CCU6 m Service Request 3 (m = 0-1)	042C _H + (m × 10 _H)	U, SV	SV, P0, P1	3	87 + m*4
SRC_GPT1 20CIRQ	GPT1 20	GPT120 CAPREL Service Request	0460 _H	U, SV	SV, P0, P1	3	92
SRC_GPT1 20T2	GPT1 20	GPT120 T2 Overflow/Underflow Service Request	0464 _H	U, SV	SV, P0, P1	3	93
SRC_GPT1 20T3	GPT1 20	GPT120 T3 Overflow/Underflow Service Request	0468 _H	U, SV	SV, P0, P1	3	94
SRC_GPT1 20T4	GPT1 20	GPT120 T4 Overflow/Underflow Service Request	046C _H	U, SV	SV, P0, P1	3	95
SRC_GPT1 20T5	GPT1 20	GPT120 T5 Overflow/Underflow Service Request	0470 _H	U, SV	SV, P0, P1	3	96
SRC_GPT1 20T6	GPT1 20	GPT120 T6 Overflow/Underflow Service Request	0474 _H	U, SV	SV, P0, P1	3	97
SRC_STMm SR0	STM m	System Timer m Service Request 0 (m = 0-1)	0490 _H + (m × 8 _H)	U, SV	SV, P0, P1	3	98 + m*2
SRC_STMm SR1	STM m	System Timer m Service Request 1 (m = 0-1)	0494 _H + (m × 8 _H)	U, SV	SV, P0, P1	3	99 + m*2
-	-	Reserved	04A0 _H -04AC _H	BE	BE	-	-
SRC_FCE	FCE	FCE Error Service Request	04B0 _H	U, SV	SV, P0, P1	3	102

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Res et Class	Inde x NR.
				Rea d	Write		
-	-	Reserved	04B4 _H - 04EC _H	BE	BE	-	-
SRC_DMAE RR	DMA	DMA Error Service Request	04F0	U, SV	SV, P0, P1	3	103
-	-	Reserved	04F4 _H - 04FC _H	BE	BE	3	-
SRC_DMACH m	DMA	DMA Channel m Service Request (m = 0-47)	0500 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	104 + m
-	-	Reserved	0580 _H - 08EC _H	BE	BE	-	-
SRC_ETH	ETH	Ethernet Service Request	08F0 _H	U, SV	SV, P0, P1	3	152
-	-	Reserved	08F4 _H - 08FC _H	BE	BE	-	-
SRC_CANI NTm	MultiC AN+	MultiCAN+ Service Request m (m = 0-15)	0900 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	153 + m
-	-	Reserved	0940 _H - 097C _H	BE	BE	-	-
SRC_VADC G0SRm	VADC	VADC Group 0 Service Request m (m = 0-3)	0980 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	169 + m
SRC_VADC G1SRm	VADC	VADC Group 1 Service Request m (m = 0-3)	0990 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	173 + m
SRC_VADC G2SRm	VADC	VADC Group 2 Service Request m (m = 0-3)	09A0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	177 + m

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Res et Class	Inde x NR.
				Rea d	Write		
SRC_VADC G3SRm	VADC	VADC Group 3 Service Request m (m = 0-3)	09B0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	181 + m
-	-	Reserved	09C0 _H - 0A9C _H	BE	BE	-	-
SRC_VADC CG0SRm	VADC	VADC Common Group 0 Service Request m (m = 0-3)	0AA0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	185 + m
SRC_VADC CG1SRm	VADC	VADC Common Group 1 Service Request m (m = 0-3)	0AB0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	189 + m
-	-	Reserved	0AC0 _H - 0B4C _H	BE	BE	-	-
SRC_DSAD CSRMO	DSAD C	DSADC SRM0 Service Request	0B50 _H	U, SV	SV, P0, P1	3	193
SRC_DSAD CSRA0	DSAD C	DSADC SRA0 Service Request	0B54 _H	U, SV	SV, P0, P1	3	194
-	-	Reserved	0B58 _H - 0B5C _H	BE	BE	-	-
SRC_DSAD CSRMO2	DSAD C	DSADC SRM2 Service Request	0B60 _H	U, SV	SV, P0, P1	3	195
SRC_DSAD CSRA2	DSAD C	DSADC SRA2 Service Request	0B64 _H	U, SV	SV, P0, P1	3	196
SRC_DSAD CSRMO3	DSAD C	DSADC SRM3 Service Request	0B68 _H	U, SV	SV, P0, P1	3	197
SRC_DSAD CSRA3	DSAD C	DSADC SRA3 Service Request	0B6C _H	U, SV	SV, P0, P1	3	198

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Reset Class	Index NR.
				Read	Write		
-	-	Reserved	0B70 _H -0BDC _H	BE	BE	-	-
SRC_ERAY INT0	E- RAY	E-RAY Service Request 0	0BE0 _H	U, SV	SV, P0, P1	3	199
SRC_ERAY INT1	E- RAY	E-RAY Service Request 1	0BE4 _H	U, SV	SV, P0, P1	3	200
SRC_ERAY TINT0	E- RAY	E-RAY Timer Interrupt 0 Service Request	0BE8 _H	U, SV	SV, P0, P1	3	201
SRC_ERAY TINT1	E- RAY	E-RAY Timer Interrupt 1 Service Request	0BEC _H	U, SV	SV, P0, P1	3	202
SRC_ERAY NDAT0	E- RAY	E-RAY New Data 0 Service Request	0BF0 _H	U, SV	SV, P0, P1	3	203
SRC_ERAY NDAT1	E- RAY	E-RAY New Data 1 Service Request	0BF4 _H	U, SV	SV, P0, P1	3	204
SRC_ERAY MBSC0	E- RAY	E-RAY Message Buffer Status Changed 0 Service Request	0BF8 _H	U, SV	SV, P0, P1	3	205
SRC_ERAY MBSC1	E- RAY	E-RAY Message Buffer Status Changed 1 Service Request	0BFC _H	U, SV	SV, P0, P1	3	206
SRC_ERAY OBUSY	E- RAY	E-RAY Output Buffer Busy Service Request	0C00 _H	U, SV	SV, P0, P1	3	207
SRC_ERAY IBUSY	E- RAY	E-RAY Input Buffer Busy Service Request	0C04 _H	U, SV	SV, P0, P1	3	208

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Res et Class	Inde x NR.
				Rea d	Write		
-	-	Reserved	0C08 _H -0C2C _H	BE	BE	-	-
SRC_PMU00	PMU0	PMU 0 Service Request 0	0C30 _H	U, SV	SV, P0, P1	3	209
SRC_PMU01	PMU0	PMU 0 Service Request 1	0C34 _H	U, SV	SV, P0, P1	3	210
-	-	Reserved	0C38 _H -0CCC _H	BE	BE	-	-
SRC_SCUDTS	SCU	SCU DTS Busy Service Request	0CD0 _H	U, SV	SV, P0, P1	3	211
SRC_SCUERUm	SCU	SCU ERU Service Request x (m = 0-3)	0CD4 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	212 + m
-	-	Reserved	0CE4 _H -0CFC _H	BE	BE	-	-
SRC_SCR	SCR	Stand By Controller Service Request	0D00 _H	U, SV	SV, P0, P1	3	216
-	-	Reserved	0D04 _H -0D0C _H	BE	BE	-	-
SRC_SMUm	SMU	SMU Service Request m (m = 0-2)	0D10 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	217 + m
-	-	Reserved	0D1C _H -0D2C _H	BE	BE	-	-
SRC_PSI5m	PSI5	PSI5 Service Request m (m = 0-7)	0D30 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	220 + m
-	-	Reserved	0D5C _H -0D9C _H	BE	BE	-	-

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Res et Class	Inde x NR.
				Rea d	Write		
SRC_CIFMI	CIF	CIF MI Service Request	0DA0 _H	U, SV	SV, P0, P1	3	228
SRC_CIFMI EP	CIF	CIF MI EP Service Request	0DA4 _H	U, SV	SV, P0, P1	3	229
SRC_CIFIS P	CIF	CIF ISP Service Request	0DA8 _H	U, SV	SV, P0, P1	3	230
SRC_CIFMJPEG	CIF	CIF MJPEG Service Request	0DAC _H	U, SV	SV, P0, P1	3	231
-	-	Reserved	0DB0 _H -0DDC _H	BE	BE	-	-
SRC_LMU	LMU	LMU Error Service Request	0DE0 _H	U, SV	SV, P0, P1	3	232
-	-	Reserved	0DE4 _H -0DEC _H	BE	BE	-	-
SRC_PSI5S m	PSI5-S	PSI5-S Service Request m (m = 0-7)	0DF0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	233 + m
-	-	Reserved	0E10 _H -0FBC _H	BE	BE	-	-
SRC_FFTD ONE	-	FFT Done Service Request	0FC0 _H	U, SV	SV, P0, P1	3	441
SRC_FFTE RR	-	FFT Error Service Request	0FC4 _H	U, SV	SV, P0, P1	3	422
SRC_FFTR FS	-	FFT Ready For Start Service Request	0FC8 _H	U, SV	SV, P0, P1	3	423

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Reset Class	Index NR.
				Read	Write		
-	-	Reserved	0FCC _H - 0FFC _H	BE	BE	-	-
SRC_GPSR 0m	IR	General Purpose Service Request 0 m (m = 0-3)	1000 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	244 + m
-	-	Reserved	1010 _H - 101C _H	BE	BE	-	-
SRC_GPSR 1m	IR	General Purpose Service Request 1 m (m = 0-3)	1020 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	248 + m
-	-	Reserved	1030 _H - 15FC _H	BE	BE	-	-
SRC_GTMA EIIRQ	GTM	AEI Shared Service Request	1600 _H	U, SV	SV, P0, P1	3	252
SRC_GTMA RUIRQm	GTM	ARU Shared Service Request m (m = 0-2)	1604 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	253 + m
-	-	Reserved	1610 _H	BE	BE	-	-
SRC_GTMB RCIRQ	GTM	BRC Shared Service Request	1614 _H	U, SV	SV, P0, P1	3	256
SRC_GTMC MPIRQ	GTM	CMP Shared Service Request	1618 _H	U, SV	SV, P0, P1	3	257
SRC_GTMS PE0IRQ	GTM	SPE0 Shared Service Request	161C _H	U, SV	SV, P0, P1	3	258
SRC_GTMS PE1IRQ	GTM	SPE1 Shared Service Request	1620 _H	U, SV	SV, P0, P1	3	259

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Res et Class	Index NR.
				Read	Write		
SRC_GTMP SM0m	GTM	PSM0 Shared Service Request m (m = 0-7)	162C _H + (m × 4 _H)	U, SV	SV, P0, P1	3	260 + m
-	-	Reserved	1644 _H - 16A0 _H	BE	BE	-	-
SRC_GTMD PLLm	GTM	DPLL Service Request m (m = 0-26)	16A4 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	268 + m
-	-	Reserved	1710 _H - 176C _H	BE	BE	-	-
SRC_GTME RR	GTM	Error Service Request	1770 _H	U, SV	SV, P0, P1	3	295
-	-	Reserved	1774 _H - 177C _H	BE	BE	-	-
SRC_GTMT IM0m	GTM	TIM0 Shared Service Request m (m = 0-7)	1780 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	296 + m
SRC_GTMT IM1m	GTM	TIM1 Shared Service Request m (m = 0-7)	17A0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	304 + m
SRC_GTMT IM2m	GTM	TIM2 Shared Service Request m (m = 0-7)	17C0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	312 + m
-	-	Reserved	17E0 _H - 197C _H	BE	BE	-	-
SRC_GTM MCS0m	GTM	MCS0 Shared Service Request m (m = 0-7)	1980 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	320 + m
SRC_GTM MCS1m	GTM	MCS1 Shared Service Request m (m = 0-7)	19A0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	328 + m

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Res et Class	Inde x NR.
				Rea d	Write		
SRC_GTM MCS2m	GTM	MCS2 Shared Service Request m (m = 0-7)	19C0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	336 + m
-	-	Reserved	19E0 _H - 1B7C _H	BE	BE	-	-
SRC_GTMT OM0m	GTM	TOM0 Shared Service Request m (m = 0-7)	1B80 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	344 + m
SRC_GTMT OM1m	GTM	TOM1 Shared Service Request m (m = 0-7)	1BA0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	352 + m
-	-	Reserved	1BC0 _H - 1D7C _H	BE	BE	-	-
SRC_GTMA TOM0m	GTM	ATOM0 Shared Service Request m (m = 0-3)	1D80 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	360 + m
SRC_GTMA TOM1m	GTM	ATOM1 Shared Service Request m (m = 0-3)	1D90 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	364 + m
SRC_GTMA TOM2m	GTM	ATOM2 Shared Service Request m (m = 0-3)	1DA0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	368 + m
SRC_GTMA TOM3m	GTM	ATOM3 Shared Service Request m (m = 0-3)	1DB0 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	372 + m
-	-	Reserved	1DC0 _H - 1EFC _H	BE	BE	-	-
SRC_GTM MCSW0m	GTM	GTM Multi Channel Sequencer 0 Service Request m (m = 0-2)	1F00 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	376 + m
-	-	Reserved	1F0C _H - 1F3C _H	BE	BE	-	-

Interrupt Router (IR)
Table 17-4 Registers Overview - Service Request Control Registers

Short Name	Module	Description	Offset Addr.	Access Mode		Reset Class	Index NR.
				Read	Write		
SRC_GTM MCSW1m	GTM	GTM Multi Channel Sequencer 1 Service Request m (m = 0-2)	1F10 _H + (m × 4 _H)	U, SV	SV, P0, P1	3	379 + m
-	-	Reserved	1F18 _H - 1FFC _H	BE	BE	-	-

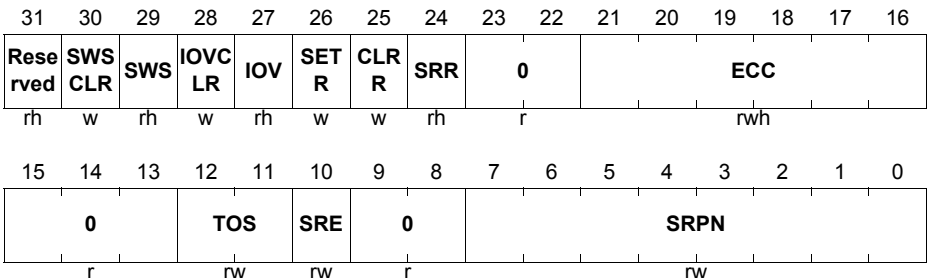
Note: Register bits marked "r" in the following register description are virtual registers and do not contain flip-flops. They are always read as 0.

Note: SRC_xxx[31:16] is P0 write protected (P0 control via IR.ACCEN01/00).

Note: SRC_xxx[15:0] is P1 write protected (P1 control via IR.ACCEN11/10).

Interrupt Router (IR)

SRC_CPUxSBSRC (x=0-1) CPU x Software Breakpoint Service Request (0000 _H +x*4 _H)	Reset Value: 0000 0000 _H
SRC_EMEM Emulation Memory Service Request (0020 _H)	Reset Value: 0000 0000 _H
SRC_AGBT AGBT Service Request (0024 _H)	Reset Value: 0000 0000 _H
SRC_BCUSPBSB SRC Bus Control Unit SPB Service Request (0040 _H)	Reset Value: 0000 0000 _H
SRC_XBARSRC XBAR_SRI Service Request (0048 _H)	Reset Value: 0000 0000 _H
SRC_CERBERUSm (m=0-1) Cerberus Service Request m (0050 _H +m*04 _H)	Reset Value: 0000 0000 _H
SRC_ASCLINmTX (m=0-3) ASCLIN m Transmit Service Request (0080 _H +m*0C _H)	Reset Value: 0000 0000 _H
SRC_ASCLINmRX (m=0-3) ASCLIN m Receive Service Request (0084 _H +m*0C _H)	Reset Value: 0000 0000 _H
SRC_ASCLINmERR (m=0-3) ASCLIN m Error Service Request (0088 _H +m*0C _H)	Reset Value: 0000 0000 _H
SRC_QSPImTX (m=0-3) QSPI m Transmit Service Request (0190 _H +m*18 _H)	Reset Value: 0000 0000 _H
SRC_QSPImRX (m=0-3) QSPI m Receive Service Request (0194 _H +m*18 _H)	Reset Value: 0000 0000 _H



Interrupt Router (IR)

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority ... FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.

Interrupt Router (IR)

Field	Bits	Type	Description
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

Interrupt Router (IR)

SRC_QSPImERR (m=0-3) QSPI m Error Service Request	(0198 _H +m*18 _H)	Reset Value: 0000 0000 _H
SRC_QSPImPT (m=0-3) QSPI m Phase Transition Service Request	(019C _H +m*18 _H)	Reset Value: 0000 0000 _H
SRC_RESERVED1m (m=0-1) Reserved Service Request 1m	(01A0 _H +m*18 _H)	Reset Value: 0000 0000 _H
SRC_QSPImHC (m=2-3) QSPI m High Speed Capture Service Request	(01A0 _H +m*18 _H)	Reset Value: 0000 0000 _H
SRC_QSPImU (m=0-3) QSPI m User Defined Service Request	(01A4 _H +m*18 _H)	Reset Value: 0000 0000 _H
SRC_HSCT HSCT Service Request	(0290 _H)	Reset Value: 0000 0000 _H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rese rved	SWS CLR	SWS	IOVC LR	IOV	SET R	CLR R	SRR	0					ECC		
rh	w	rh	w	rh	w	w	rh	r					rwh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0		TOS	SRE	0							SRPN			
	r		rw	rw	r							rw			

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority ... FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled

Interrupt Router (IR)

Field	Bits	Type	Description
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

Interrupt Router (IR)
SRC_HSSLCOKm (m=0-3)
Channel m OK Service Request m(02A0_H+m*10_H) **Reset Value: 0000 0000_H**
SRC_HSSLRDIm (m=0-3)
Channel m Read Data Service Request m(02A4_H+m*10_H) **Reset Value: 0000 0000_H**
SRC_HSSLERRm (m=0-3)
Channel m Error ServiceRequest m(02A8_H+m*10_H) **Reset Value: 0000 0000_H**
SRC_HSSLTRGm (m=0-3)
Channel m Trigger Service Request m(02AC_H+m*10_H) **Reset Value: 0000 0000_H**
SRC_HSSLEXI
Exception Service Request (02E0_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOV CLR	IOV	SET R	CLR R	SRR	0	ECC						
rh	w	rh	w	rh	w	w	rh	r	rwh						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TOS			SRE	0			SRPN						
r		rw			rw	r			rw						

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC

Interrupt Router (IR)

Field	Bits	Type	Description
SRR	24	rh	Service Request Flag 0_B No service request is pending 1_B A service request is pending
CLRR	25	w	Request Clear Bit 0_B No action 1_B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0_B No action 1_B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0_B No Interrupt Trigger Overflow detected 1_B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0_B No action 1_B Clear IOV; bit value is not stored; read always returns 0.
SWS	29	rh	SW Sticky Bit 0_B No interrupt was initiated via SETR 1_B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0_B No action 1_B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

Interrupt Router (IR)

SRC_I2C0BREQ I2C 0 Burst Data Transfer Request	(0300 _H)	Reset Value: 0000 0000 _H
SRC_I2C0LBREQ I2C 0 Last Burst Data Transfer Service Request	(0304 _H)	Reset Value: 0000 0000 _H
SRC_I2C0SREQ I2C 0 Single Data Transfer Service Request	(0308 _H)	Reset Value: 0000 0000 _H
SRC_I2C0LSREQ I2C 0 Last Single Data Transfer Service Request	(030C _H)	Reset Value: 0000 0000 _H
SRC_I2C0ERR I2C 0 Error Service Request	(0310 _H)	Reset Value: 0000 0000 _H
SRC_I2C0P I2C 0 Kernel Service Request	(0314 _H)	Reset Value: 0000 0000 _H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rese rved	SWS CLR	SWS	IOVC LR	IOV	SET R	CLR R	SRR	0					ECC		
rh	w	rh	w	rh	w	w	rh	r					rwh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0		TOS	SRE	0							SRPN			
	r		rw	rw	r							rw			

Field	Bits	Type	Description
SRPN	[7:0]	rw	<p>Service Request Priority Number</p> <p>00_H Service request is on lowest priority</p> <p>01_H Service request is one before lowest priority</p> <p>... ..</p> <p>FF_H Service request is on highest priority</p> <p><i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i></p>
SRE	10	rw	<p>Service Request Enable</p> <p>0_B Service request is disabled</p> <p>1_B Service request is enabled</p>

Interrupt Router (IR)

Field	Bits	Type	Description
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

Interrupt Router (IR)

SRC_SENTm (m=0-5)		
SENT TRIGm Service Request	(0350_H+m*04_H)	Reset Value: 0000 0000_H
SRC_MSCmSR0 (m=0-1)		
MSC m Service Request 0	(03E0_H+m*14_H)	Reset Value: 0000 0000_H
SRC_MSCmSR1 (m=0-1)		
MSC m Service Request 1	(03E4_H+m*14_H)	Reset Value: 0000 0000_H
SRC_MSCmSR2 (m=0-1)		
MSC m Service Request 2	(03E8_H+m*14_H)	Reset Value: 0000 0000_H
SRC_MSCmSR3 (m=0-1)		
MSC m Service Request 3	(03EC_H+m*14_H)	Reset Value: 0000 0000_H
SRC_MSCmSR4 (m=0-1)		
MSC m Service Request 4	(03F0_H+m*14_H)	Reset Value: 0000 0000_H
SRC_CCU6mSR0 (m=0-1)		
CCU6 m Service Request 0	(0420_H+m*10_H)	Reset Value: 0000 0000_H
SRC_CCU6mSR1 (m=0-1)		
CCU6 m Service Request 1	(0424_H+m*10_H)	Reset Value: 0000 0000_H
SRC_CCU6mSR2 (m=0-1)		
CCU6 m Service Request 2	(0428_H+m*10_H)	Reset Value: 0000 0000_H
SRC_CCU6mSR3 (m=0-1)		
CCU6 m Service Request 3	(042C_H+m*10_H)	Reset Value: 0000 0000_H
SRC_GPT120CIRQ		
GPT120 CAPREL Service Request	(0460_H)	Reset Value: 0000 0000_H
SRC_GPT120T2		
GPT120 T2 Overflow/Underflow Service Request	(0464_H)	Reset Value: 0000 0000_H
SRC_GPT120T3		
GPT120 T3 Overflow/Underflow Service Request	(0468_H)	Reset Value: 0000 0000_H
SRC_GPT120T4		
GPT120 T4 Overflow/Underflow Service Request	(046C_H)	Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOV CLR	IOV	SET R	CLR R	SRR	0	ECC						
rh	w	rh	w	rh	w	w	rh	r	rwh						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TOS		SRE		0		SRPN							
r		rw		rw		r		rw							

Interrupt Router (IR)

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.

Interrupt Router (IR)

Field	Bits	Type	Description
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

SRC_GPT120T5
GPT120 T5 Overflow/Underflow Service Request
(0470_H)
Reset Value: 0000 0000_H
SRC_GPT120T6
GPT120 T6 Overflow/Underflow Service Request
(0474_H)
Reset Value: 0000 0000_H
SRC_STMxSR0 (x=0-1)
System Timer x Service Request 0
(0490_H+x*08_H)
Reset Value: 0000 0000_H
SRC_STMxSR1 (x=0-1)
System Timer x Service Request 1
(0494_H+x*08_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOV CLR	IOV	SET R	CLR R	SRR	0					ECC		
rh	w	rh	w	rh	w	w	rh	r					rwh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0		TOS		SRE		0						SRPN		
	r		rw		rw		r						rw		

Interrupt Router (IR)

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.

Interrupt Router (IR)

Field	Bits	Type	Description
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

SRC_FCE
FCE Error Service Request (04B0_H) Reset Value: 0000 0000_H
SRC_DMAERR
DMA Error Service Request (04F0_H) Reset Value: 0000 0000_H
SRC_DMACHm (m=0-47)
DMA Channel m Service Request (0500_H+m*4_H) Reset Value: 0000 0000_H
SRC_ETH
Ethernet Service Request (08F0_H) Reset Value: 0000 0000_H
SRC_CANINTm (m=0-15)
MULTICAN+ Service Request m (0900_H+m*4_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOV CLR	IOV	SET R	CLR R	SRR	0	ECC						
rh	w	rh	w	rh	w	w	rh	r	rwh						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TOS		SRE		0		SRPN							
r		rw		rw		r		rw							

Interrupt Router (IR)

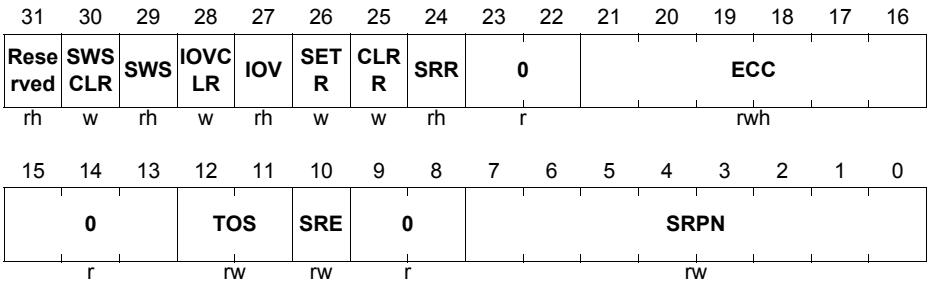
Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.

Interrupt Router (IR)

Field	Bits	Type	Description
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

Interrupt Router (IR)

- SRC_VADCG0SRm (m=0-3)**
VADC Group 0 Service Request m
 (0980_H+m*4_H) **Reset Value: 0000 0000_H**
- SRC_VADCG1SRm (m=0-3)**
VADC Group 1 Service Request m
 (0990_H+m*4_H) **Reset Value: 0000 0000_H**
- SRC_VADCG2SRm (m=0-3)**
VADC Group 2 Service Request m
 (09A0_H+m*4_H) **Reset Value: 0000 0000_H**
- SRC_VADCG3SRm (m=0-3)**
VADC Group 3 Service Request m
 (09B0_H+m*4_H) **Reset Value: 0000 0000_H**
- SRC_VADCCG0SRx (x=0-3)**
VADC Common Group 0 Service Request x
 (0AA0_H+x*4_H) **Reset Value: 0000 0000_H**
- SRC_VADCCG1SRx (x=0-3)**
VADC Common Group 1 Service Request x
 (0AB0_H+x*4_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>

Interrupt Router (IR)

Field	Bits	Type	Description
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.

Interrupt Router (IR)

Field	Bits	Type	Description
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

SRC_DSADCSRMO
DSADC SRM0 Service Request (0B50_H) Reset Value: 0000 0000_H
SRC_DSADCSRA0
DSADC SRA0 Service Request (0B54_H) Reset Value: 0000 0000_H
SRC_DSADCSRMM (m=2-3)
DSADC SRMm Service Request (0B50_H+m*8_H) Reset Value: 0000 0000_H
SRC_DSADCSRAM (m=2-3)
DSADC SRAM Service Request (0B54_H+m*8_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOV CLR	IOV	SET R	CLR R	SRR	0	ECC						
rh	w	rh	w	rh	w	w	rh	r	rwh						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TOS		SRE		0		SRPN							
r		rw		rw		r		rw							

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority ... FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled

Interrupt Router (IR)

Field	Bits	Type	Description
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated Other bit combinations are reserved.
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

Interrupt Router (IR)

SRC_ERAYINT0		
E-RAY Service Request 0	(0BE0 _H)	Reset Value: 0000 0000 _H
SRC_ERAYINT1		
E-RAY Service Request 1	(0BE4 _H)	Reset Value: 0000 0000 _H
SRC_ERAYTINT0		
E-RAY Timer Interrupt 0 Service Request	(0BE8 _H)	Reset Value: 0000 0000 _H
SRC_ERAYTINT1		
E-RAY Timer Interrupt 1 Service Request	(0BEC _H)	Reset Value: 0000 0000 _H
SRC_ERAYNDAT0		
E-RAY New Data 0 Service Request	(0BF0 _H)	Reset Value: 0000 0000 _H
SRC_ERAYNDAT1		
E-RAY New Data 1 Service Request	(0BF4 _H)	Reset Value: 0000 0000 _H
SRC_ERAYMBSC0		
E-RAY Message Buffer Status Changed 0 Service Request	(0BF8 _H)	Reset Value: 0000 0000 _H
SRC_ERAYMBSC1		
E-RAY Message Buffer Status Changed 1 Service Request	(0BFC _H)	Reset Value: 0000 0000 _H
SRC_ERAYOBSY		
E-RAY Output Buffer Busy Service Request	(0C00 _H)	Reset Value: 0000 0000 _H
SRC_ERAYIBUSY		
E-RAY Input Buffer Busy Service Request	(0C04 _H)	Reset Value: 0000 0000 _H
SRC_PMU0m (m=0-1)		
PMU 0 Service Request m	(0C30 _H +m*04 _H)	Reset Value: 0000 0000 _H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOV CLR	IOV	SET R	CLR R	SRR	0	ECC						
rh	w	rh	w	rh	w	w	rh	r	rwh						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TOS		SRE		0		SRPN							
r		rw		rw		r		rw							

Interrupt Router (IR)

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.

Interrupt Router (IR)

Field	Bits	Type	Description
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

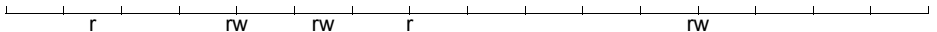
Interrupt Router (IR)

SRC_SCUDTS		
SCU DTS Busy Service Request	(0CD0 _H)	Reset Value: 0000 0000 _H
SRC_SCUERUm (m=0-3)		
SCU ERU Service Request m	(0CD4 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_SCR		
Stand By Controller Service Request	(0D00 _H)	Reset Value: 0000 0000 _H
SRC_SMUm (m=0-2)		
SMU Service Request m	(0D10 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_PSI5m (m=0-7)		
PSI5 Service Request m	(0D30 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_CIFMI		
CIF MI Service Request	(0DA0 _H)	Reset Value: 0000 0000 _H
SRC_CIFMI EP		
CIF MI EP Service Request	(0DA4 _H)	Reset Value: 0000 0000 _H
SRC_CIFISP		
CIF ISP Service Request	(0DA8 _H)	Reset Value: 0000 0000 _H
SRC_CIFMJPEG		
CIF MJPEG Service Request	(0DAC _H)	Reset Value: 0000 0000 _H
SRC_LMU		
LMU Service Request	(0DE0 _H)	Reset Value: 0000 0000 _H
SRC_PSI5Sm (m=0-7)		
PSI5-S Service Request m	(0DF0 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_FFTDONE		
FFT Done Service Request	(0FC0 _H)	Reset Value: 0000 0000 _H
SRC_FFERR		
FFT Error Service Request	(0FC4 _H)	Reset Value: 0000 0000 _H
SRC_FFTRFS		
FFT Ready For Start Service Request	(0FC8 _H)	Reset Value: 0000 0000 _H
SRC_GPSR0m (m=0-3)		
General Purpose Service Request 0m	(1000 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_GPSR1m (m=0-3)		
General Purpose Service Request 1m	(1020 _H +m*4 _H)	Reset Value: 0000 0000 _H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOVC LR	IOV	SET R	CLR R	SRR	0	ECC						
rh	w	rh	w	rh	w	w	rh	r	rwh						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			TOS	SRE	0			SRPN							

Interrupt Router (IR)

SRC_SCUDTS		
SCU DTS Busy Service Request	(0CD0 _H)	Reset Value: 0000 0000 _H
SRC_SCUERUm (m=0-3)		
SCU ERU Service Request m	(0CD4 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_SCR		
Stand By Controller Service Request	(0D00 _H)	Reset Value: 0000 0000 _H
SRC_SMUm (m=0-2)		
SMU Service Request m	(0D10 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_PSI5m (m=0-7)		
PSI5 Service Request m	(0D30 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_CIFMI		
CIF MI Service Request	(0DA0 _H)	Reset Value: 0000 0000 _H
SRC_CIFMI EP		
CIF MI EP Service Request	(0DA4 _H)	Reset Value: 0000 0000 _H
SRC_CIFISP		
CIF ISP Service Request	(0DA8 _H)	Reset Value: 0000 0000 _H
SRC_CIFMJPEG		
CIF MJPEG Service Request	(0DAC _H)	Reset Value: 0000 0000 _H
SRC_LMU		
LMU Service Request	(0DE0 _H)	Reset Value: 0000 0000 _H
SRC_PSI5Sm (m=0-7)		
PSI5-S Service Request m	(0DF0 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_FFTDONE		
FFT Done Service Request	(0FC0 _H)	Reset Value: 0000 0000 _H
SRC_FFERR		
FFT Error Service Request	(0FC4 _H)	Reset Value: 0000 0000 _H
SRC_FFTRFS		
FFT Ready For Start Service Request	(0FC8 _H)	Reset Value: 0000 0000 _H
SRC_GPSR0m (m=0-3)		
General Purpose Service Request 0m	(1000 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_GPSR1m (m=0-3)		
General Purpose Service Request 1m	(1020 _H +m*4 _H)	Reset Value: 0000 0000 _H



Interrupt Router (IR)

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.

Interrupt Router (IR)

Field	Bits	Type	Description
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

Interrupt Router (IR)

SRC_GTMAEIIIRQ GTM AEI Shared Service Request (1600 _H)	Reset Value: 0000 0000 _H
SRC_GTMARUIRQm (m=0-2) GTM ARU Shared Service Request m (1604 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_GTMBCIRQ GTM BRC Shared Service Request (1614 _H)	Reset Value: 0000 0000 _H
SRC_GTMCMPIRQ GTM CMP Shared Service Request (1618 _H)	Reset Value: 0000 0000 _H
SRC_GTMSPE0IRQ GTM SPE0 Shared Service Request (161C _H)	Reset Value: 0000 0000 _H
SRC_GTMSPE1IRQ GTM SPE1 Shared Service Request (1620 _H)	Reset Value: 0000 0000 _H
SRC_GTMPSM0m (m=0-7) GTM PSM0 Shared Service Request m (162C _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_GTMDPLLm (m=0-26) GTM DPLL Service Request m (16A4 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_GTMERR GTM Error Service Request (1770 _H)	Reset Value: 0000 0000 _H
SRC_GTMTIM0m (m=0-7) GTM TIM0 Shared Service Request m (1780 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_GTMTIM1m (m=0-7) GTM TIM1 Shared Service Request m (17A0 _H +m*4 _H)	Reset Value: 0000 0000 _H
SRC_GTMTIM2m (m=0-7) GTM TIM2 Shared Service Request m (17C0 _H +m*4 _H)	Reset Value: 0000 0000 _H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOV CLR	IOV	SET R	CLR R	SRR	0	ECC						
rh	w	rh	w	rh	w	w	rh	r	rwh						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		TOS		SRE		0			SRPN						
r		rw		rw		r			rw						

Interrupt Router (IR)

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.

Interrupt Router (IR)

Field	Bits	Type	Description
SWS	29	rh	SW Sticky Bit 0_B No interrupt was initiated via SETR 1_B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0_B No action 1_B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

SRC_GTM MCS0m (m=0-7)
GTM MCS0 Shared Service Request m
 $(1980_H + m * 4_H)$
Reset Value: 0000 0000_H
SRC_GTM MCS1m (m=0-7)
GTM MCS1 Shared Service Request m
 $(19A0_H + m * 4_H)$
Reset Value: 0000 0000_H
SRC_GTM MCS2m (m=0-7)
GTM MCS2 Shared Service Request m
 $(19C0_H + m * 4_H)$
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOV CLR	IOV	SET R	CLR R	SRR	0					ECC		
rh	w	rh	w	rh	w	w	rh	r					rwh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0		TOS	SRE	0							SRPN			
	r		rw	rw	r							rw			

Interrupt Router (IR)

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.

Interrupt Router (IR)

Field	Bits	Type	Description
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

Interrupt Router (IR)

SRC_GTMATOM0m (m=0-7) GTM TOM0 Shared Service Request m (1B80 _H +m*4 _H)	Reset Value: 0000 0000_H
SRC_GTMATOM1m (m=0-7) GTM TOM1 Shared Service Request m (1BA0 _H +m*4 _H)	Reset Value: 0000 0000_H
SRC_GTMATOM0m (m=0-3) GTM ATOM0 Shared Service Request m (1D80 _H +m*4 _H)	Reset Value: 0000 0000_H
SRC_GTMATOM1m (m=0-3) GTM ATOM1 Shared Service Request m (1D90 _H +m*4 _H)	Reset Value: 0000 0000_H
SRC_GTMATOM2m (m=0-3) GTM ATOM2 Shared Service Request m (1DA0 _H +m*4 _H)	Reset Value: 0000 0000_H
SRC_GTMATOM3m (m=0-3) GTM ATOM3 Shared Service Request m (1DB0 _H +m*4 _H)	Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOV CLR	IOV	SET R	CLR R	SRR	0					ECC		
rh	w	rh	w	rh	w	w	rh	r					rwh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0		TOS	SRE	0							SRPN			
	r		rw	rw	r							rw			

Field	Bits	Type	Description
SRPN	[7:0]	rw	<p>Service Request Priority Number</p> <p>00_H Service request is on lowest priority</p> <p>01_H Service request is one before lowest priority</p> <p>... ..</p> <p>FF_H Service request is on highest priority</p> <p><i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i></p>

Interrupt Router (IR)

Field	Bits	Type	Description
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.

Interrupt Router (IR)

Field	Bits	Type	Description
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

SRC_GTMMCSW0m (m=0-2)
GTM Multi Channel Sequencer 0 Service Request m
 (1F00_H+m*4_H)

Reset Value: 0000 0000_H
SRC_GTMMCSW1m (m=0-2)
GTM Multi Channel Sequencer 1 Service Request m
 (1F40_H+m*4_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	SWS CLR	SWS	IOV CLR	IOV	SET R	CLR R	SRR	0					ECC		
rh	w	rh	w	rh	w	w	rh	r							rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0		TOS	SRE	0										SRPN
	r		rw	rw	r										rw

Field	Bits	Type	Description
SRPN	[7:0]	rw	Service Request Priority Number 00 _H Service request is on lowest priority 01 _H Service request is one before lowest priority ... FF _H Service request is on highest priority <i>Note: For a CPU 01H is the lowest priority as 00H is never serviced. For the DMA 00H triggers channel 0</i>
SRE	10	rw	Service Request Enable 0 _B Service request is disabled 1 _B Service request is enabled
TOS	[12:11]	rw	Type of Service Control 0 _H CPU0 service is initiated 1 _H CPU1 service is initiated 3 _H DMA service is initiated

Interrupt Router (IR)

Field	Bits	Type	Description
ECC	[21:16]	rwh	ECC
SRR	24	rh	Service Request Flag 0 _B No service request is pending 1 _B A service request is pending
CLRR	25	w	Request Clear Bit 0 _B No action 1 _B Clear SRR; bit value is not stored; read always returns 0; no action if SETR is set in parallel.
SETR	26	w	Request Set Bit 0 _B No action 1 _B Set SRR and SWS; SRR bit value is not stored; read always returns 0; no action if CLRR is set in parallel.
IOV	27	rh	Interrupt Trigger Overflow Bit 0 _B No Interrupt Trigger Overflow detected 1 _B Interrupt Overflow Detected.
IOVCLR	28	w	Interrupt Trigger Overflow Clear Bit 0 _B No action 1 _B Clear IOV; bit value is not stored; read always returns 0.
SWS	29	rh	SW Sticky Bit 0 _B No interrupt was initiated via SETR 1 _B Interrupt was initiated via SETR
SWSCLR	30	w	SW Sticky Clear Bit 0 _B No action 1 _B Clear SWS; bit value is not stored; read always returns 0.
Reserved	31	rh	Reserved Read as 0 or 1; should be written with 0.
0	[23:22], [15:13], [9:8]	r	Reserved Read as 0; should be written with 0.

18 System Timer (STM)

This chapter describes the System Timer (STM). The TC26x B-Step's STM is designed for global system timing applications requiring both high precision and long period.

18.1 Overview

The STM has the following features:

- Free-running 64-bit counter
- All 64 bits can be read synchronously
- Different 32-bit portions of the 64-bit counter can be read synchronously
- Flexible service request generation based on compare match with partial STM content
- Counting starts automatically after an Application Reset
- STM registers are reset by an Application Reset if bit ARSTDIS.STMxDIS is cleared. If bit ARSTDIS.STMxDIS is set, the STM registers are not reset.

Special STM register semantics provide synchronous views of the entire 64-bit counter, or 32-bit subsets at different levels of resolution.

18.2 Operation

The STM is an upward counter, running at frequency f_{STM} . In case of an Application Reset, the STM is reset if bit SCU_ARSTDIS.STMxDIS is cleared. After reset, the STM is enabled and immediately starts counting up. It is not possible to affect the content of the timer during normal operation. The timer registers can only be read but not written to.

The STM can be optionally disabled for power-saving purposes, or suspended for debugging purposes. In suspend mode, the STM clock is stopped but all registers are still readable.

Due to the 64-bit width of the STM, it is not possible to read its entire content with one instruction. It needs to be read with two load instructions. Since the timer would continue to count between the two load operations, there is a chance that the two values read are not consistent (due to possible overflow from the low part of the timer to the high part between the two read operations). To enable a synchronous and consistent reading of the STM content, a capture register (CAP) is implemented. It latches the content of the high part of the STM each time when one of the registers TIM0 to TIM5 is read. Thus, CAP holds the upper value of the timer at exactly the same time when the lower part is read. The second read operation would then read the content of the CAP to get the complete timer value.

The STM can also be read in sections from seven registers, TIM0 through TIM6, that select increasingly higher-order 32-bit ranges of the STM. These can be viewed as individual 32-bit timers, each with a different resolution and timing range.

System Timer (STM)

The content of the 64-bit System Timer can be compared against the content of two compare values stored in the CMP0 and CMP1 registers. Service requests can be generated on a compare match of the STM with the CMP0 or CMP1 registers.

Figure 18-1 provides an overview on the STM module. It shows the options for reading parts of the STM content.

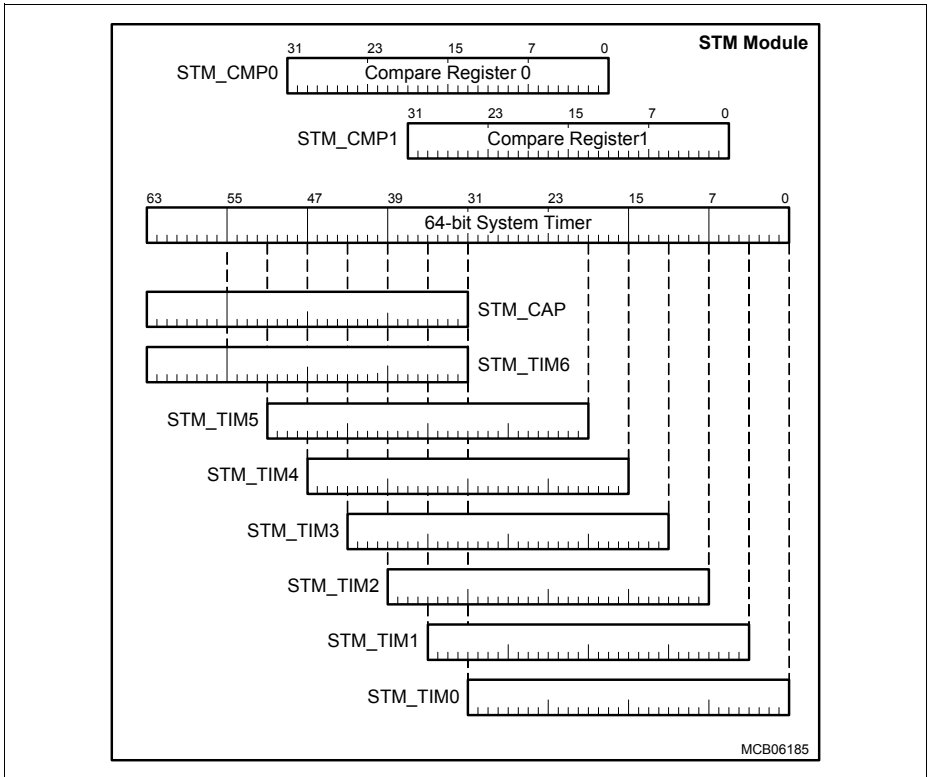


Figure 18-1 General Block Diagram of the STM Module

18.2.1 Compare Register Operation

The content of the 64-bit STM can be compared against the content of two compare values stored in the CMP0 and CMP1 registers. Service requests can be generated on a compare match of the STM with the CMP0 or CMP1 registers.

Two parameters are programmable for the compare operation:

1. The width of the relevant bits in registers CMP0/CMP1 (compare width MSIZE_x) that is taken for the compare operation can be programmed from 0 to 31.
2. The first bit location in the 64-bit STM that is taken for the compare operation can be programmed from 0 to 31.

These programming capabilities make compare functionality very flexible. It even makes it possible to detect bit transitions of a single bit n ($n = 0$ to 31) within the 64-bit STM by setting $MSIZE = 0$ and $MSTART = n$.

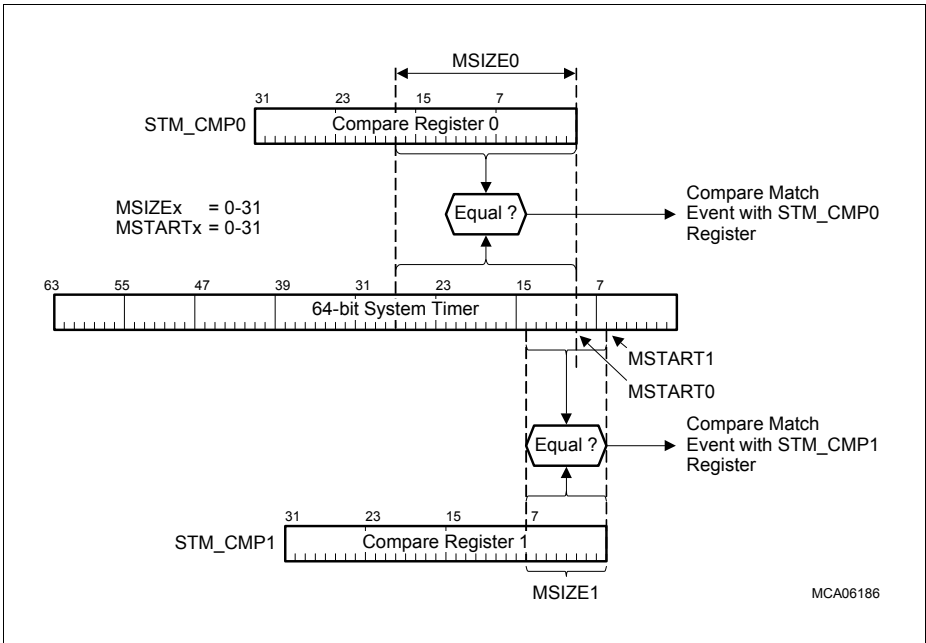


Figure 18-2 Compare Mode Operation

Figure 18-2 shows an example of the compare operation. In this example the following parameters are programmed:

- $MSIZE0 = 10001_B = 17_D$; $MSTART0 = 01010_B = 10_D$
- $MSIZE1 = 00111_B = 7_D$; $MSTART1 = 00111_B = 7_D$

System Timer (STM)

A compare operation with MSIZE not equal 11111_B always implies that the compared value as stored in the CMP register is right-extended with zeros. This means that in the example of **Figure 18-2**, the compare register content CMP0[17:0] plus ten zero bits right-extended is compared with STM[27:0] with STM[9:0] = 000_H. In case of register CMP1, STM[14:0] with STM[6:0] = 00_H are compared with CMP1[9:0] plus seven zero bits right-extended.

18.2.2 Compare Match Interrupt Control

The compare match interrupt control logic is shown in **Figure 18-3**. Each CMPx register has its compare match interrupt request flag (ICR.CMPxIR) that is set by hardware on a compare match event. The interrupt request flags can be set (ISSR.CMPxIRS) or cleared (ISSR.CMPxIRR) by software. Note that setting ICR.CMPxIR by writing a 1 into ISSR.CMPxIRS does not generate an interrupt at STMIRx. The compare match interrupts from CMP0 and CMP1 can be further directed by ICR.CMPxOS to either output signal STMIR0 or STMIR1.

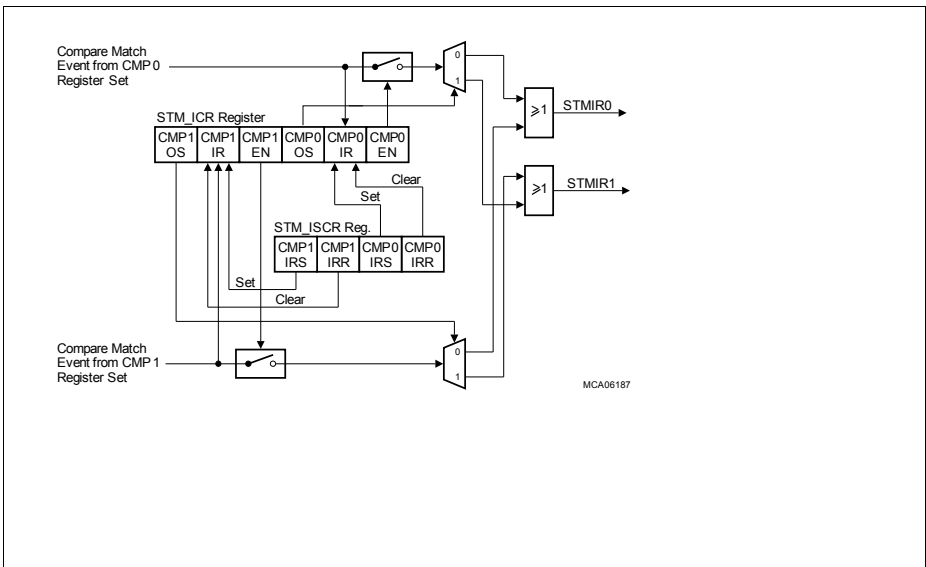


Figure 18-3 STM Interrupt Control

The compare match interrupt flags ICR.CMPxIR are immediately set after an STM reset operation, caused by a compare match event with the reset values of the STM and the compare registers CMPx. This setting of the CMPxIR flags does not directly generate compare match interrupts because the compare match interrupts are automatically disabled after a STM reset operation (CMPxEN = 0). Therefore, before enabling a

System Timer (STM)

compare match interrupt after a STM Application Reset, the CMPxIR flags should be cleared by software (writing register ISSR with CMPxIRR set). Otherwise, undesired compare match interrupt events are triggered.

18.2.3 Using Multiple STMs

For systems that include multiple CPUs there are also multiple STMs available. Each STM is aimed to serve as time base for one individual CPU operating system main scheduler clock trigger. This is done by the usage of one compare register and the associated interrupt generating the trigger.

All STM modules are connected and controlled by f_{STM} and can therefore operate on the same frequency if desired.

Note: For synchronisation of CPUs they contain timer called temporal protection system.

18.2.4 STM as Reset Trigger

A compare match triggered by an CMP0 event can generate a reset in the system. The reset has to be enable for each STM module individually in register SCU_RSTCON.

18.3 STM Registers

This section describes the STM registers of the STM.

Table 18-1 Registers Address Space

Module	Base Address	End Address	Note
STM0	F000 0000 _H	F000 00FF _H	STM for CPU0
STM1	F000 0100 _H	F000 01FF _H	STM for CPU1

Table 18-2 Registers Overview - STM Control Registers

Short Name	Description	Offset Addr.	Access Mode		Reset	Description See
			Read	Write		
CLC	Clock Control Register	00 _H	U, SV	SV, E, P	Application ¹⁾	Page 18-8
-		04 _H	BE	BE	-	-
ID	Identification Register	08 _H	U, SV	BE	Application	Page 18-9
-		0C _H	BE	BE	-	-
TIM0	Timer 0 Register	10 _H	U, SV	BE	Application	Page 18-10
TIM1	Timer 1 Register	14 _H	U, SV	BE	Application	Page 18-10

Table 18-2 Registers Overview - STM Control Registers

Short Name	Description	Offset Addr.	Access Mode		Reset	Description See
			Read	Write		
TIM2	Timer 2 Register	18 _H	U, SV	BE	Application	Page 18-11
TIM3	Timer 3 Register	1C _H	U, SV	BE	Application	Page 18-11
TIM4	Timer 4 Register	20 _H	U, SV	BE	Application	Page 18-11
TIM5	Timer 5 Register	24 _H	U, SV	BE	Application	Page 18-12
TIM6	Timer 6 Register	28 _H	U, SV	BE	Application	Page 18-12
CAP	Timer Capture Register	2C _H	U, SV	BE	Application	Page 18-13
CMP0	Compare Register 0	30 _H	U, SV	U, SV	Application	Page 18-13
CMP1	Compare Register 1	34 _H	U, SV	U, SV	Application	Page 18-13
CMCON	Compare Match Control Register	38 _H	U, SV	U, SV	Application	Page 18-15
ICR	Interrupt Control Register	3C _H	U, SV	U, SV	Application	Page 18-17
ISCR	Interrupt Set/Clear Register	40 _H	U, SV	U, SV	Application	Page 18-19
-		44 _H - 4C _H	BE	BE	-	-
TIM0SV	Timer 0 Register Second View	50 _H	U, SV	BE	Application	Page 18-10
CAPSV	Timer Capture Register Second View	54 _H	U, SV	BE	Application	Page 18-13
-		58 _H - E4 _H	BE	BE	-	-
OCS	OCDS Control and Status Register	E8 _H	U, SV	SV, P	Debug	Page 18-20
KRSTCLR	Reset Status Clear Register	EC _H	U, SV	SV, E, P	Application	Page 18-24
KRST1	Reset Control Register 1	F0 _H	U, SV	SV, E, P	Application	Page 18-23
KRST0	Reset Control Register 0	F4 _H	U, SV	SV, E, P	Application	Page 18-22

Table 18-2 Registers Overview - STM Control Registers

Short Name	Description	Offset Addr.	Access Mode		Reset	Description See
			Read	Write		
ACCEN1	Access Enable Register 1	F8 _H	U, SV	SV, SE	Application	Page 18-21
ACCEN0	Access Enable Register 0	FC _H	U, SV	SV, SE	Application	Page 18-21

1) These registers are reset by an application reset if bit ARSTDIS.STMxDIS is cleared.

System Timer (STM)

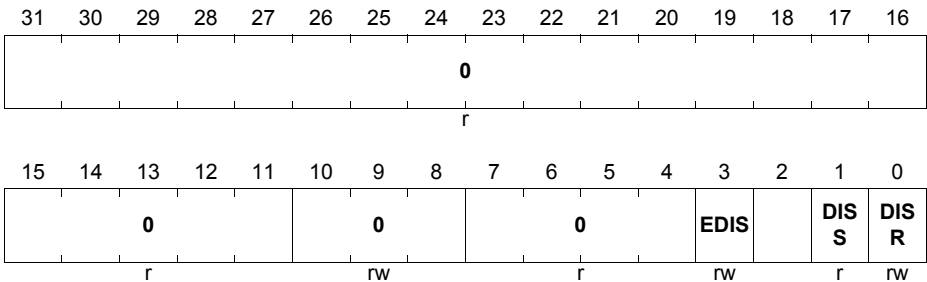
18.3.1 Clock Control Register

The STM clock control register is used to switch the STM on or off and to control its input clock rate. After reset, the STM is always enabled and starts counting. The STM can be disabled by setting bit DISR to 1.

CLC

Clock Control Register

 (00_H)

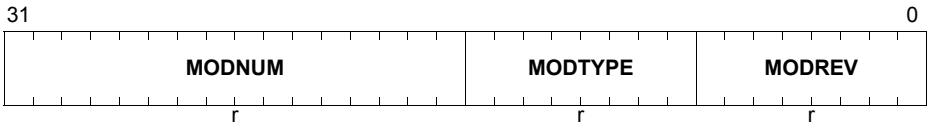
 Reset Value: 0000 0000_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the STM module. 0 _B No disable requested 1 _B Disable requested <i>Note: f_{STM} is generated by the CCU.</i>
DISS	1	r	Module Disable Status Bit Bit indicates the current status of the STM module. 0 _B STM module is enabled 1 _B STM module is disabled
EDIS	3	rw	Sleep Mode Enable Control Used for module sleep mode control.
0	[10:8]	rw	Reserved Should be written with 0.
0	2, [7:4], [31:11]	r	Reserved Read as 0; should be written with 0.

The STM Module Identification Register ID contains read-only information about the module version.

System Timer (STM)

ID
Module Identification Register (08_H) **Reset Value: 0000 C0XX_H**



Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number MODREV defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MODTYPE	[15:8]	r	Module Type This bit field defines the module as a 32-bit module: C0 _H
MODNUM	[31:16]	r	Module Number Value This bit field defines the module identification number for the STM: 0000 _H

System Timer (STM)

18.3.2 Timer/Capture Registers

Registers TIM0 to TIM6 provide 32-bit views at varying resolutions of the underlying STM counter.

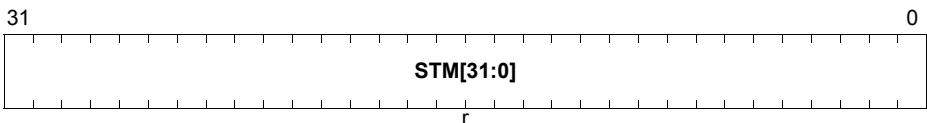
Register TIM0SV address the STM[31:0] bits at a second optional address as TIM0.

TIM0

Timer Register 0 (10_H) **Reset Value: 0000 0000_H**

TIM0SV

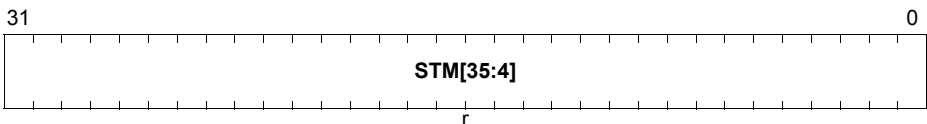
Timer Register 0 Second View (50_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
STM[31:0]	[31:0]	r	System Timer Bits [31:0] This bit field contains bits [31:0] of the 64-bit STM.

TIM1

Timer Register 1 (14_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
STM[35:4]	[31:0]	r	System Timer Bits [35:4] This bit field contains bits [35:4] of the 64-bit STM.

System Timer (STM)

TIM2

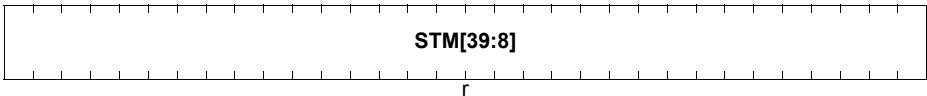
Timer Register 2

 (18_H)

 Reset Value: 0000 0000_H

31

0



Field	Bits	Type	Description
STM[39:8]	[31:0]	r	System Timer Bits [39:8] This bit field contains bits [39:8] of the 64-bit STM.

TIM3

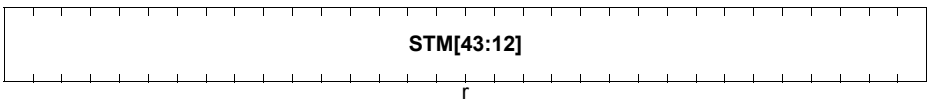
Timer Register 3

 (1C_H)

 Reset Value: 0000 0000_H

31

0



Field	Bits	Type	Description
STM[43:12]	[31:0]	r	System Timer Bits [43:12] This bit field contains bits [43:12] of the 64-bit STM.

TIM4

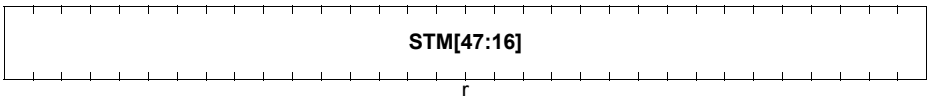
Timer Register 4

 (20_H)

 Reset Value: 0000 0000_H

31

0



Field	Bits	Type	Description
STM[47:16]	[31:0]	r	System Timer Bits [47:16] This bit field contains bits [47:16] of the 64-bit STM.

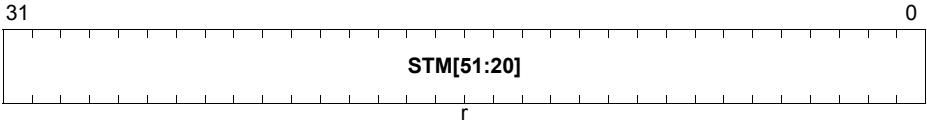
System Timer (STM)

TIM5

Timer Register 5

(24_H)

Reset Value: 0000 0000_H



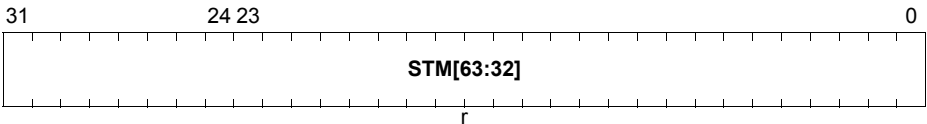
Field	Bits	Type	Description
STM[51:20]	[31:0]	r	System Timer Bits [51:20] This bit field contains bits [51:20] of the 64-bit STM.

TIM6

Timer Register 6

(28_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
STM[63:32]	[31:0]	r	System Timer Bits [63:32] This bit field contains bits [63:32] of the 64-bit STM.

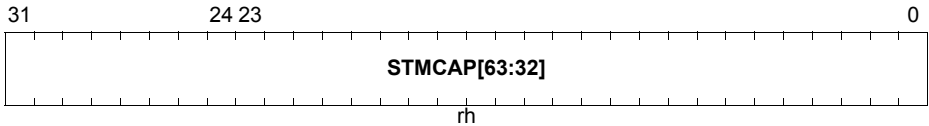
System Timer (STM)

CAP

Timer Capture Register (2C_H) Reset Value: 0000 0000_H

CAPSV

Timer Capture Register Second View (54_H) Reset Value: 0000 0000_H



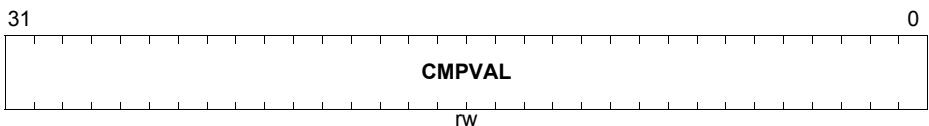
Field	Bits	Type	Description
STMTCAP[63:32]	[31:0]	rh	<p>Captured System Timer Bits [63:32] The capture register STMTCAP always captures the STM bits [63:32] when one of the registers TIM0 to TIM5 and TIMOSV is read. This capture operation is performed in order to enable software to operate with a coherent value of all the 64 STM bits at one time stamp. This bit field contains bits [63:32] of the 64-bit STM.</p> <p><i>Note: Reading register TIMOSV capture also the read value for register TIM6. In this way reading TIMOSV followed by CAP6SV delivers the timer values for the first read request.</i></p>

18.3.3 Compare Registers

The compare register CMPx holds up to 32-bits; its value is compared to the value of the STM.

CMPx (x = 0-1)

Compare Register x (30_H+x*4_H) Reset Value: 0000 0000_H



System Timer (STM)

Field	Bits	Type	Description
CMPVAL	[31:0]	rw	Compare Value of Compare Register x This bit field holds up to 32 bits of the compare value (right-adjusted).

System Timer (STM)

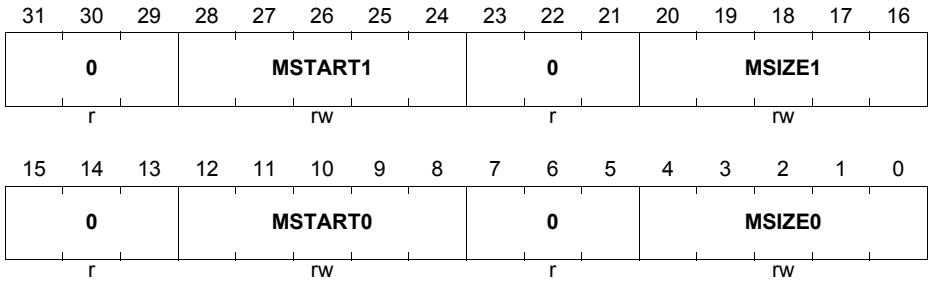
The STM Compare Match Control Register controls the parameters of the compare logic.

CMCON

Compare Match Control Register

(38_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
MSIZE0	[4:0]	rw	<p>Compare Register Size for CMP0</p> <p>This bit field determines the number of bits in register CMP0 (starting from bit 0) that are used for the compare operation with the System Timer.</p> <p>00000_B CMP0[0] used for compare operation 00001_B CMP0[1:0] used for compare operation ... 11110_B CMP0[30:0] used for compare operation 11111_B CMP0[31:0] used for compare operation</p>
MSTART0	[12:8]	rw	<p>Start Bit Location for CMP0</p> <p>This bit field determines the lowest bit number of the 64-bit STM that is compared with the content of register CMP0 bit 0. The number of bits to be compared is defined by bit field MSIZE0.</p> <p>00000_B STM[0] is the lowest bit number 00001_B STM[1] is the lowest bit number ... 11110_B STM[30] is the lowest bit number 11111_B STM[31] is the lowest bit number</p>

System Timer (STM)

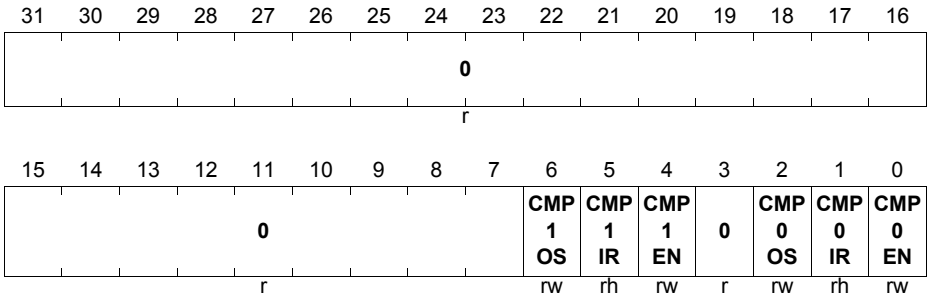
Field	Bits	Type	Description
MSIZE1	[20:16]	rw	<p>Compare Register Size for CMP1</p> <p>This bit field determines the number of bits in register CMP1 (starting from bit 0) that are used for the compare operation with the System Timer.</p> <p>00000_B CMP1[0] used for compare operation 00001_B CMP1[1:0] used for compare operation ... 11110_B CMP1[30:0] used for compare operation 11111_B CMP1[31:0] used for compare operation</p>
MSTART1	[28:24]	rw	<p>Start Bit Location for CMP1</p> <p>This bit field determines the lowest bit number of the 64-bit STM that is compared with the content of register CMP1 bit 0. The number of bits to be compared is defined by bit field MSIZE1.</p> <p>00000_B STM[0] is the lowest bit number 00001_B STM[1] is the lowest bit number ... 11110_B STM[30] is the lowest bit number 11111_B STM[31] is the lowest bit number</p>
0	[7:5], [15:13], [23:21], [31:29]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

18.3.4 Interrupt Registers

The two compare match interrupts of the STM are controlled by the STM Interrupt Control Register.

ICR

Interrupt Control Register (3C_H) **Reset Value: 0000 0000_H**



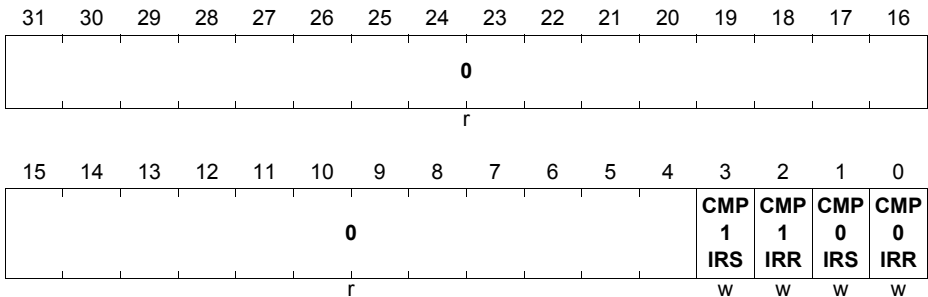
Field	Bits	Type	Description
CMP0EN	0	rw	<p>Compare Register CMP0 Interrupt Enable Control This bit enables the compare match interrupt with compare register CMP0.</p> <p>0_B Interrupt on compare match with CMP0 disabled 1_B Interrupt on compare match with CMP0 enabled</p>
CMP0IR	1	rh	<p>Compare Register CMP0 Interrupt Request Flag This bit indicates whether or not a compare match interrupt request of compare register CMP0 is pending. CMP0IR must be cleared by software.</p> <p>0_B A compare match interrupt has not been detected since the bit has been cleared for the last time. 1_B A compare match interrupt has been detected. CMP0IR must be cleared by software and can be set by software, too (see CMPISCR register). After a STM reset operation, CMP0IR is immediately set as a result of a compare match event with the reset values of the STM and the compare registers CMP0.</p>

System Timer (STM)

Field	Bits	Type	Description
CMP0OS	2	rw	Compare Register CMP0 Interrupt Output Selection This bit determines the interrupt output that is activated on a compare match event of compare register CMP0. 0 _B Interrupt output STMIR0 selected 1 _B Interrupt output STMIR1 selected
CMP1EN	4	rw	Compare Register CMP1 Interrupt Enable Control This bit enables the compare match interrupt with compare register CMP1. 0 _B Interrupt on compare match with CMP1 disabled 1 _B Interrupt on compare match with CMP1 enabled
CMP1IR	5	rh	Compare Register CMP1 Interrupt Request Flag This bit indicates whether or not a compare match interrupt request of compare register CMP1 is pending. CMP1IR must be cleared by software. 0 _B A compare match interrupt has not been detected since the bit has been cleared for the last time. 1 _B A compare match interrupt has been detected. CMP1IR must be cleared by software and can be set by software, too (see CMPISCR register). After a STM reset, CMP1IR is immediately set as a result of a compare match event with the reset values of the STM and the compare register CMP1.
CMP1OS	6	rw	Compare Register CMP1 Interrupt Output Selection This bit determines the interrupt output that is activated on a compare match event of compare register CMP1. 0 _B Interrupt output STMIR0 selected 1 _B Interrupt output STMIR1 selected
0	3, [31:7]	r	Reserved Read as 0; should be written with 0.

System Timer (STM)

The bits in the STM Interrupt Set/Clear Register make it possible to set or cleared the compare match interrupt request status flags of register ICR.

ISCR
Interrupt Set/Clear Register
(40_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CMP0IRR	0	w	Reset Compare Register CMP0 Interrupt Flag 0 _B Bit ICR.CMP0IR is not changed. 1 _B Bit ICR.CMP0IR is cleared.
CMP0IRS	1	w	Set Compare Register CMP0 Interrupt Flag 0 _B Bit ICR.CMP0IR is not changed. 1 _B Bit ICR.CMP0IR is set. The state of bit CMP0IRR is "don't care" in this case.
CMP1IRR	2	w	Reset Compare Register CMP1 Interrupt Flag 0 _B Bit ICR.CMP1IR is not changed. 1 _B Bit ICR.CMP1IR is cleared.
CMP1IRS	3	w	Set Compare Register CMP1 Interrupt Flag 0 _B Bit ICR.CMP1IR is not changed. 1 _B Bit ICR.CMP1IR is set. The state of bit CMP1IRR is "don't care" in this case.
0	[31:4]	r	Reserved Read as 0; should be written with 0.

Note: Reading register ISCR always returns 0000 0000_H.

18.3.5 Interface Registers

OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The OCS register can only be written when the OCDS is enabled.

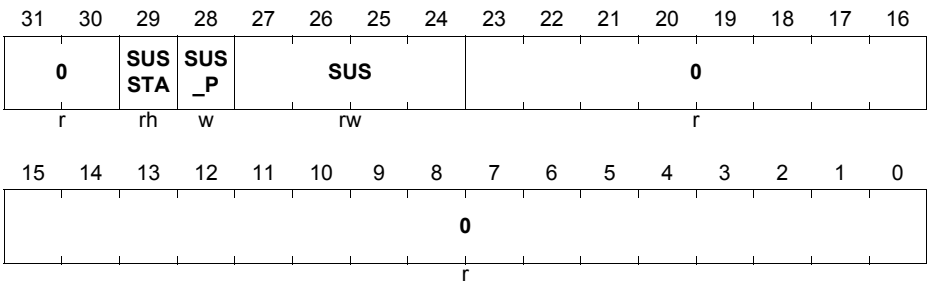
If OCDS is being disabled, the OCS register value will not change.

When OCDS is disabled the OCS suspend control is ineffective.

Write access is 32 bit wide only and requires Supervisor Mode.

OCS

OCDS Control and Status (E8_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Reserved, do not use this combination 2 _H 64-bit counter will be stopped others , Reserved, do not use this combination <i>Note:</i> For details see the OCDS chapter.
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:0], [31:30]	r	Reserved Read as 0; must be written with 0.

System Timer (STM)

The Access Enable Register 0 controls write access for transactions with the on chip bus master TAG ID 00000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 00000_B, EN1 -> TAG ID 00001_B, ... , EN31 -> TAG ID 011111_B.

ACCEN0
Access Enable Register 0
(FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register (ACCEN1)

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID master peripheral mapping).

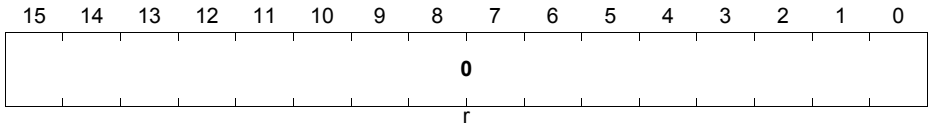
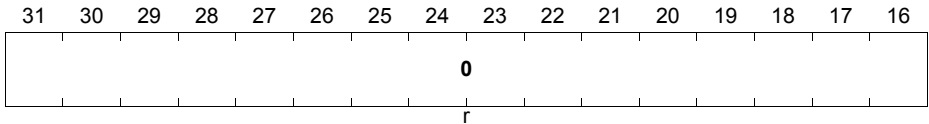
Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000_B, EN1 -> TAG ID 100001_B, ... , EN31 -> TAG ID 111111_B.

ACCEN1

Access Enable Register 1

(F8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

The Kernel Reset Register 0 is used to reset the related module kernel. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers related to the module kernel that should be reset (kernel 0 or kernel 1). In order support modules with two kernel the BPI_FPI provides two set of kernel reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

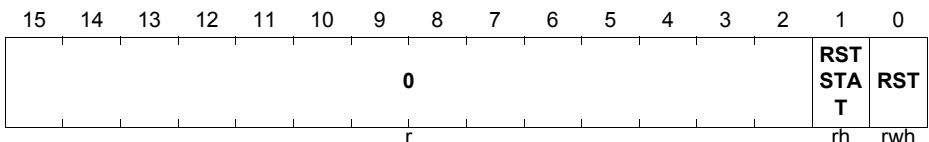
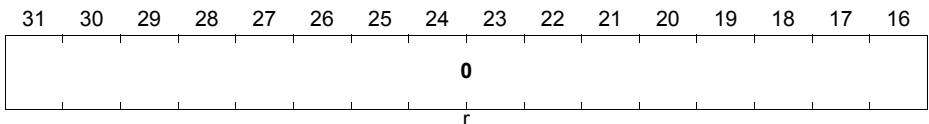
Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing to it with '1'.

KRST0

Kernel Reset Register 0

(F4_H)

Reset Value: 0000 0000_H



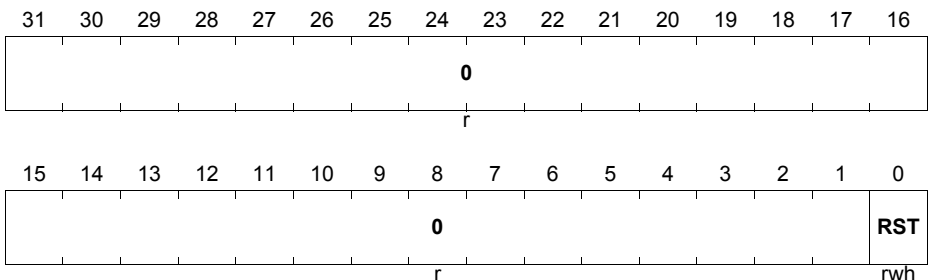
Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0_B No kernel reset was requested 1_B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
RSTSTAT	1	rw	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits. 0_B No kernel reset was executed 1_B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the STM kernel. STM kernel registers related to the Debug Reset (Class 1) are not influenced. To reset the STM kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be cleared with the end of the BPI kernel reset sequence.

KRST1

Kernel Reset Register 1 (F0_H) **Reset Value: 0000 0000_H**

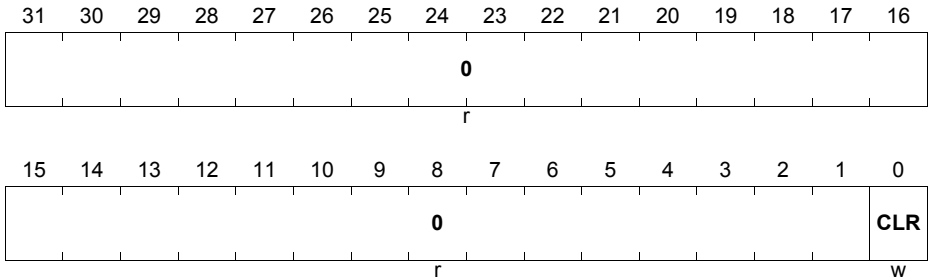


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. 0_B No kernel reset was requested 1_B A kernel reset was requested The RST bit will be cleared (re-set to '0') after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

The Kernel Reset Register Clear register is used to clear the Kernel Reset Status bit (<>_KRST0.RSTSTAT).

KRSTCLR

Kernel Reset Status Clear Register (EC_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0_B No action 1_B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)

19 Asynchronous/Synchronous Interface (ASCLIN)

The main purpose of the ASCLIN module is to provide asynchronous serial communication with external devices using only data-in, data-out signals.

The focus of the module is set to fast and flexible communication: either fast point-to-point or master-to-many slaves communication using the LIN protocol.

Additionally, the module supports the synchronous SPI communication.

Figure 19-1 shows an overview of the ASCLIN module.

Note: The RX, TX, RTS, CTS, SCLKO, SLISO signal names are prefixed with "A" in order to achieve unique names on chip level, distinct from signal names of other communication modules.

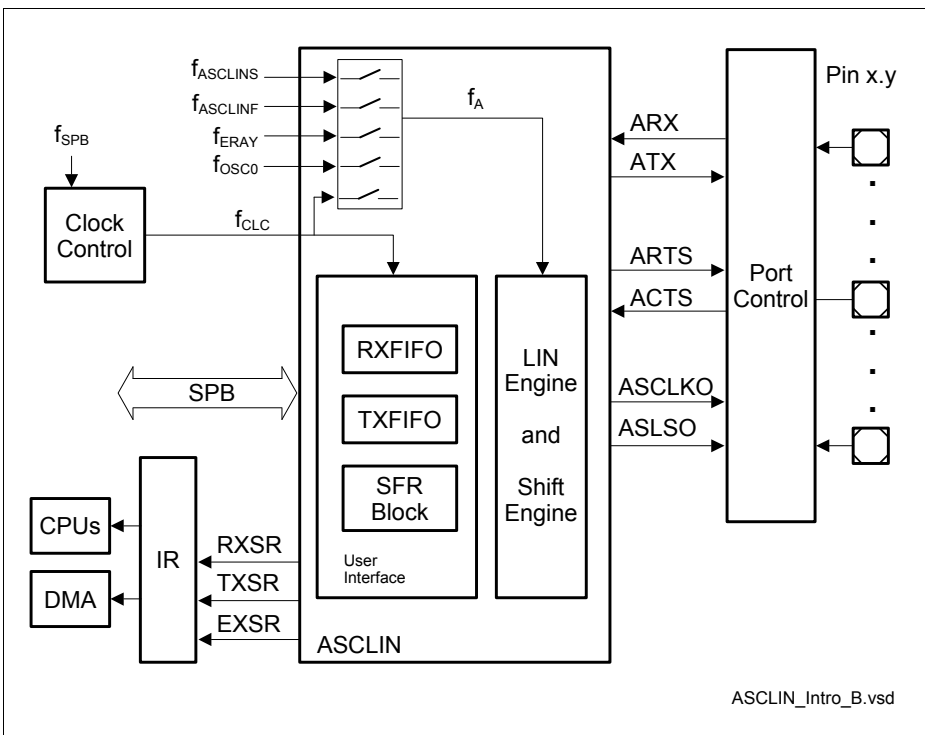


Figure 19-1 Block Diagram of the ASCLIN module.

>> [Registers Overview](#)

>> [BPI_FPI Module Registers](#)

Asynchronous/Synchronous Interface (ASCLIN)

19.1 Feature List

This section describes the features of the ASCLIN module.

General Features

- 16 bytes TxFIFO
- 16 bytes RxFIFO
- Pack / unpack capabilities of the Tx and Rx FIFO
- Interrupt generation
 - On a configurable transmit FIFO level
 - On a configurable receive FIFO level
 - On an error condition (frame, parity, overrun error)
 - On various module internal events (end of ASC/SPI frame, LIN events)
- Interrupt signals capable of triggering either a CPU or a DMA
- Programmable oversampling of 4 to 16 times per bit
- Programmable sampling point position
- Programmable digital glitch filter and median filter for incoming bit stream
- Shift direction LSB first
- Internal loop-back mode

Standard ASC Features

- Full-duplex asynchronous operating modes
 - 7-bit, 8-bit or 9-bit (or up to 16-bit) data frames, LSB first
 - Parity-bit generation/checking
 - One or two stop bits
 - Max baud rate $f_A / 16$ (6.25 MBaud @ 100 MHz f_A module clock)
 - Min. baud rate $f_A / 268\ 435\ 456$ (0.37 Baud @ 100 MHz f_A module clock)
- Optional RTS / CTS handshaking
- Support of
 - JASO D 903

Extended ASC Features

- Programmable oversampling of 4 to 16 times per bit
 - module capability of up to 4 times higher baud rates ($f_A / 4$) then the standard ASC
 - system considerations like pad type, incoming signal quality and accumulated PLL jitter can lead to constraints regarding the usable oversampling ratios (for example $f_A / 8 = 200\text{MHz} / 8 = 25\ \text{MBaud}$)
- Programmable sampling point position

LIN Features

- Support of

Asynchronous/Synchronous Interface (ASCLIN)

- LIN version 1.3
- LIN version 2.0
- LIN version 2.1 and
- J2602
- Break detection
- Break injection
- Sync field generation
- Auto baud detection based on Sync Field measurement
- Optional Collision detection, required for LIN version 2.1
- LIN Watchdogs
 - Header time-out
 - Frame or Response time-out
- Stuck at zero/one monitoring
- Bus idle time monitoring
- Wake-Up
- Minimum CPU load in master mode
 - Single interrupt indicating the end of the frame
- Minimum CPU load in slave mode
 - Single interrupt at the end of the header reception
 - Single interrupt at the end of the response or end of frame
- Standard operation with one interrupt per transmitted or received byte supported

SPI Features

- SPI master modes (slave mode not supported):
 - Four-wire or three-wire (with / without slave select output signal)
- Up to 16-bit data width
- Full-duplex and half-duplex
 - Min. baud rate $f_A / 268\,435\,456$ MBaud (= 0.37 Baud @ 100 MHz f_A module clock)
 - Max. baud rate $f_A / 4$ MBaud (= 25 MBaud @ 100 MHz f_A module clock)
- Programmable leading and trailing delays

Asynchronous/Synchronous Interface (ASCLIN)

19.2 Overview

This section shows the essential sub blocks of the module when used as standard ASC, as LIN or as SPI master.

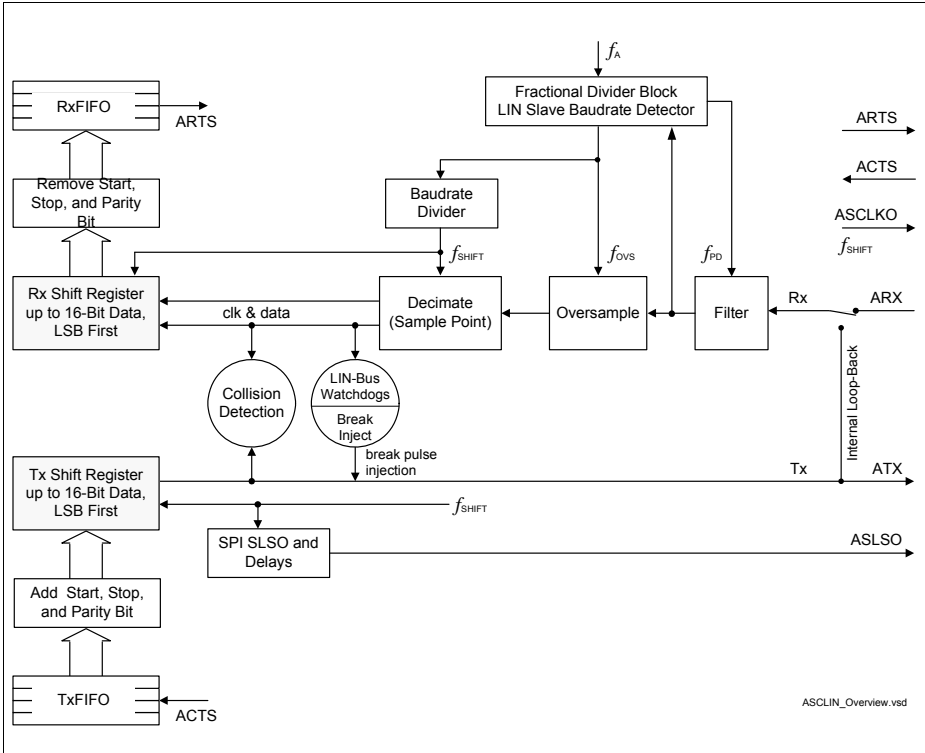


Figure 19-2 Architecture Overview

Note: Collision detection is mandatory only in the LIN specification V2.1.

Note: In LIN mode, both ARX and ATX signal must be connected to the LIN bus in order ASCLIN module to work properly.

Asynchronous/Synchronous Interface (ASCLIN)

19.3 External Signals

The ASCLIN module provides the following external signals:

- Serial clock output ASCLK
- Receive data input ARX (Master Receive MR input in SPI mode)
- Transmit data output ATX (Master Transmit MT output in SPI mode)
- Slave select signal output ASLSO
- Request to send handshake output ARTS
- Clear to send handshake input ACTS

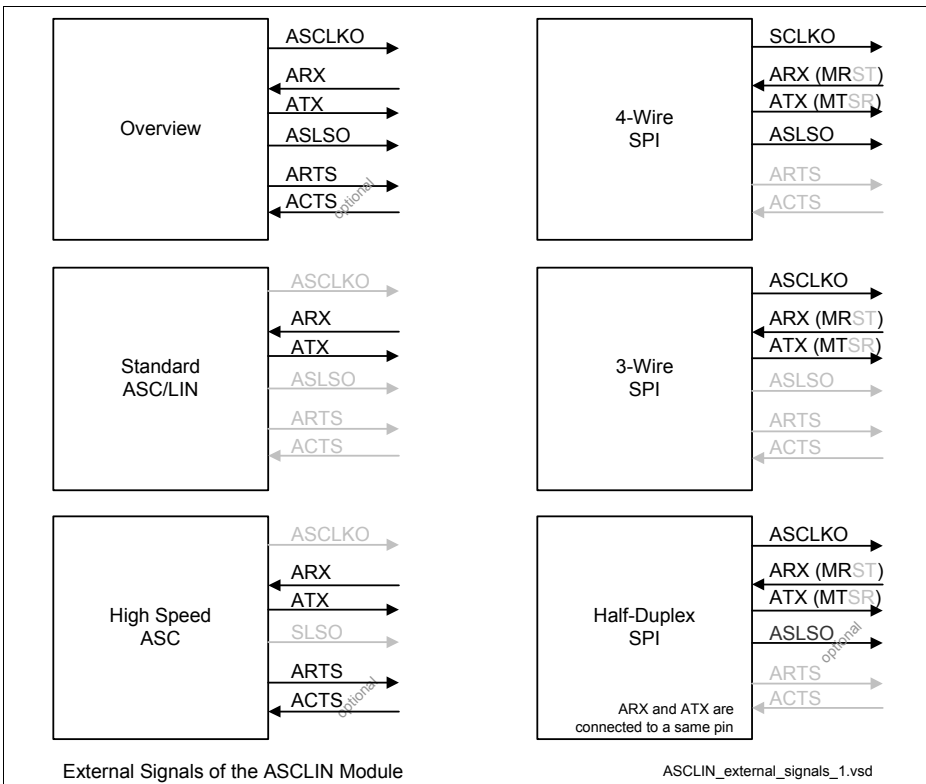


Figure 19-3 External Signals of the ASCLIN Module in Different Modes

Asynchronous/Synchronous Interface (ASCLIN)

19.4 User Interface

The user interface contains a Tx FIFO and an Rx FIFO.

They provide the following services: takes data packets with width optimal for the FPI¹⁾ bus and packs them to serial frames, buffers the FPI bus data, manages handshaking. These features are designed to optimize the use of the module in LIN, high-speed ASC and SPI mode. They also optimize the use of the module in standard ASC mode.

19.4.1 TxFIFO Overview

The Tx FIFO has the ability to pack 16-bit writes from the FPI bus to two up-to-8-bit frames or one up-to-16-bit frame. It also has the ability to pack 32-bit writes from the FPI bus to four up-to-8-bit or two up-to-16-bit frames.

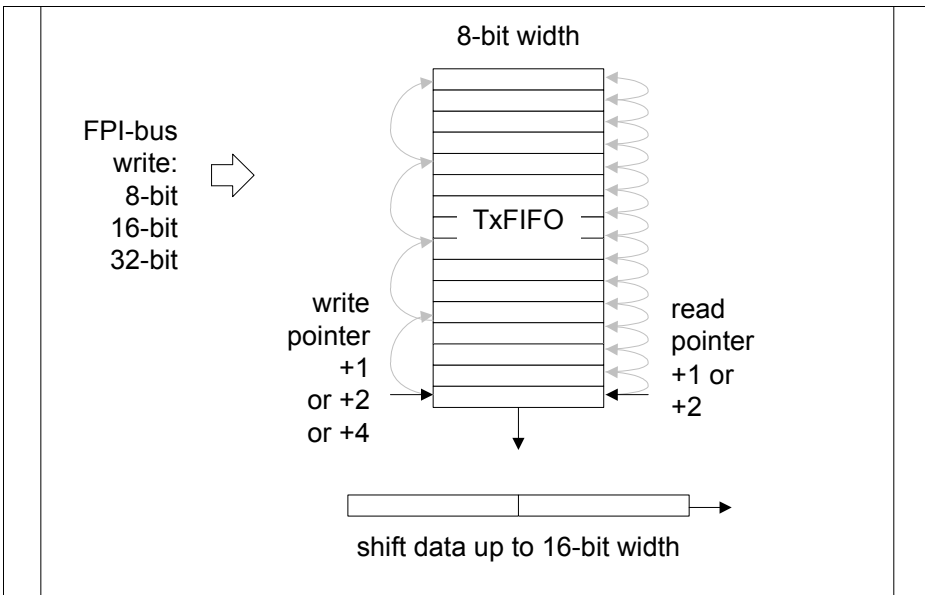


Figure 19-4 TxFIFO Overview

The number of bytes written to the TxFIFO is always defined with the **TXFIFOCON.INW**, does not depend on the SPB write width, and should be equal to it.

If the SPB write width is shorter than INW, then the missing part is padded with zeros.

If the SPB write width is longer than INW, then the excessive part is lost.

1) FPI is the protocol used on the System Peripheral Bus (SPB).

Asynchronous/Synchronous Interface (ASCLIN)

On the shift register side, the TxFIFO is always emptied by a number of bytes as needed for the data width field **DATCON.DATLEN**.

The **Table 19-1** describes all the possible write (fill) combinations. The bytes which will be filled into the TXFIFO are labeled as A, B, C, D and 0.

Table 19-1 Inlet Width versus SPB Write Width

TxFIFO Contents		SPB Bus Write Width		
		8-Bit A	16-Bit BA	32-Bit DCBA
Inlet Width INW	8-Bit	A	A	A
	16-Bit	0A	BA	BA
	32-Bit	000A	00BA	DCBA

19.4.2 Using the TxFIFO

The Tx FIFO has the ability to pack 16-bit writes from the FPI bus to two up-to-8-bit frames or one up-to-16-bit frame.

Note: Some (but not all) use-cases are described in the sections below. The software can always access the TXFIFO 8, 16 or 32 bit wide, according to its requirements.

19.4.2.1 Standard ASC Mode

The most standard usage of the ASC module is to transmit 7 or 8-bit values, and to fill the FIFO with 8-bit wide FPI-bus accesses. The FPI bus access is 8-bit wide, and there is only BS0 (Byte Select 0) signal active.

The transmit data is written to the address TXDATA. The Tx_Inlet_Width is 8-bit, and the Tx_Outlet_Width is 8-bit.

Asynchronous/Synchronous Interface (ASCLIN)

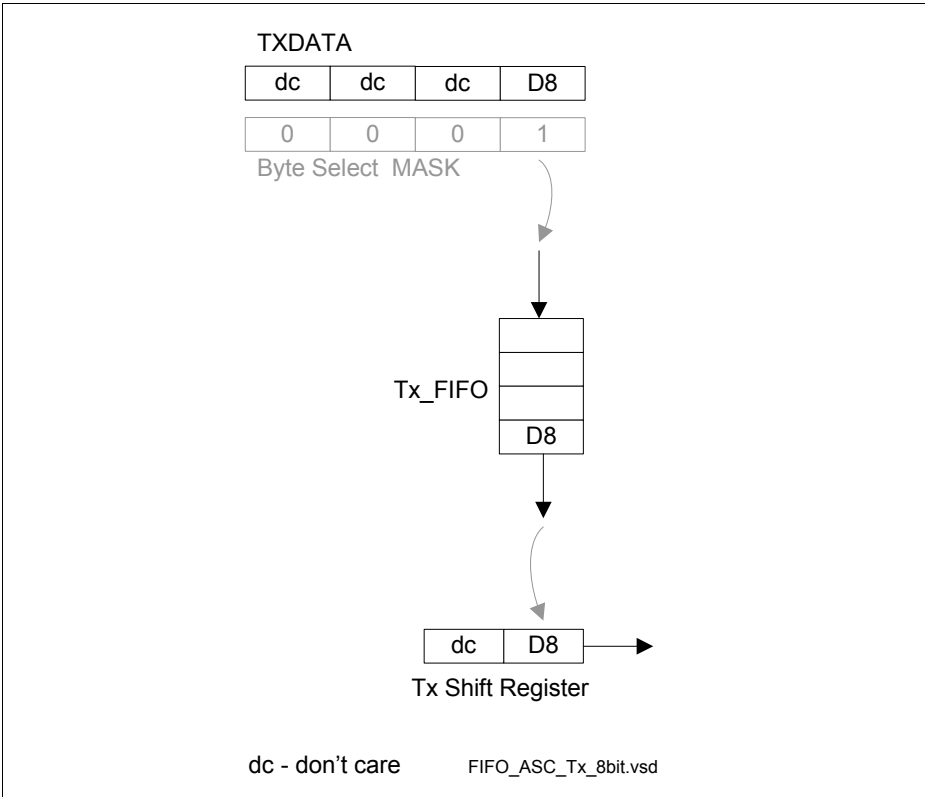


Figure 19-5 FIFO Operation in 7- or 8-Bit ASC Mode, with or without Parity

A less common mode of operation of the ASC module is to transmit 9-bit values, containing either 9-bit data or 8-bit data. The FPI bus access should be 16-bit wide, and there are BS0 and BS1 (Byte Select 0 and 1) signals active. The shift register is filled with two 8-bit values at the location read pointer and read pointer + 1.

The transmit data is written to the address TXDATA. The Tx_Inlet_Width is 16-bit, and the Tx_Outlet_Width is 16-bit.

Asynchronous/Synchronous Interface (ASCLIN)

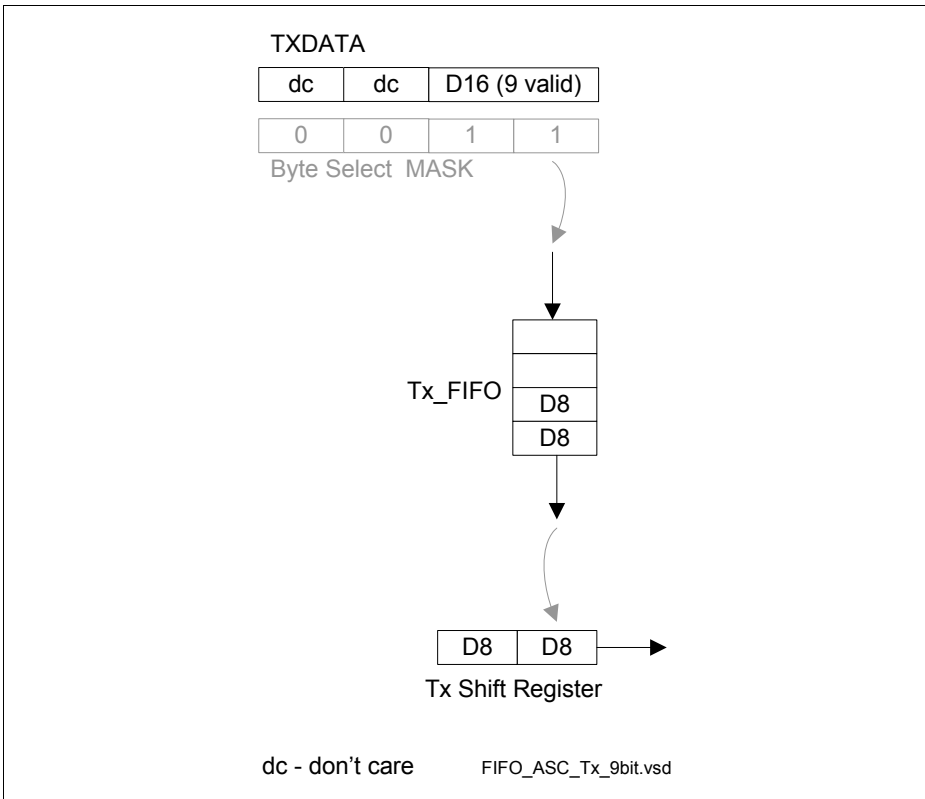


Figure 19-6 FIFO Operation in 9-Bit ASC Mode

19.4.2.2 High Speed ASC Mode

In order to reduce the FPI bus load by not using a 32-bit move for 7 or 8-bit values, one option is to fill the FIFO with 32-bit wide accesses containing four 8-bit wide values. All Byte Select signals BS[3:0] are active, and all four bytes are written in the TX FIFO in one cycle. At the same time, the write pointer of the TXFIFO jumps by the amount of four. The transmit data is written to the address TXDATA. The Tx_Inlet_Width is 32-bit, and the Tx_Outlet_Width is 8-bit.

Asynchronous/Synchronous Interface (ASCLIN)

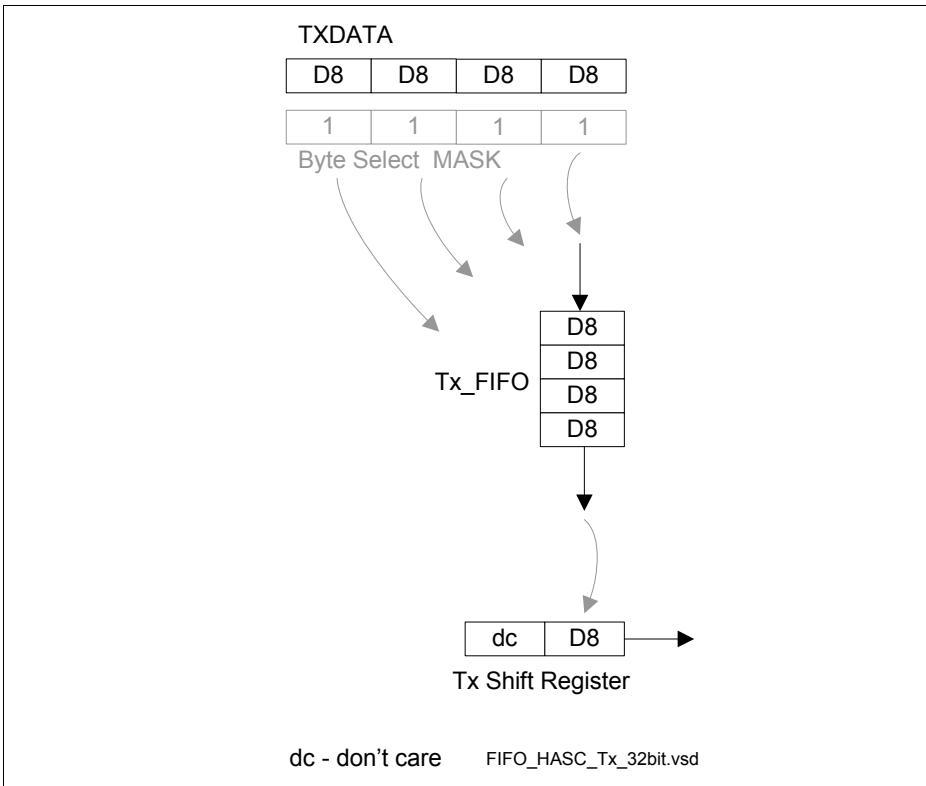


Figure 19-7 High-Speed Communication Scenario

Asynchronous/Synchronous Interface (ASCLIN)

19.4.2.3 LIN Mode

In LIN mode, several 8-bit frames are preceded by a low break pulse.

The generation of the break pulse is programmable with an 8 bit timer in units of bits, where the wide range of break pulses can be generated, for example in a range between 13 and 26 bits and beyond (up to 256 bits, but these lengths would violate the maximum LIN header length). The detection of the break pulse is programmable with an 8 bit timer in units of bits. Among others, the standard thresholds of 10 and 11 bit times can be set. .

The pulse is generated by an module internal timer. The transmit sequence consists of filling the stopped TXFIFO with the appropriate bytes, and then starting the break pulse, which activates the TxFIFO.

The transmit data is written to the address TXDDATA. The Tx_Inlet_Width can be 8-bit (or 16-bit, or 32-bit), and the Tx_Outlet_Width is 8-bit.

19.4.2.4 SPI Mode

SPI mode is used most often to send or receive data of 8 or 16-bits length, or some length in between. Therefore the reading out of the Tx FIFO must be 8 or 16-bit wide, and write could be 16-bit wide, or even 32-bit wide if two frames are packed in one FPI bus access.

The transmit data is written to the address TXDATA. The Tx_Inlet_Width is up to 32-bit, and the Tx_Outlet_Width is up to 16-bit.

Asynchronous/Synchronous Interface (ASCLIN)

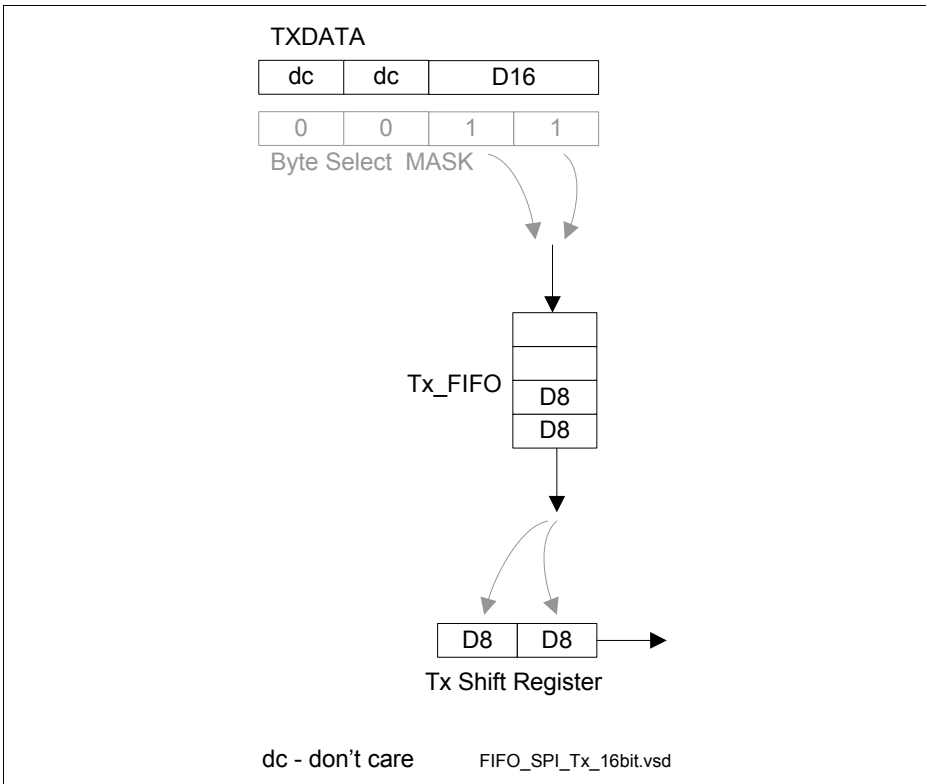


Figure 19-8 TXFIFO Operation in SPI Mode

19.4.3 RxFIFO Overview

The Rx FIFO has the ability to pack two up-to-8-bit frames or one up-to-16-bit frame to one 16-bit write to the FPI bus. It also has the ability to pack four up-to-8-bit 2-up-to-16-bit frames to one 32-bit write to the FPI bus.

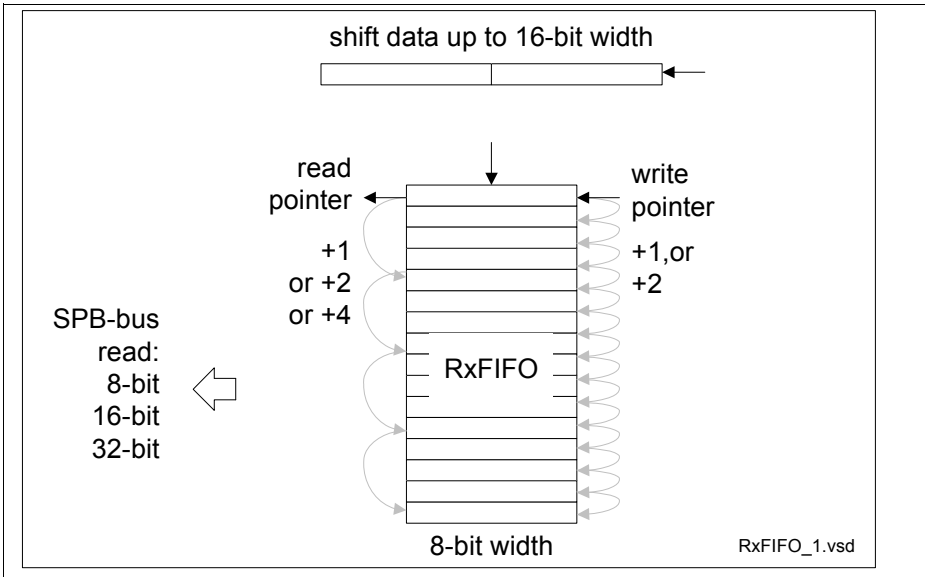


Figure 19-9 RxFIFO Overview

The number of bytes taken out of the RxFIFO is always **RXFIFOCON.OUTW**, does not depend on the SPB bus read width, which should be normally equal to OUTW.

If the SPB read width is shorter than the OUTW, then the missing part is filled with zeros.

If the SPB read width is longer than OUTW width, then the excessive part is lost.

On the shift register side, the RxFIFO is always filled with a number of bytes as needed for the data width field **DATCON.DATLEN**.

The **Table 19-2** describes all the possible combinations. The bytes in the RxFIFO are named A, B, C and D, A being the first one to take out. The padding byte is 0.

*Note: In case of an underflow, i.e. when a read access is performed to RXDATA and the number of bytes to be taken out of the FIFO as given by the setting of **RXFIFOCON.OUTW** exceeds the number of bytes that are actually stored in the FIFO, as indicated by **RXFIFOCON.FILL**, then the RxFIFO delivers the available data padded with zeros up to the read access width. However, the data remain in the FIFO and the filling level **RXFIFOCON.FILL** is not changed. To remove the data, the user software must flush the RxFIFO. The bit **FLAGS.RFU** is set.*

Asynchronous/Synchronous Interface (ASCLIN)

Table 19-2 Outlet Width versus SPB Read Width

SPB-Read Data, taken from the RxFIFO		SPB Bus Read Width		
		8-Bit	16-Bit	32-Bit
Outlet Width OUTW	8-Bit	A	A	A
	16-Bit	0A	BA	BA
	32-Bit	000A	00BA	DCBA

Asynchronous/Synchronous Interface (ASCLIN)

19.4.4 Using the Rx FIFO

The Rx FIFO has the ability to pack two up-to-8-bit frames or one up-to-16-bit frame to one 16-bit write to the FPI bus.

19.4.4.1 Standard ASC Mode

The received data is read from the address RXDATA.

In case of data width of 7 or 8 bits, one read on this address delivers one byte and empties the FIFO for one element, if the Rx_Outlet_Width is 8-bit. The Rx_Inlet_Width is 8-bit.

In case of data width of 9 bits, one read on this address delivers two byte and empties the FIFO for two elements, if the Rx_Outlet_Width is 16-bit. The Rx_Inlet_Width is 16-bit.

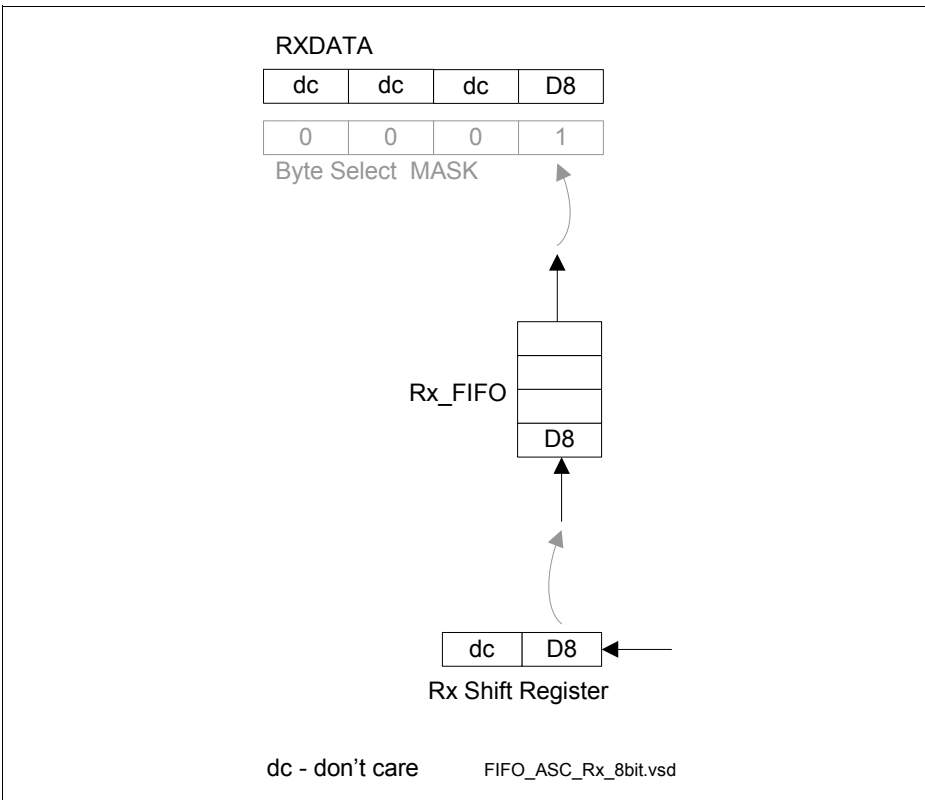


Figure 19-10 8-Bit ASC Reception

Asynchronous/Synchronous Interface (ASCLIN)

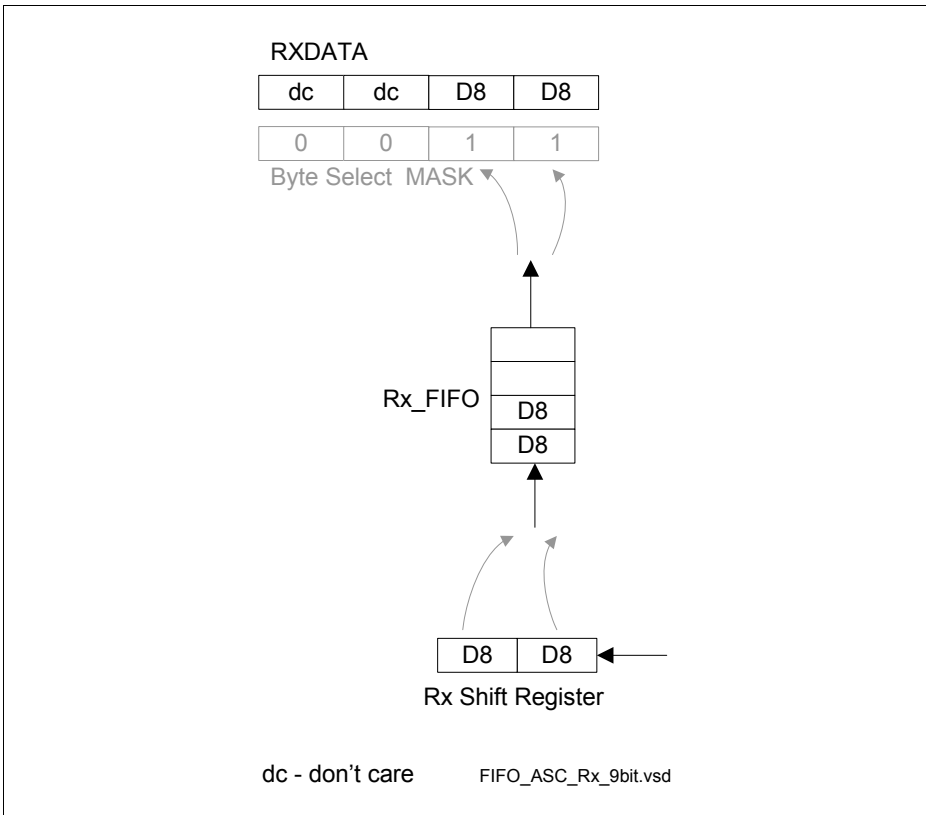


Figure 19-11 9-Bit ASC Reception

Asynchronous/Synchronous Interface (ASCLIN)

19.4.4.2 High Speed ASC Mode

The received data is read from the address RXDATA.

In case of data width of 7 or 8 bits, one read on this address can deliver (one or two or) four bytes and empties the FIFO by four elements, if the Rx_Outlet_Width is 32-bit. The Rx_Inlet_Width is 8-bit.

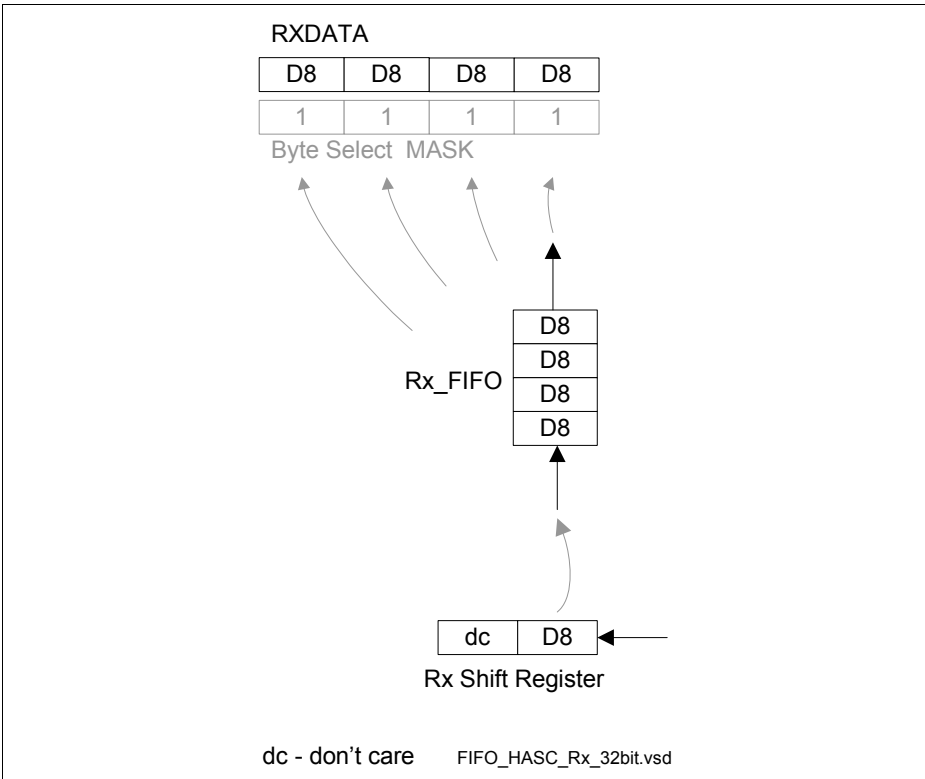


Figure 19-12 High Speed ASC Communication

19.4.4.3 LIN Mode

The received data is read from the address RXDATA.

The read data width is 8 bits (or 16 bit or 32 bit), one read on this address delivers one byte and empties the FIFO by one element, if the Rx_Outlet_Width is 8-bit. The Rx_Inlet_Width is 8-bit

Asynchronous/Synchronous Interface (ASCLIN)

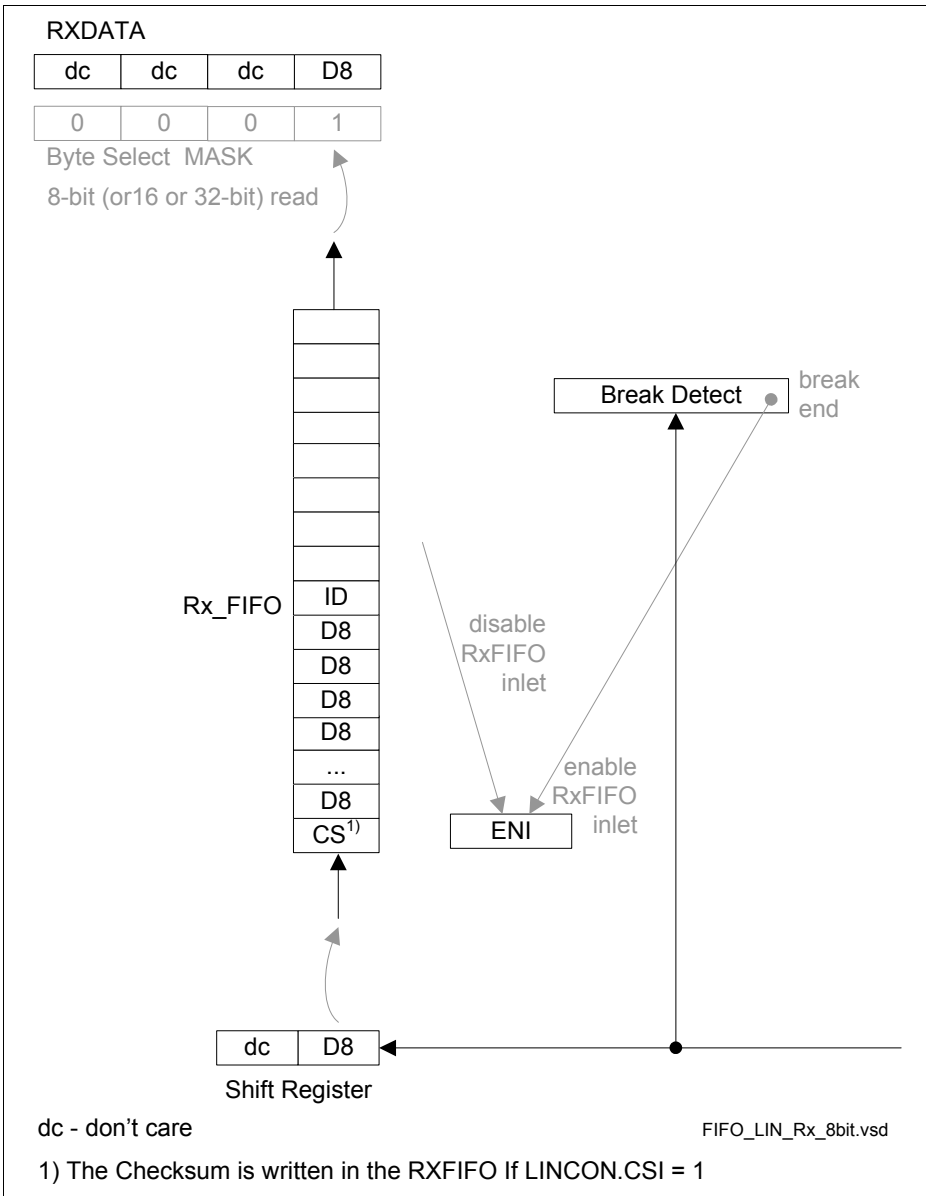


Figure 19-13 RXFIFO Operation in LIN Mode

Asynchronous/Synchronous Interface (ASCLIN)

19.4.4.4 SPI Mode

The received data is read from the address RXDATA.

In case of data width of 16 bits, one read on this address delivers two bytes and empties the FIFO by two elements, if the Rx_Outlet_Width is 16-bit. The Rx_Inlet_Width is 16-bit

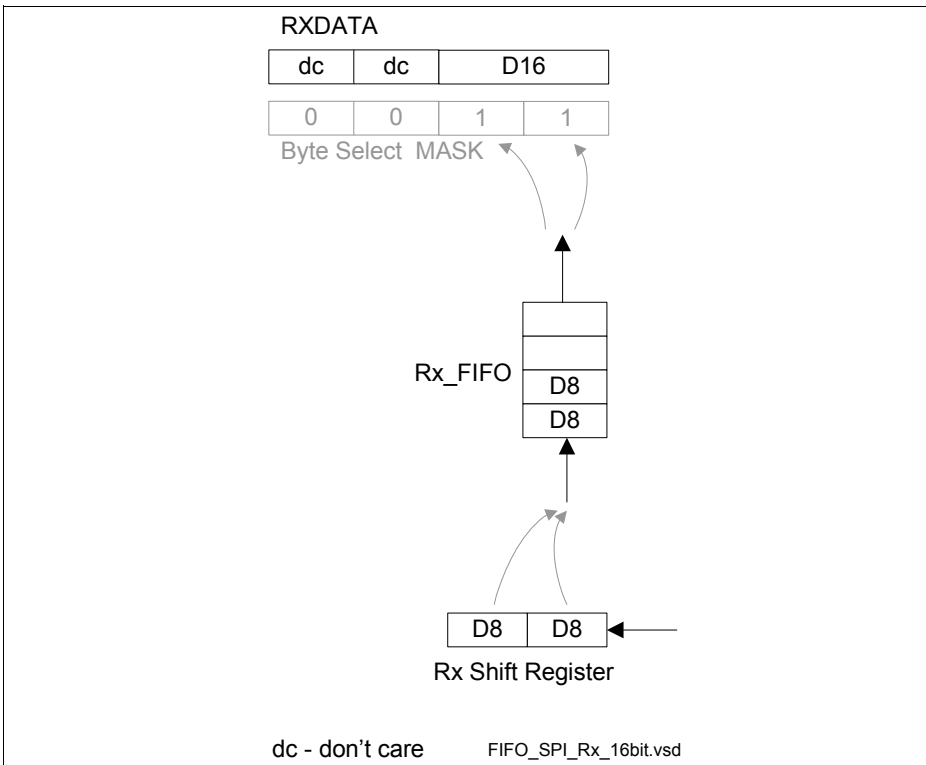


Figure 19-14 RXFIFO Operation in SPI Mode

19.4.5 RTS / CTS Handshaking

A receiver deactivates the RTS (Request to Send) output signal when the RXFIFO is almost full, in order to avoid its overload. When the emptying of the RXFIFO starts, for example by a DMA, and the filling level falls below the threshold level, the RTS output is activated again. The threshold level is fixed to RXFIFO size minus four, in order to support byte, two bytes and four bytes DMA transfers.

The transmitter receives the RTS output of the receiver on its CTS input and accordingly pauses and resumes the transmission.

Asynchronous/Synchronous Interface (ASCLIN)

19.5 Clock System

The clock system generates all the clocks needed for the proper operation of the ASCLIN module: digital filter clock, oversampling clock, bit time and serial SPI clock.

The clock used for the clock system f_A is independent from the SPB bus clock and remains constant if the SPB bus clock changes, for example in power saving scenarios. The bit field **CSR.CLKSEL** selects the clock source for the f_A frequency, which can be synchronous or asynchronous, higher or lower than the SPB bus frequency.

19.5.1 Baud Rate Generation

Fractional Divider, n-divider and oversampling divider with configurable sample point.

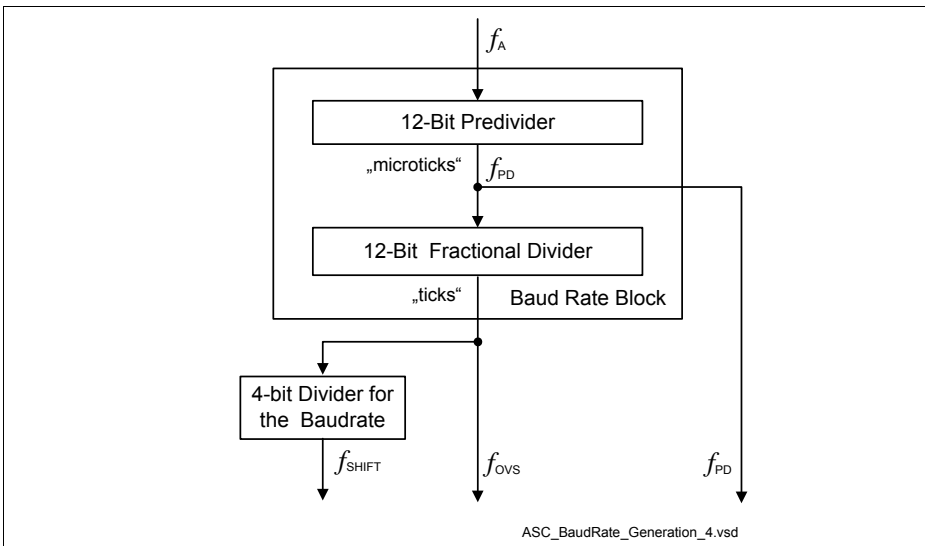


Figure 19-15 Baud Rate Generation

The following bit fields are available for configuring of the 28-bit baud rate divider chain:

- **BITCON.PRESCALER** - the division ratio of the predivider
- **BRG.NUMERATOR** - the nominator of the fractional divider
- **BRG.DENOMINATOR** - the denominator of the fractional divider
- **BITCON.OVERSAMPLING** - the division ratio of the baudrate post divider

The chain of the generated frequencies from f_A down to the f_{SHIFT} (the baudrate) is calculated as follows:

$$f_{PD} = f_A / (\mathbf{BITCON.PRESCALER} + 1)$$

$$f_{OVS} = f_{PD} * \mathbf{BRG.NUMERATOR} / \mathbf{BRG.DENOMINATOR},$$

Asynchronous/Synchronous Interface (ASCLIN)

$$f_{\text{SHIFT}} = f_{\text{OVS}} / (\text{BITCON.OVERSAMPLING} + 1)$$

*Note: Fractional division requires that **BRG.NUMERATOR** is less or equal than the **BRG.DENOMINATOR**.*

The overall formula is given as follows:

(19.1)

$$\text{BAUDRATE} = \frac{f_A \times \text{NUMERATOR}}{(\text{PRESCALER} + 1) \times \text{DENOMINATOR} \times (\text{OVERSAMPLING} + 1)}$$

19.5.2 Bit Timing Properties

The ASCLIN module provides flexible programming of the bit oversampling, sampling point and input signal filtering properties.

The oversampling factor for the incoming bit-stream is configurable from 4 to 16 ticks (or time quanta) per bit.

At the same time, using the oversampling frequency, a digital median filter can be enabled to filter the incoming bit stream. The filtering uses the standard majority out of three procedure. If the filter is disabled, then each bit is sampled only once.

The sampling point is also configurable and should be used in conjunction with the oversampling factor. One standard setting is 16x oversampling and using the samples 7, 8 and 9 for the data. Another possible setting could be 8x oversampling and using the samples 3, 4, and 5 for data.

The following bitfields are available for configuring the bit properties:

- **BITCON.PRESCALER** - the twelve bit integer divider defining the microtick used by the fractional divider to generate the baud rate, and by the digital filter for the deglitching of the RX input signal.
- **BITCON.SAMPLEPOINT** - the bit field defining the sampling point position, and the duty cycle in the SPI mode.
- **BITCON.SM** - the bit enables the digital median filter (majority out of three): 1 or 3 samples per bit.
- **IOCR.DEPTH** - the bitfield defining the floating average filter depth: off or 1 to 63 microticks
- **BITCON.OVERSAMPLING** - the bitfield defining the number of ticks per bit, in the range of 4 to 16. This is a post-divider located after the fractional divider.

Asynchronous/Synchronous Interface (ASCLIN)

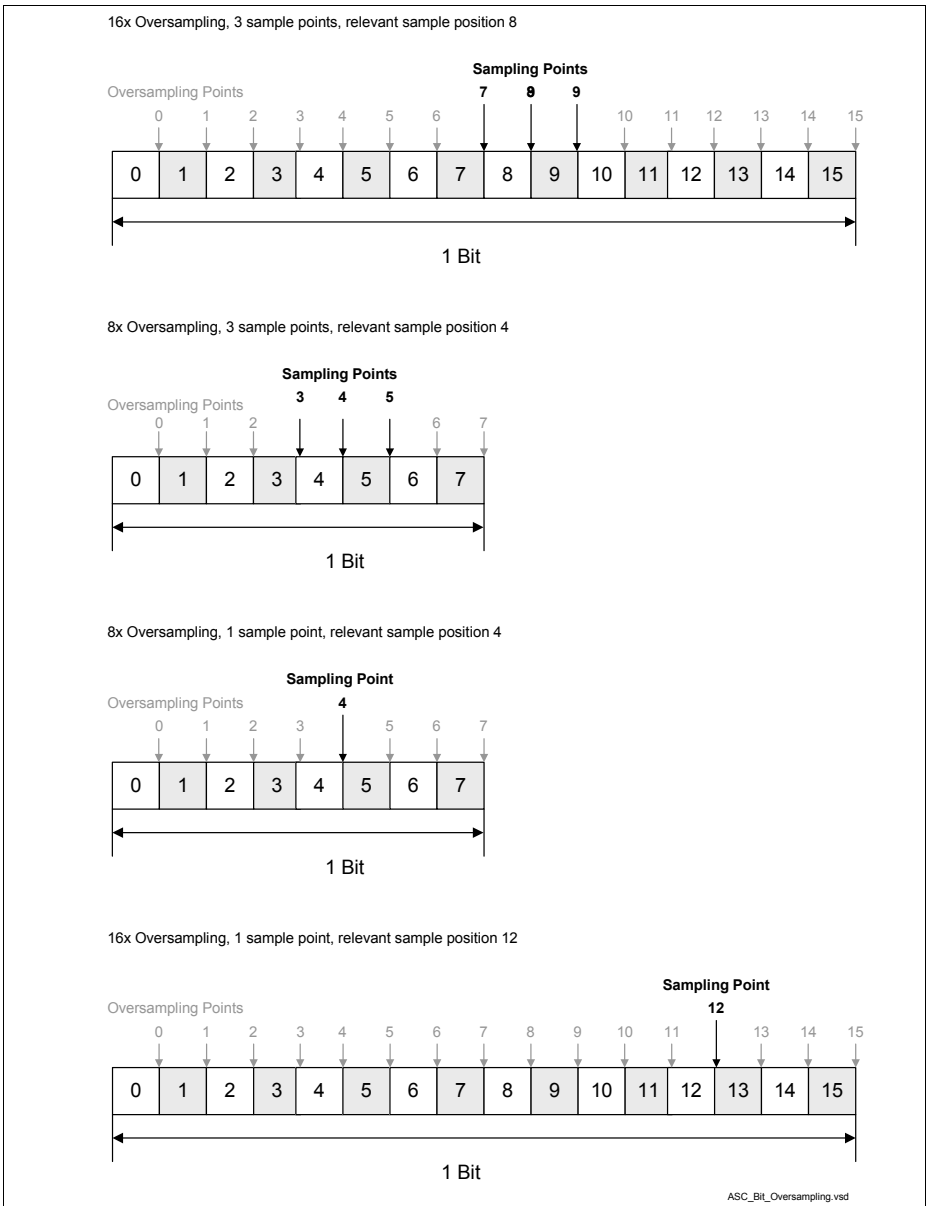


Figure 19-16 ASCLIN Bit Structure

Asynchronous/Synchronous Interface (ASCLIN)

Generally, the sampling point should be placed in the middle of the bit. This position is optimal in case the baud rate (the oscillator frequency) is not very precise and stable.

If the oscillator precision is very high, which is usually the case when two microcontrollers driven by quartz oscillators communicate, but the signal edges are very unsymmetrical, which is the case if open drain half-duplex connection is used, it can be of advantage to move the sample point somewhere in the second half of the bit. Open drain connection usually causes the “0” bits to be longer than the “1” bits. In such a case, optimizing the sampling point would mean placing it in the middle of the shorter “1” bit.

At the end, different combinations of oscillator precision, asymmetry of the edges, and loop-delays for collision detection result in different optimal positions of the sampling point.

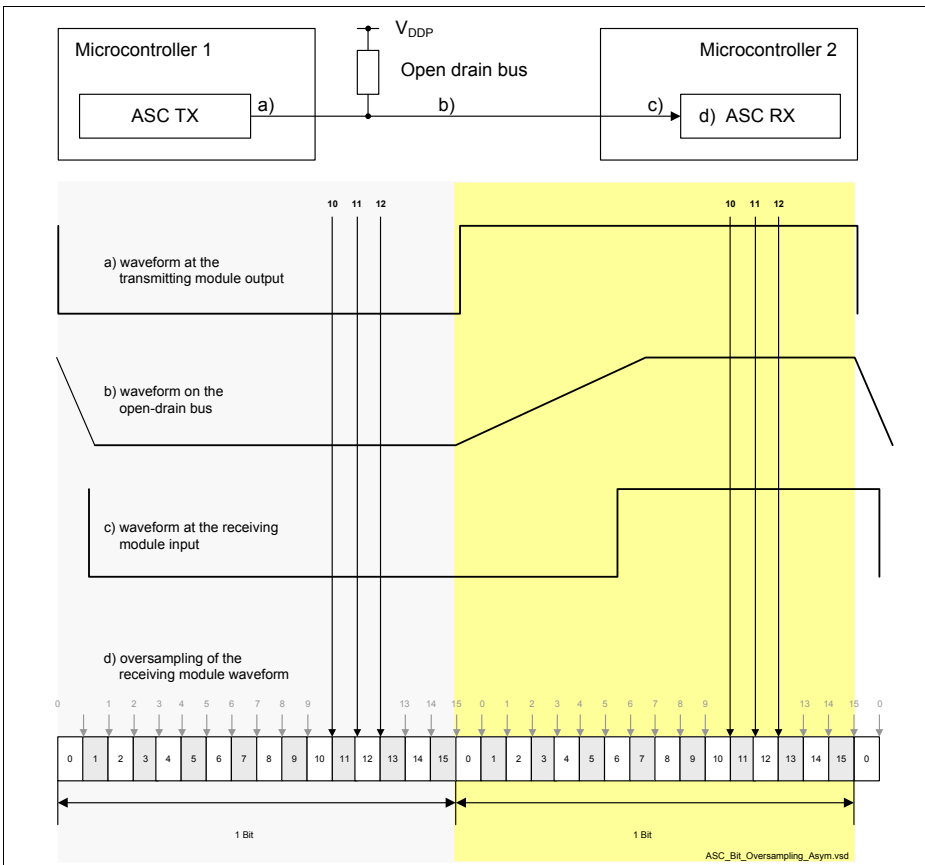


Figure 19-17 Bit Length Asymmetry and the Sampling Point

Asynchronous/Synchronous Interface (ASCLIN)

19.6 Data Frame Configuration

Applicable as transmitter and receiver, the parity scheme is configured in the bit fields **FRAMECON**.ODD, the data length in the bit fields **DATCON**.DATLEN and the stop bits in the bit fields **FRAMECON**.STOP.

19.7 Miscellaneous Configuration

Loop - back

Asynchronous/Synchronous Interface (ASCLIN)

19.8 Synchronous Mode

In synchronous mode the module supports the SPI setting of shift edge first, than the latch edge, see [Figure 19-18](#). The module is set in synchronous mode by using the bit field [FRAMECON.MODE](#).

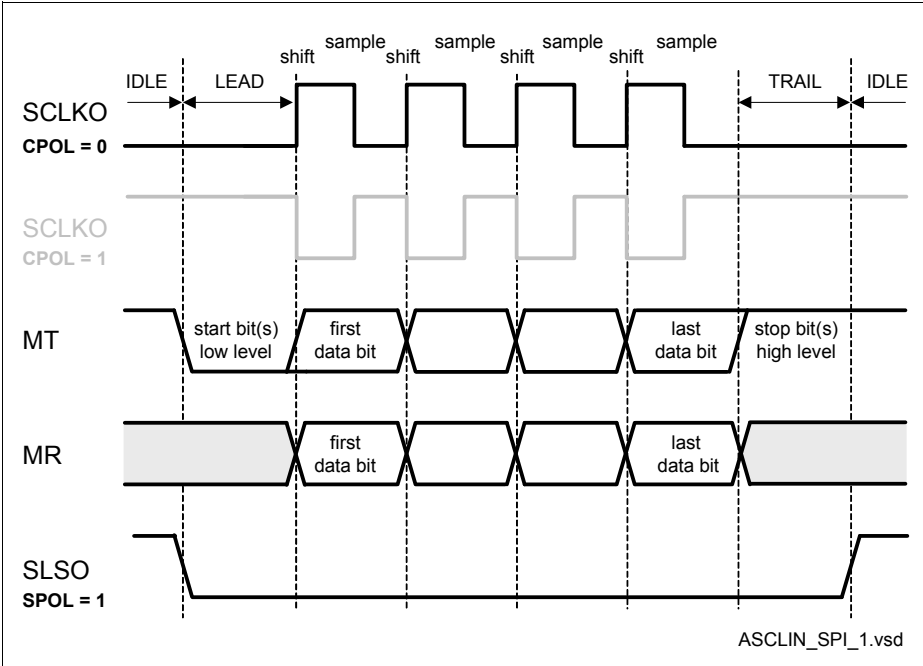


Figure 19-18 SPI Timing

19.8.1 Baud Rate and Clock Generation

The baud rate and clock generation in the synchronous mode uses generally the same counters and principles as in the asynchronous mode. It uses the same prescaler, fractional divider, and oversampling divider with configurable sample point. The only difference is that the shift clock is driven as output signal at the pin SCLKO. If a symmetrical shift clock is required, the oversampling ratio should be an even number, and the sampling point should be set to one half of the oversampling ratio (see the [BITCON](#) register).

The clock polarity is configured in the bit [IOCR.CPOL](#).

Asynchronous/Synchronous Interface (ASCLIN)

19.8.2 Data Frame Configuration

The leading and trailing delays are configured in the bit fields **FRAMECON.LEAD** and **FRAMECON.STOP**. The IDLE phase duration is configured using the bit field **FRAMECON.IDLE**. The data length of 2 to 16 bit is defined in the bit field **DATCON.DATLEN**.

19.8.3 Slave Selects Configuration

The SPI master activates automatically the slave select output signal for each data word. The polarity of the slave select can be configured by using **IOCR.SPOL**.

19.8.4 Miscellaneous Configuration

Loop - back

Asynchronous/Synchronous Interface (ASCLIN)**19.9 LIN Support**

The ASCLIN module provides hardware support for the LIN protocol.

It supports all four elementary LIN transactions:

- TxH - Transmission of Header
- TxR - Transmission of Response
- RxH - Reception of Header
- RxR - Reception of Response

By supporting additionally the combinations of these elementary transactions, the module actually supports all LIN use cases: sending and receiving headers and responses as

- LIN master or
- LIN slave

A LIN master is engaged in three elementary transactions: TxH, TxR, RxR. It never engages in RxH, because a master never receives a header, it only transmits a header.

A LIN slave engages also in three elementary transactions: RxH, TxR, RxR. It never engages in TxH, because a slave never transmits a header, it only receives a header.

Each elementary transaction needs some hardware resources in order to be completed with minimum CPU intervention. Here is a list of tasks per transaction, supported by hardware, and the required hardware resources:

- TxH - Transmission of Header - master mode only
 - break generation: 8-bit bit-field defining the break length in units of bits
 - sync-field generation: hard coded 55H byte
 - ID transmission with interrupt generation
- TxR - Transmission of Response - master and slave mode
 - number of bytes parameter: bit-field of length 4
 - checksum generation: hardware engine, supporting classic and enhanced checksum, which can be enabled or disabled
- RxH - Reception of Header - slave mode only
 - optional auto-baud detection: fractional divider with programmable nominator and denominator
 - number of bytes parameter: bit-field of length 4
 - checksum detection: hardware that supports classic and enhanced checksum, which can be enabled or disabled; a checksum error is flagged, and an error interrupt can be triggered, if enabled
 - interrupt at end of header (necessary to set the number of bytes for the RxR phase)
 - timeout on overflow: 8-bit timer
 - break detection: 8 bit timer with programmable threshold in units of bits
- RxR - Reception of Response - master and slave mode
 - number of bytes parameter: bit-field of length 4

Asynchronous/Synchronous Interface (ASCLIN)

- checksum: the received checksum is optionally delivered in the RXFIFO
- timeout on overflow: 8-bit timer

The break detection feature is always active.

Wake-up signal generation in both slave and master mode

For many LIN configuration parameters, the hardware of the ASCLIN module provides wider ranges than the LIN standard parameters. Therefore the application software shall take care that appropriate LIN standard values are used to configure the module. Such configuration parameters are:

- break length
- data width
- break threshold
- wake-up length
- header, frame and response timeout
- idle time

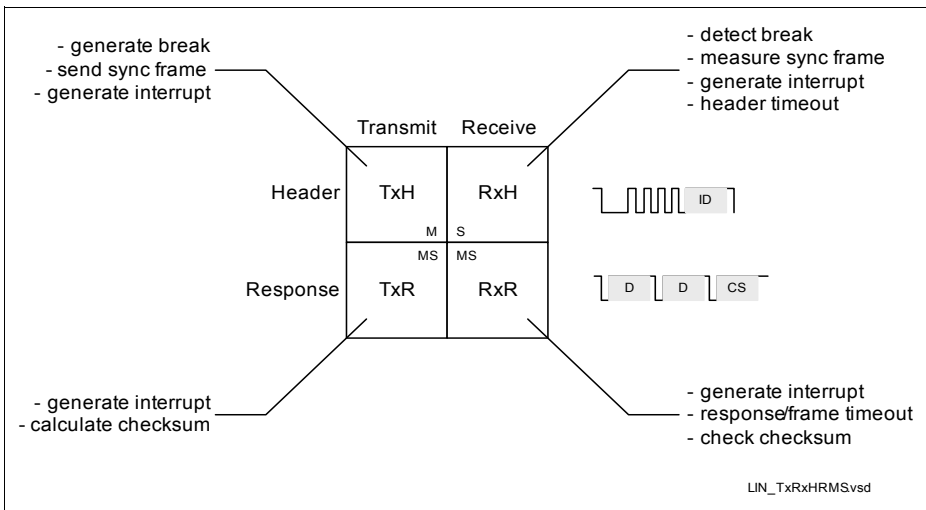


Figure 19-19 Overview of the elementary LIN transactions

Asynchronous/Synchronous Interface (ASCLIN)

19.9.1 LIN Watchdog

The LIN watchdog monitors the duration of the header, the frame or the response, and appearance of break pulses. It checks against the pre-defined time limits. If the limits are violated, timeout interrupts are generated.

The LIN watchdog is necessary in slave mode.

The wake pulse generation is performed using the shift register and an appropriate data byte containing several consecutive zero bits. The wake pulse detection is done using falling edge detection.

For monitoring the bus for long idle or zero states flags for rising and falling edge are available. These flags can be polled with some appropriate time raster (in microseconds or milliseconds range).

The header timeout value is known at module initialization time and remains constant for all frames.

The frame or response timeout value depends on the length of the response (1 to 8 bytes) and must be set by software depending on the received ID, after the header has been received. The initial value of this timeout should be set by software to the maximum allowed by the timer, that is 256.

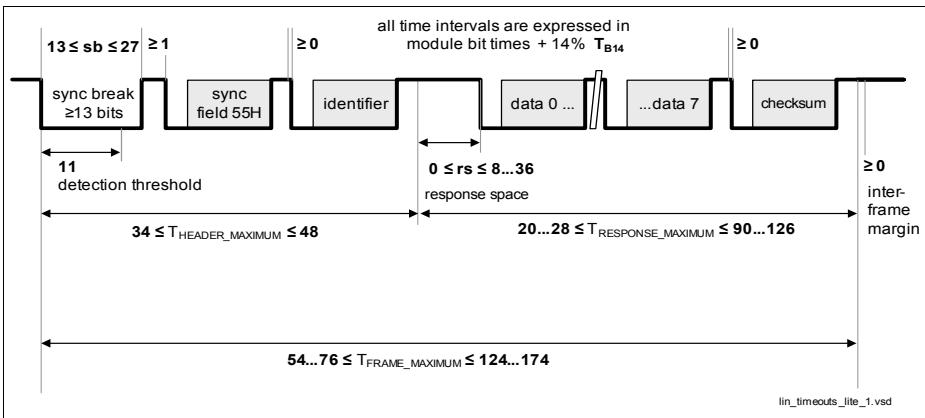


Figure 19-20 Duration of the Elements of a LIN Frame

The module provides timer blocks running in parallel, performing watchdog functions on specific timing requirements of the LIN protocol.

Asynchronous/Synchronous Interface (ASCLIN)

19.9.1.1 LIN Break, Wake, Stuck Handling

This subsection describes:

- the monitoring of the LIN Bus
- Break Pulse detection and generation and
- Wake Pulse detection and generation.

Monitoring the Bus

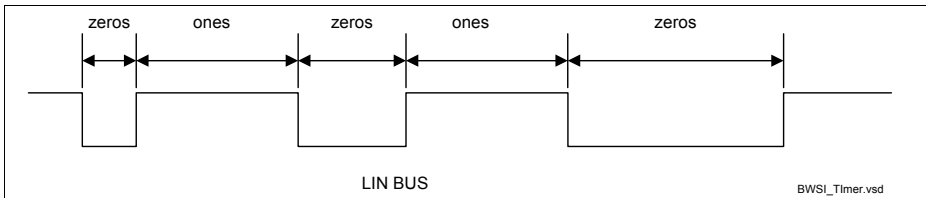


Figure 19-21 LIN Bus View as Sequence of Sequences of Zeros and Ones

Monitoring the bus for long idle periods (or stuck periods) is supported by providing the bit fields **FLAGS.RED** and **FLAGS.FED**. Polling these fields periodically within some time raster defined by the operating system or timer for example each 1ms or each 10 ms can be used for detecting very long inactive periods in the range, for example, of 150 ms to 10s. Additionally, an interrupt on edge can be enabled by using the corresponding enable bits in the **FLAGSENABLE** registers. Setting and clearing the flags can be done by software by using the **FLAGSET** and **FLAGSCLEAR** register.

Note: The low break or wake up pulses do not cause a dummy byte to be transferred to the receive FIFO and do not cause a frame violation error interrupt. After the pulse the shift register goes into initial state and waits for the falling edge of the start bit of the next frame.

Break and Wake Pulse Detection

The detection of the break pulse, which is defined as low pulse with a minimum duration of 13 bit times, is performed using a threshold of either 10 or 11 bit times. The break detection threshold is set using the bitfield **LINBTIMER.BREAK**.

Detecting a break pulse anywhere in the frame resets the LIN state machine to the "Break Delimiter Detected" state BDD and the whole sequence inclusive the watchdog timeout counting starts from the beginning .

Note: Detecting a break pulse anywhere in the frame could result in setting the Frame Error Flag (FE)

Wake low-pulse detection is done by monitoring the bus for a falling edge in sleep mode. If an early wake-up shall be suppressed, the parameter **IOCR.DEPTH** (gliche filter) may be used.

Asynchronous/Synchronous Interface (ASCLIN)

Break and Wake Pulse Generation

The generated break low-pulse duration is defined by the bit field **LINBTIMER.BREAK** in the range of 1 to 64 bits.

Wake low-pulse is nominal 5 bit times long, but can be set to any value between 1 and 9, by writing the appropriate character in the TXFIFO and requesting its transmission as the wake pulse by setting **FLAGS.TWRQ**.

Note: Injecting the low pulse disables the reception with the shift register, so that this low pulse is not detected or treated as a normal ASC frame.

Asynchronous/Synchronous Interface (ASCLIN)

19.9.1.2 LIN Header and Response Timers

The LIN header and response times can be monitored separately.

(>>LINHTIMER, DATCON description)

- Header duration measurement
 - In master mode starting point of the time measurement is falling edge of the break pulse.
 - In slave mode starting point of the time measurement is the moment of detecting that there is a break pulse going on, that is the moment of detection of a low pulse longer than either 10 or 11 bit times, as configured in the LINBTIMER register.
- Response duration measurement
 - In both master and slave modes the response duration measurement starts with the end of the header and ends with the end of the response. “End” means end of the last bit of the last byte (of the checksum byte).

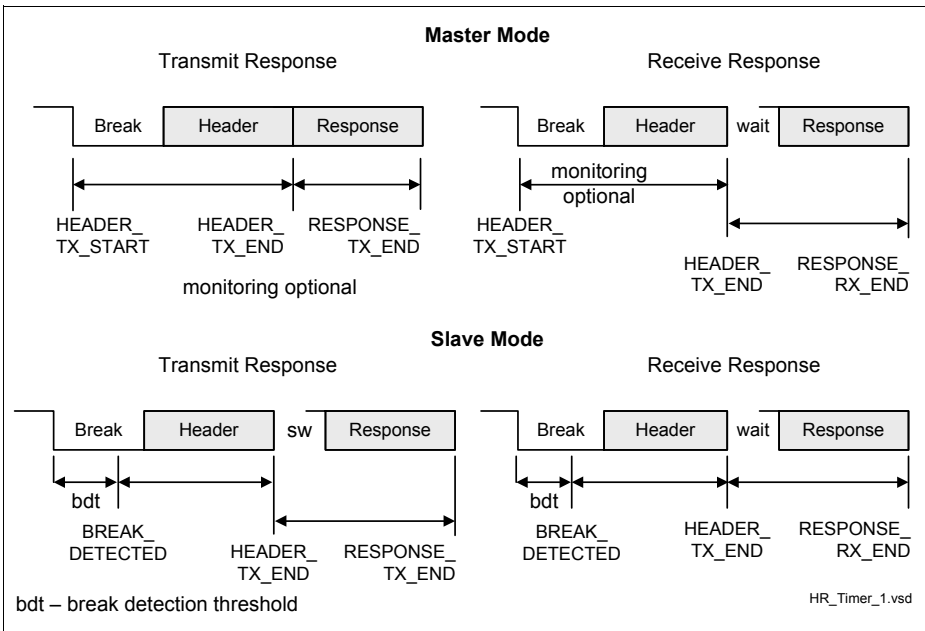


Figure 19-22 Duration Measurement of the Header and Response

Note: Transmitting master node can optionally monitor its own header or response timeouts, in order to detect some error conditions, like TXFIFO not containing the ID or data to be transmitted. However, these error conditions can be detected also in other ways.

Asynchronous/Synchronous Interface (ASCLIN)

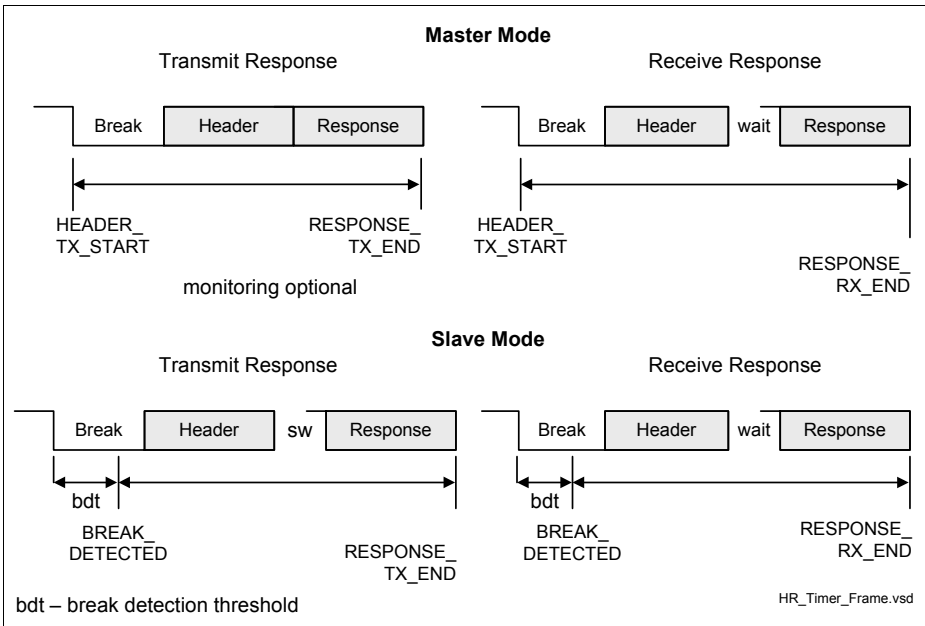


Figure 19-23 Duration Measurement of the Response or Frame

According to the setting of the bit `DATCON.RM`, the `DATCON.RESPONSE` bit field defines response or frame duration threshold.

Asynchronous/Synchronous Interface (ASCLIN)
19.9.2 LIN Master Sequences

The sequences described below are examples how the elementary LIN transactions can be executed. There exist other ways to do them, for example by using other events to control the protocol flow: fifo level events instead of end of header / response events. Some alternatives are indicated in the lists below with brackets.

Initialize the module in LIN master mode:

- deactivate the clock source : **CSR.CLKSEL** = 0 (off)
- wait or poll for **CSR.CON** = 0 (see also **Clock Reconfiguration**)
- enter the INIT mode : **FRAMECON.MODE** = INIT
- activate the clock source : set **CSR.CLKSEL**
- wait or poll for **CSR.CON** = 1
- deactivate the clock source : **CSR.CLKSEL** = 0 (off)
- wait or poll for **CSR.CON** = 0
- select the operation mode : **FRAMECON.MODE** = 3 (LIN)
- configure the master mode : **LINCON.MS** = 1
- configure the baud rate : **BRG**
- configure the bit timing : **BITCON.PRESCALER** and **OVERSAMPLING**
- configure the bit sampling : **IOCR.DEPTH**, **BITCON.SM**, **SAMPLEPOINT**
- configure the frame parameter : **FRAMECON.PEN**=0 (no parity), **STOP** = 1
- configure the break length : **LINBTIMER.BREAK** = 13 (typ)
- configure the delay parameter : **FRAMECON.LEAD**, **FRAMECON.IDLE**
- configure the checksum mode : **LINCON.CSEN**, **LINCON.CSI**
- activate the clock source : set **CSR.CLKSEL**
- wait or poll for **CSR.CON** = 1

Send only a header [master task]:

- configure the watchdog timer : **DATCON.RESPONSE** =256 (max)
- configure the RXFIFO : **RXFIFOCON.ENI** = 0 (or 1), **OUTW**, **BUF**=0, flush
- configure the TXFIFO : **TXFIFOCON.ENO** = 0, **INW**, flush
- clear the event flags : **FLAGSCLEAR**.**THC**, **TRC**, **RRC**, **CEC**, **FEC**, **HTC**, **RTC**, **LPC**, **LC**, **TFOC/LC**, **RFOC/UC/LC**
- enable the TX interrupt event : **FLAGSENABLE**.**THE**
- enable the EX interrupt events : **FLAGSENABLE**.**HTE**, **CEE**, (**FEE**, **LPE**)
- write into the TXFIFO : **TXDATA** = ID byte
- start the header transmission : **FLAGSET**.**THRQS** = 1
- configure the TXFIFO : **TXFIFOCON.ENO** = 1

Note: If the software needs the sent ID in the RXFIFO, it should add the setting of ENI=1 to the sequence above - before triggering the header transmission. In such a case

Asynchronous/Synchronous Interface (ASCLIN)

after the header has been transmitted the ID byte will be available in the RXFIFO, but not the sync byte.

React to the interrupt(s) indicating the end of transmission (and reception) of the ID byte:

- check the error flags : **FLAGS**.HT, CE (FE, LP)
- read from the RXFIFO : **RXDATA**=received ID, if no error has been detected

Check the transmitted / received ID to determine with which response sequence: transmit, receive or ignore to continue (same procedure as in the slave mode).

Send a response to the latest header [slave task]:

- configure the response count : **DATCON**.DATLEN
- configure the watchdog timer : **DATCON**.RESPONSE
- configure the checksum mode: **DATCON**.CSM
- configure the RXFIFO : **RXFIFOCN**.ENI = 0
- configure the TXFIFO : **TXFIFOCN**.ENO = 1
- clear the event flags : **FLAGSCLEAR**.TRC, TFOC/UC, CEC, RTC, BDC
- enable the TX interrupt event : **FLAGSENABLE**.TRE
- enable the EX interrupt events: **FLAGSENABLE**.TFOE, CEE (RTE), BDE
- write into the TXFIFO : **TXDATA** = data bytes D0, D1, ... (and optionally the checksum byte if the hardware checksum generation is disabled)
- start response transmission : **FLAGSET**.TRRQS = 1
- the corresponding interrupts signal an error or the end of the response transmission

Receive a response to the latest header [slave task]:

- configure the response count : **DATCON**.DATLEN
- configure the watchdog timer : **DATCON**.RESPONSE
- configure the checksum mode: **DATCON**.CSM
- configure the RXFIFO : **RXFIFOCN**.ENI = 1
- clear the event flags : **FLAGSCLEAR**.RRC, RFOC/UC, FEC, LCC, RTC, FEDC, REDC, BDC
- enable the RX interrupt event : **FLAGSENABLE**.RRE
- enable the EX interrupt events: **FLAGSENABLE**.RFOE/UE, FEE, (LCE), RTE, BDE
- The corresponding interrupts signal that the whole response has been received (= end of the frame) or an error has occurred.

Note: If checksum injection has been enabled, the received (custom) checksum is also written into the RXFIFO and should be taken into account.

- Fetch from the RXFIFO : **RXDATA** = received data, if no error occurred

*Note: The hardware checksum check can be enabled using the bit **LINCON**.CSEN. The received checksum write to the RXFIFO can be enabled using the **LINCON**.CSI.*

Asynchronous/Synchronous Interface (ASCLIN)
Ignore the latest header [slave task]:

- configure the RXFIFO : **RXFIFOCON**.ENI = 0
- set the header only mode : **DATCON**.HO = 1
- clear the event flags : **FLAGSCLEAR**.FEDC, REDC, BDC

The LIN master always knows if the header is followed by a response transmission (by himself) or a response reception (by a slave) or the response transmission is from slave to slave, except in case of a slave that does not respond always, but driven by internal events.

In case of addressing an always responding slave, it is recommended first to configure the header and response (transmission or reception), and afterwards to start the header transmission and optionally, at the same time, the response transmission.

To start a transaction consisting of sending a header and sending or receiving a response, which makes one whole LIN frame [master node]:

- configure the response count : **DATCON**.DATLEN
- configure the watchdog timer : **DATCON**.RESPONSE
- configure the checksum mode : **DATCON**.CSM
- configure the RXFIFO : **RXFIFOCON**.BUF = 0, flush, ENI = 0
- configure the TXFIFO : **TXFIFOCON** flush, ENO = 0
- clear the interrupt event flags : **FLAGSCLEAR**.THC, TRC, RRC, CEC, FEC, HTC, RTC, LPC, LCC, TFOC/UC/LC, RFOC/UC/LC, FEDC, REDC, BDC
- send case
 - enable the TX interrupt event : **FLAGSENABLE**.TRE, (THE, HTE)
 - enable the EX interrupt events: **FLAGSENABLE**.TFOE, CEE, (FEE, RTE, LPE, BDE)
- receive case
 - enable the RX interrupt event: **FLAGSENABLE**.RRE. (THE, HTE)
 - enable the EX interrupt events: **FLAGSENABLE**.RFOE/UEE, CEE, FEE, RTE, (LCE, BDE)
- write into the TXFIFO : **TXDATA** = ID byte
- send case
 - write to TXFIFO : **TXDATA** = data bytes (and optionally checksum byte if the hardware checksum is disabled)
- start the header transmission : **FLAGSSET**.THRQS = 1
- configure the TXFIFO : **TXFIFOCON**.ENO = 1

Note: If the software needs the sent ID in the RXFIFO, it should add the setting of ENI=1 to the sequence above - before triggering the header transmission. In such a case after the header has been transmitted the ID byte will be available in the RXFIFO, but not the sync byte.

- send case
 - start transmit response : also set **FLAGSSET**.TRRQS = 1

Asynchronous/Synchronous Interface (ASCLIN)

- the corresponding interrupt signals an error or the end of the response transmission
- read from the RXFIFO : if no error detected, **RXDATA** = received ID
- receive case
 - fetch from the RXFIFO : if no error has been detected, **RXDATA** = received data bytes and optionally the checksum byte

Asynchronous/Synchronous Interface (ASCLIN)
19.9.3 LIN Slave Sequences

The sequences described below are examples how the elementary LIN transactions can be executed. There exist other ways to do them, for example by using other events to control the protocol flow: fifo level events instead of end of header / response events. Some alternatives are indicated in the lists below with brackets.

Initialize the module in slave mode:

- deactivate the clock source : **CSR.CLKSEL** = 0 (off)
- wait or poll for **CSR.CON** = 0 (see also **Clock Reconfiguration**)
- enter the INIT mode : **FRAMECON.MODE** = INIT
- activate the clock source : set **CSR.CLKSEL**
- wait or poll for **CSR.CON** = 1
- deactivate the clock source : **CSR.CLKSEL** = 0 (off)
- wait or poll for **CSR.CON** = 0
- select the operation mode : **FRAMECON.MODE** = 3 (LIN)
- configure the slave mode : **LINCON.MS** = 0
- configure the baud rate : **BRG**, autobaud detection enable, **BRD.UPPER** / **LOWERLIMIT**
- configure the bit timing : **BITCON.PRESCALER** and **OVERSAMPLING**
- configure the bit sampling : **IOCR.DEPTH**, **BITCON.SM**, **SAMPLEPOINT**
- configure the frame parameter : **FRAMECON.PEN** = 0 (no parity), **STOP**= 1
- configure the break length : **LINBTIMER.BREAK** = 11 (typ)
- configure the header timeout : **LINHTIMER.HEADER**
- configure the delay parameter : **FRAMECON.IDLE**
- configure the checksum mode : **LINCON.CSEN**, **LINCON.CSI**
- activate the clock source : set **CSR.CLKSEL**
- wait or poll for **CSR.CON** = 1

In slave mode, the module waits for a header from the master, that is, it waits for a break pulse followed by the sync byte and the ID.

Configure for header reception [slave task]:

- configure the watchdog timer : **DATCON.RESPONSE** =256 (max)
- configure the RXFIFO : **RXFIFOCON** bufmode = 0, flush, **ENI** = 0
- clear the interrupt flags : **FLAGSCLEAR.RHC**, **TRC**, **RRC**, **FEC**, **HTC**, **RTC**, **BDC**, **LPC**, **LAC**, **LCC**, **TFOC/UC/LC**, **RFOC/UC/LC**
- enable the RX interrupt event : **FLAGSENABLE.RHE**
- enable the EX interrupt event : **FLAGSENABLE.HTE**, **CEE**, **FEE**, **LAE**, **LPE**

React to the interrupt(s) generated after receiving the ID byte:

- check error flags : **FLAGS.HT**, **FE**, **LA**, **LP**
- read from the RXFIFO : if no error detected, **RXDATA** = received ID

Asynchronous/Synchronous Interface (ASCLIN)

Check the received ID to determine which response sequence (send, receive or ignore header) to use next.

The LIN slave does not know in advance if it will respond to the header with a transmission himself or it will receive a response by another slave. It looks up the received ID and decides if this ID is associated with a transmit or receive response or if it doesn't care (it is for some other slave). This look-up and the subsequent configuration of the module must be performed by software within the allowed response time.

Send a response to the latest header [slave task]:

(same sequence as in master mode)

Receive a response to the latest header [slave task]:

(same sequence as in master mode)

Ignore the latest header (if the received ID contains an error or is not for this slave) [slave task]:

(same sequence as in master mode)

19.9.4 Using the ENI and HO Bits

Both the ENI (Enable Input) and HO (Header Only) bits affect the reception of the various byte types in the LIN frame.

The HO bit is normally used in cases where the response part of a frame should be ignored; the module waits for the next break signal. The ENI bit can be set by the user software, but is also set by the hardware in slave mode after the sync byte reception, in order to enable the transfer of the ID byte of the header in the RXFIFO.

Asynchronous/Synchronous Interface (ASCLIN)

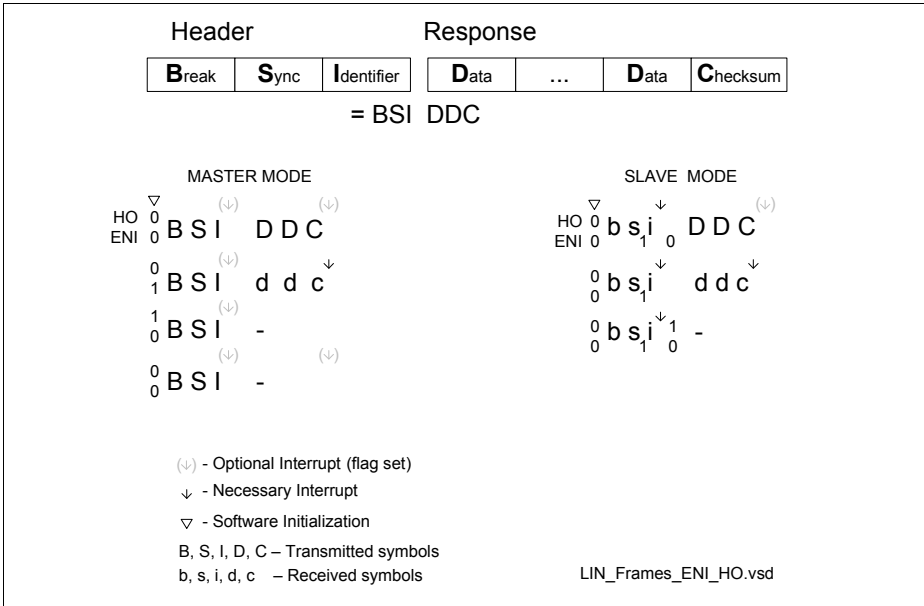


Figure 19-24 ENI and HO bits

19.9.5 LIN Error Recovery

This section describes the behavior of the module in case of errors detected during reception and during transmission of a LIN frame.

Reception Related Errors

In case of an reception error, the receive state machine goes into the state for waiting for break, and the corresponding interrupt is triggered.

ID Parity error

Checksum error

Timeout error

Framing error

Baud Rate error

Asynchronous/Synchronous Interface (ASCLIN)

Transmission Related Errors

Collision error (LIN2.1 mandatory). If the collision detection mechanism is enabled **FLAGSENABLE.CEN**, the frame will be aborted and the transmitter state machine goes to the idle state.

19.9.6 LIN Sleep and LIN Wake-Up

Wake up low pulse in duration of 250us to 5ms can be generated by the module.

Wake up low pulse in duration longer than 150us wakes up a sleeping module.

A master node which has received a wake up pulse can start polling the slaves, that is it can start sending break pulses and headers, and sending or receiving corresponding responses.

A slave which has been woken up shall be capable of receiving LIN headers after a wake-up time of maximum 100ms. A slave issuing a wake-up pulse expects to receive a header within 150ms to 250ms after the end of the wake-up pulse. If the header does not come, the slave issues a wake up pulse again.

Asynchronous/Synchronous Interface (ASCLIN)

19.10 Auto Baud Rate Detection

Auto baud rate detection is active in slave mode during the reception of the sync field in the LIN header. It measures the longest time interval between two falling edges in the 55H sync field. The measured value is loaded in the denominator of the fractional divider and used afterwards for generating the baudrate for the remainder of this frame if the auto baud rate usage is enabled (**LINCON.ABD** = 1).

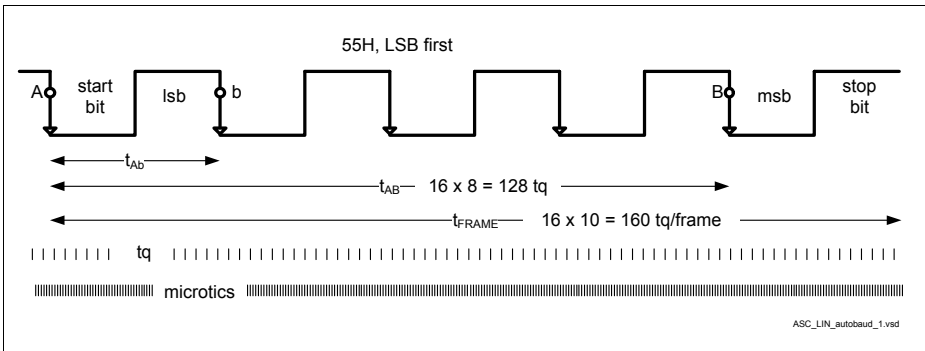


Figure 19-25 Measurement of the Sync field

The following bitfields are available for monitoring the autobaud detection:

- **BRD.MEASURED** - the measured time interval between the first and the fifth falling edge of the sync byte
- **BRD.UPPERLIMIT** - In case the LIN autobaud detection measures a baud rate 14% lower than the nominal one, that is measures a time interval longer than the UPPERLIMIT, a baud rate error event is triggered that, if enabled, generates an interrupt.
- **BRD.LOWERLIMIT** - In case the LIN autobaud detection measures a baud rate 14% higher than the nominal one, that is measures a time interval shorter than the LOWERLIMIT, a baud rate error event is triggered that, if enabled, generates an interrupt.

Auto Baud Rate Operation

The **BRD.UPPERLIMIT** defines the maximum allowed duration for 8 bits in microticks, and therefore the minimum allowed baud rate. In order to define 14% lower baud rate, 8-bit duration 16% longer than the nominal must be entered in the UPPERLIMIT bit field. This is due to the fact that $BaudRate = 1 / BitTime$, and UPPERLIMIT defines the time.

The **BRD.LOWERLIMIT** defines the minimum allowed duration for 8 bits in microticks, and therefore the maximum allowed baud rate. In order to define 14% higher baud rate, 8-bit duration 12% shorter than the nominal must be entered in the LOWERLIMIT bit

Asynchronous/Synchronous Interface (ASCLIN)

field. This is due to the fact that $BaudRate = 1 / BitTime$, and LOWERLIMIT defines the time.

If the autobaud is activated, the fractional divider uses a numerator value of $8 \times (BITCON.OVERSAMPLING + 1)$ and ignores the bit field BRG.NUMERATOR. For the standard LIN protocol oversampling of 16 ($BITCON.OVERSAMPLING = 15$), the numerator value used internally is 128, ignoring the bit field BRG.NUMERATOR. Nevertheless, programming 128, or $8 \times (OVERSAMPLING + 1)$ in BRG.NUMERATOR may increase the clarity of the software.

Regarding the denominator of the fractional divider, defined in BRG.DENOMINATOR, its initial value is the nominal value and is set by the application software. During the operation of the module, the BRD.MEASURED value is automatically loaded into the BRG.DENOMINATOR, as long as it is within the limits.

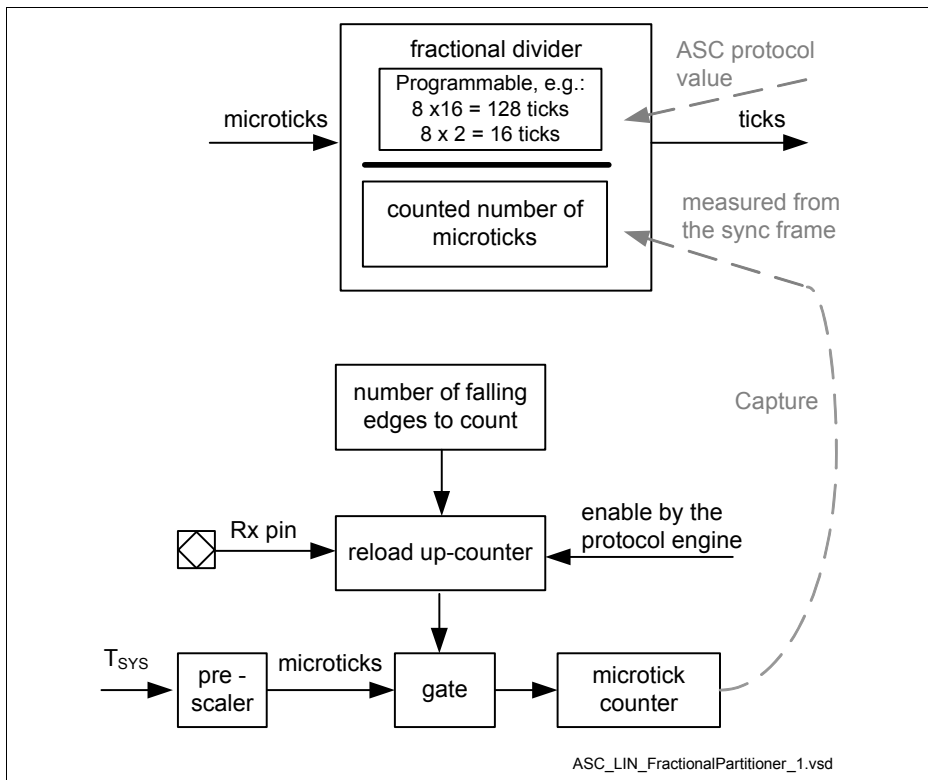


Figure 19-26 Overview of the LIN Auto Baud Detection Principle

Asynchronous/Synchronous Interface (ASCLIN)

Auto baud rate detection measures the time between the first and the fifth falling edge of the sync field in T_{SYS} units and loads this number in the denominator. The LIN protocol uses baud rates in a range between 2400 Baud to 19200 Baud. The expected time at 19.2 KBaud is $8 * 52.1 \text{ us} = 416.8 \text{ us}$ and the expected denominator value at $T_{SYS} = 10\text{ns}$ is in the range of $417\text{us} / 10\text{ns} = 41700$.

The numerator operates internally with the quantum for 8 bits, 16 times oversampling for LIN protocol: $8 * 16 = 128$. The bit field **BRG.NUMERATOR** is ignored.

19.11 Collision Detection

Collision detection monitors the consistency of transmitted and the echoed received bytes in LIN mode and half duplex SPI mode.

Asynchronous/Synchronous Interface (ASCLIN)

19.12 LIN Protocol Control

There is a central LIN protocol state machine. It is connected to the receiver and the transmitter shift registers, the Tx and Rx FIFOs, the checksum logic and the watchdog timers. The machine takes care of generating the sync byte of 55_H and the automatic handling of the checksum. Both the classic LIN V1.3 and the enhanced LIN V2.0 / V2.1 checksum are supported (>> **LINCON** register). The hardware checksum feature is switchable on and off, and the choice between using the classical and the enhanced checksum is done by software with the **DATCON.CSM** bit on a frame by frame basis. As can be seen in the **Figure 19-28**, for the LIN version 2.0 and 2.1, the enhanced checksum is calculated for the identifiers 0...59, and the classical checksum for the identifiers 60...63.

Additionally, the parity of the ID field is generated in master mode, and checked in slave mode. In slave mode, in case of a mismatch between the received and the calculated parity, an error interrupt is raised.

In receive case, if **LINCON.CSI** = 1, the received checksum byte is written to the RXFIFO after the last data byte.

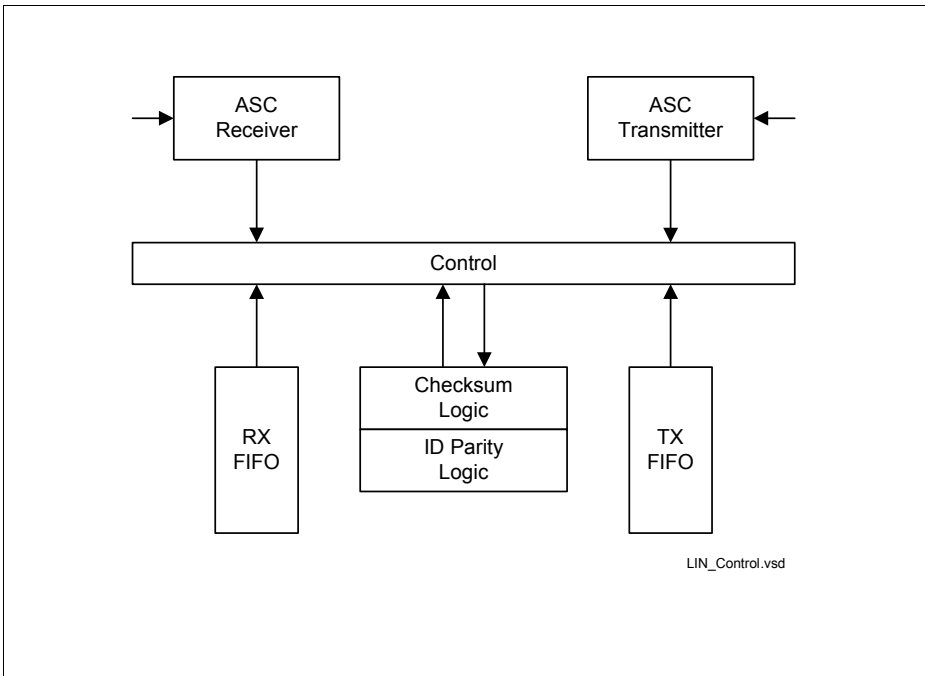


Figure 19-27 Block Diagram of the LIN Control Subsystem

Asynchronous/Synchronous Interface (ASCLIN)

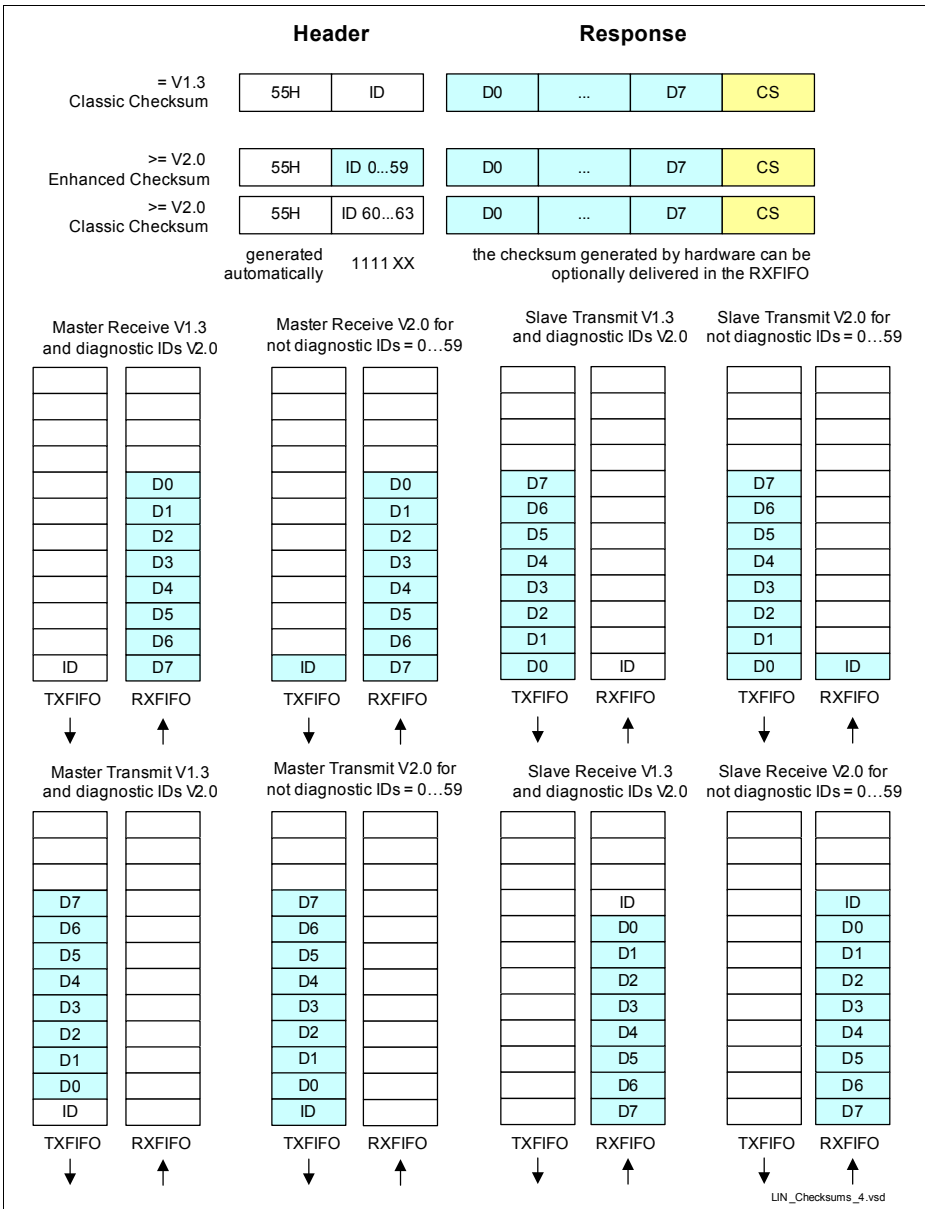


Figure 19-28 Coverage of the Checksum in Different Use Cases

Asynchronous/Synchronous Interface (ASCLIN)

19.13 Interrupts

The ASCLIN module generates three interrupts:

- TX - Transmit Interrupt
 - signals the TxFIFO level event (**FLAGS.TFL**)
- RX - Receive Interrupt
 - signals the RxFIFO level event (**FLAGS.RFL**)
- EX - Extended Error Interrupt
 - signals every error (**FLAGS.PE**, FE, CE, RFO, RFU, TFO) and
 - some additional events (**FLAGS.FED**, RED)

The LIN events related to transmission and reception of header and response are mapped to the corresponding transmit, receive and extended error interrupts

- TX:
 - transmission of the header in master mode completed (**FLAGS.TH**)
 - transmission of a response completed (**FLAGS.TR**)
- RX:
 - reception of the header in a slave mode completed (**FLAGS.RH**)
 - reception of the response completed (**FLAGS.RR**)
- EX:
 - LIN protocol (**FLAGS.BD**, TC) and
 - LIN error events (**FLAGS.HT**, RT, LP, LA, LC)

In order to determine which event is the source of a LIN interrupt, the **FLAGS** register must be polled.

Triggering a DMA

The interrupt signals are used also as DMA trigger signals. The interrupt signals are connected to the Interrupt Router Module, which routes the interrupts either to a CPU or to a DMA. There are no separate DMA trigger signals.

Asynchronous/Synchronous Interface (ASCLIN)

Relationship between the Service Request Nodes and the Event Nodes

There are several events mapped to each service request node.

Each service request (interrupt) node contains an SRC (Service Request Control) register containing a sticky flag bit and the associated set, clear and enable bits. The SRC registers are located in the IR (Interrupt Router) module.

The service request flags in the SRC registers are set by hardware, and if the enable bit is set, will be cleared by hardware when the interrupt servicing starts.

One interrupt node may be triggered by more than one events. Each event which causes an interrupt also has a sticky flag bit and associated set, clear and enable bits. These bits are distributed in four registers: **FLAGS**, **FLAGSET**, **FLAGSCLEAR**, and **FLAGSENABLE** located in the ASCLIN module. Each set of four bits associated to one event builds a virtual “event node”, see **Figure 19-29**.

The event flags are set by hardware and if enabled, trigger an interrupt. They are not cleared by hardware. They are cleared only by software in the corresponding interrupt service routine, which usually polls the flags to find the cause for the interrupts. The event flags can be also set by software for test purposes.

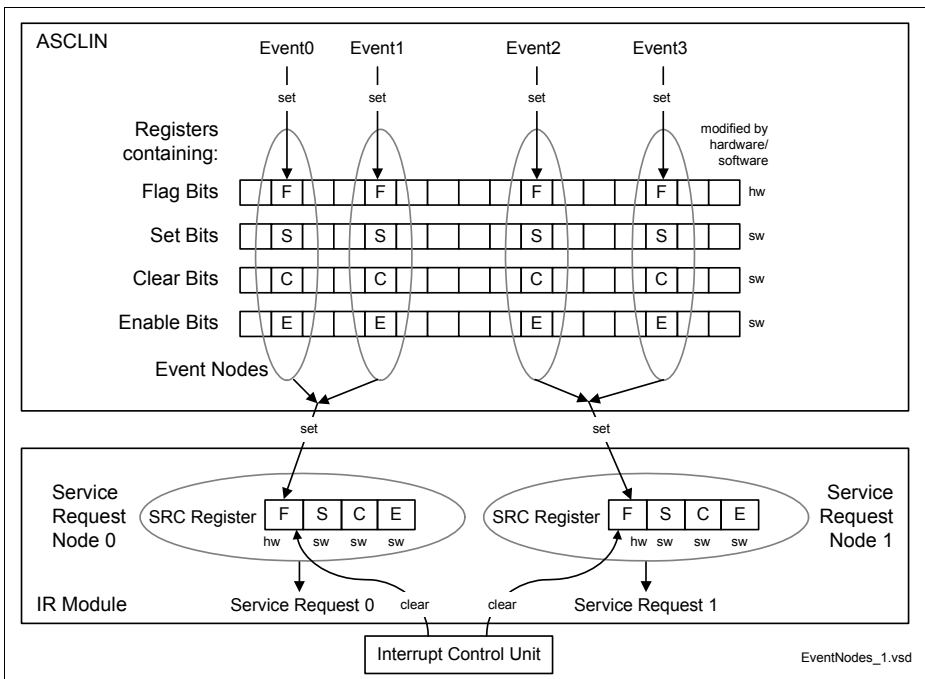


Figure 19-29 Relationship between Event Nodes and Service Request Nodes

Asynchronous/Synchronous Interface (ASCLIN)

Some flags do not generate interrupts and do not have enable bits. They are used either for issuing requests, which are subsequently acknowledged by hardware, or only for polling by software, see **Figure 19-30**.

Clearing an request / acknowledge flag by software is ignored (for example **FLAGS.TRRQ**, **THRQ**, **TWRQ**).

There are no pure polled only flags in the ASCLIN module, although by letting the enable bits disabled, all interrupt event flags can be used as polled only (for example idle and stuck monitoring by using **FLAGS.FED**, **RED**). Setting such bits per software in such a case is meaningless, except for test purposes.

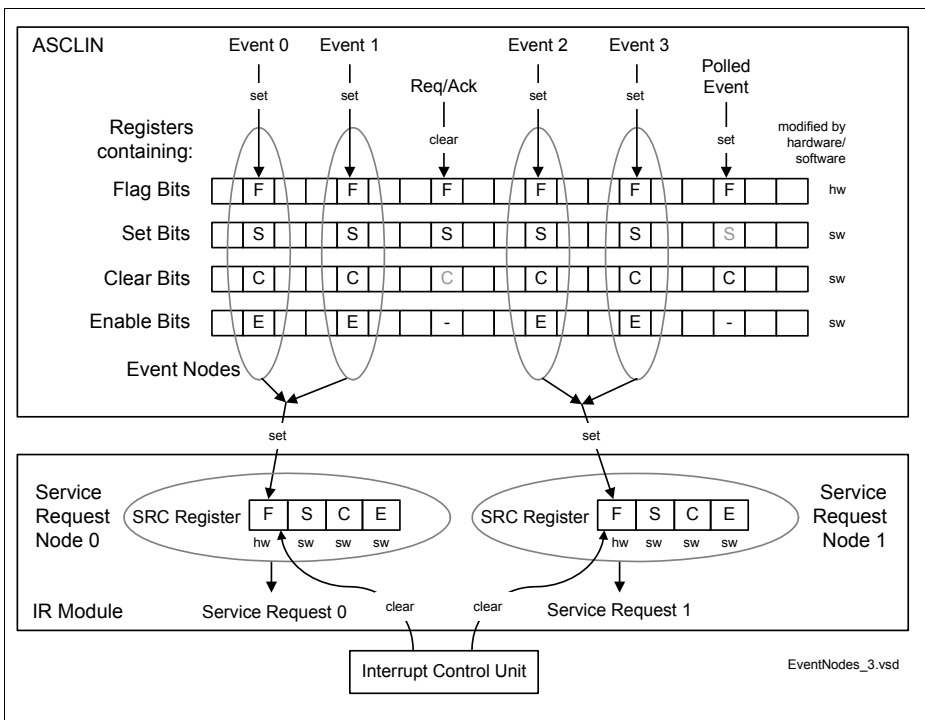
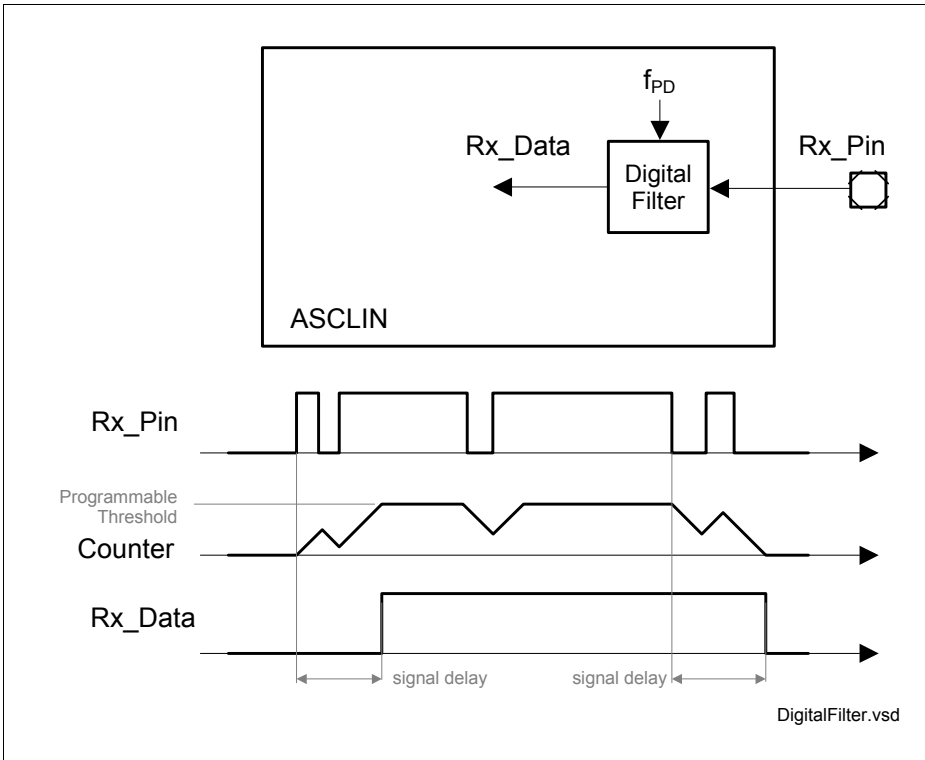


Figure 19-30 Acknowledge Flags and Polled Flags

Asynchronous/Synchronous Interface (ASCLIN)

19.14 Digital Glitch Filter

The digital glitch filter removes short glitches from the input data signal by using an increment / decrement counter with programmable threshold. On the other hand, the filter introduces a delay into the signal path, depending on the digital filter sampling frequency f_{PD} and the threshold programmed in the bit field **IOCR.DEPTH**.


Figure 19-31 Digital Filter

Asynchronous/Synchronous Interface (ASCLIN)

19.15 Suspend, Sleep and Power-Off Behavior

Generally, the ASCLIN module transmits short data frames, not longer than 16 bits, mostly 8 bit long, sometimes with very low baud rates, down to few KBaud. In LIN mode, the data frames build LIN frames.

Simply freezing the module in the middle of an asynchronous ASC frame mostly results in an erroneous data at the receiver side, depending on the data content.

Freezing the module inside the LIN frame transmission, but between the transmission of single bytes results in a timeout error at the receiver side.

If the ASCLIN module gets a request for switching off the clock, it disables immediately all request lines towards the DMA in order to stop further DMA transfers.

In order to power down the module in a well defined way, the user software must take care that the power down request is issued when there is no ongoing single frame or LIN frame transfer and that both FIFOs are empty.

19.15.1 OCDS Suspend

OCDS soft suspend request suspends the module activity at the end of the current transaction. In ASC and SPI cases, this is the end of the current frame. In LIN case, this is the end of the current LIN frame (regular end of response, header timeout, or response timeout).

OCDS hard suspend request, for debugging purposes, immediately freezes the ASCLIN module in the current state. The SPB clock feeding the kernel clock f_{CLC} is immediately switched off. If f_{CLC} is selected for f_A (by using the **CSR.CLKSEL** bit field), the asynchronous clock f_A is also immediately switched off.

*Note: The asynchronous clock f_A remains switched on if a clock source other than f_{CLC} is selected for f_A (by using the **CSR.CLKSEL** bit field).*

Note: Reading and writing of registers is possible but will enable the kernel clock f_{CLC} for a few cycles. Attention: register accesses with clocking in Hard Suspend Mode can have unintended side effects like signals becoming and staying active. This can affect also other modules, so a ASCLIN kernel reset might not be sufficient to bring the system into a defined state.

19.15.2 Sleep Mode

Going into sleep mode implies that the ASCLIN module has finished all activities and that the TX and RX FIFOs are empty. It is the responsibility of the user software to issue a sleep request in a safe time interval, where no race conditions or pipeline effects between the ASCLIN module and the DMA can occur.

The ASCLIN module behaves in the same way as in case of disable request..

Asynchronous/Synchronous Interface (ASCLIN)

19.15.3 Disable Request (Power-Off)

Going into power off implies that the ASCLIN module has finished all activities and that the TX and RX FIFOs are empty. It is the responsibility of the user software to issue a power-off request in a safe time interval, where no race conditions or pipeline effects between the ASCLIN module and the DMA can occur. This is no issue if the power-off procedure is one-way, that is no continuation of operation without reinitialization of the module is expected.

The ASCLIN module sets immediately all outputs to inactive, stops reacting to inputs, the internal state machines go to initial state, the registers remain unchanged, and then the module acknowledges the request. All clocks to the module are switched off.

19.16 Reset Behavior

There are two sources of reset:

- BPI (Bus Peripheral Interface)
- Reset bit in the kernel register KRSTx0.RST and KRSTx1.RST.

Both sources reset the ASCLIN module. They execute an Application Reset, which resets all the registers except the OCS register. All output signals get the reset value.

The **OCS** register is reset by the Debug Reset.

Asynchronous/Synchronous Interface (ASCLIN)
19.17 Use Case Example ASC Interface

This section explains the core functionality of the ASCLIN interface with a code example. The example uses the ASC feature of the module to transmit and receive a data set via loop back mode (see also figure 21-2 (LINK!)). The following settings are used:

- Data length: 8 bit
- Baud rate: 9600 Bd
- One stop bit, parity bit checking, oversampling factor 16, sampling points 7,8 and 9
- Rx and Tx interrupt

Step description to initialize the ASC module with the settings from above:

(Line 1) reset ENDINIT to get access to ENDINIT protected register, here ASCLIN0_CLC. (see also ENDINIT protection chapter 8.9.3(LINK))

(Line 2) enable the control of the module in the clock control register CLC (LINK)

(Line 3) store CLC register in a dummy variable (has to be defined before). Read back to avoid pipeline effects.

(Line 4) set ENDINIT to lock the protected register again.

(Line 5) The clk source gets set in the clock selection register CSR (LINK). Before setting a source, no clk supply ([3:0]=0) must be set. Final clk selection in Line 20.

(Line 6) This line sets the loop back mode in the input and output control register IOCR (LINK). (see also figure 21-2 (LINK))

(Line 7) This line clears and enables the Tx FIFO and also defines the writing size of 1 byte per clk. The Tx FIFO filling level to trigger an interrupt gets set to 0 ([11:8] =0). The Tx FIFO triggers a Tx interrupt, if the Tx FIFO filling level falls to or below that defined level. The Tx interrupt get enabled in Line 14. (see also chapter 21.4.1 TxFIFO Overview (LINK) and TXFIFOCON (LINK) register)

(Line 8) This line installs the Rx FIFO identically like the Tx FIFO from the line above. (see also RXFIFOCON (LINK) and chapter 21.4.2 RxFIFO Overview (LINK))

(Line 9) The oversampling factor (here 16), the sample points (here 7,8 and 9) and the prescaler for the baudrate (here 10) gets configured in the bit configuration register BITCON (LINK). (see also section 21.5. Clock System (LINK))

(Line 10) One stop bit and initialize mode as basic operation mode (necessary before switching to another mode) are getting configured in the FRAMECON (LINK) register. The parity bit feature with parity type even gets also enabled.

(Line 11) The data length of 8 bits gets set in the DATCON (LINK) register.

(Line 12) the clk divider for the baud rate gets set to 48/3125 in the baud rate generation register BRG (LINK) (see also 21.5. Clock System (LINK))

*Note: assuming $f_{clk} = 100$ MHz and (defined in line 8) $prescaler = 10$, $oversampling = 16$:
 $f_{shift} = (f_{clk}/prescaler * 48/3125) / oversampling = 9600$ Bd*

Asynchronous/Synchronous Interface (ASCLIN)

(Line 13) Clear all interrupt flags in the FLAGSCLEAR (LINK) register. Before setting the respective interrupt flags in Line 14

(Line 14) enable the Tx and Rx interrupts in the FLAGSENABLE (LINK) register. The interrupts getting triggered by the FIFO's (see Line 6/7).

(Line 15) This line enables the Tx interrupt in the service request control register SRC_ASCLIN0TX (LINK) and sets the interrupt priority to ASC0TX_PRIO (1...255).

(Line 16) This line enables the Rx interrupt in the service request control register SRC_ASCLIN0RX (LINK) and sets the interrupt priority to ASC0RX_PRIO (1...255).

(Line 17 and 18) The interruptHandlerInstall function gets called (this function has to be written first, see interrupt handler example (LINK) for more details). This function installs the interrupt service routine (ISR) entry address in the interrupt vector array with the priority ASC0TX_PRIO respectively ASC0RX_PRIO.

(Line 19) setting operating mode finally to ASC (see also Line 9).

(Line 20) setting clk source finally to fclk (see also Line 4).

Note: your Tx ISR function prototype would be: void ASC0_TX_irq (void); here could be your ISR code;

Initialization of the ASC interface:

```

(1)  SCU_vResetENDINIT (0);           // enable ENDINIT register
(2)  ASCLIN0_CLC = 0x000;             // disable module control
(3)  dummy = ASCLIN0_CLC;             //read back
(4)  SCU_vSetENDINIT (0);             // lock ENDINIT register
(5)  ASCLIN0_CSR = 0;                 // clk source, no clk
(6)  ASCLIN0_IOCR = 0x10000001;       // Loopback
(7)  ASCLIN0_TXFIFOCON = 0x00000043;  // init TXFIFO
(8)  ASCLIN0_RXFIFOCON = 0x00000043;  // init RXFIFO
(9)  ASCLIN0_BITCON = 0x830F0009;     // OS 16, SP 7,8,9, PS 10
(10) ASCLIN0_FRAMECON = 0x40000200;   // 1 Stop, Init Mode, P
(11) ASCLIN0_DATCON = 0x7;            //8 Data Bits
(12) ASCLIN0_BRG = 0x00300C35;        // divider for Baudrate
(13) ASCLIN0_FLAGSCLEAR = 0xFFFFFFFF; // Clear all Flags
(14) ASCLIN0_FLAGSENABLE = 0xF0000000; // Enable TX/RX Int.

(15) SRC_ASCLIN0TX = (1 << 10) | ASC0TX_PRIO;
(16) SRC_ASCLIN0RX = (1 << 10) | ASC0RX_PRIO;

```

Asynchronous/Synchronous Interface (ASCLIN)

```
(17) interruptHandlerInstall (ASC0TX_PRI0, & ASC0_TX_irq);  
(18) interruptHandlerInstall (ASC0RX_PRI0, & ASC0_RX_irq);  
  
(19) ASCLIN0_FRAMECON |= 0x00010000; // Mode ASC  
(20) ASCLIN0_CSR = 1; //clk source CLC
```

Note: Before an interrupt is able to occur, the interrupt system has to be globally enabled. The Interrupt Control Register (ICR) holds the global interrupt enable bit (ICR.IE) which enables the CPU service request system. Most compiler support the attribute (or similar):

```
__enable();
```

to set this bit. (See also Architecture Manual for more details)

Asynchronous/Synchronous Interface (ASCLIN)

19.18 Kernel Registers

This section describes the kernel registers of the ASCLIN module. All ASCLIN kernel register names described in this section will be referenced in other parts of the TC26x B-Step User’s Manual by the module name prefix “ASCLIN0_” for the ASCLIN0 interface and “ASCLIN1_” for the ASCLIN1 interface.

All registers in the ASCLIN address spaces are reset with the application reset (definition see SCU section “Reset Operation”).

ASCLIN Kernel Register Overview

Identification Register	Control Registers	Status Registers	Data Registers
ID	IOCR	FLAGS	TXDATA
	TXFIFOCON		RXDATA
	RXFIFOCON		
	BITCON		
	FRAMECON		
	BRG		
	BRD		
	LINCON		
	LINBTIMER		
	LINHTIMER		
	FLAGSSET		
	FLAGSCLEAR		
	FLAGSENABLE		
	CSR		
	DATCON		

ASCLIN_kernel_regs_ov.vsd

Figure 19-32 ASCLIN Kernel Registers

Note: Absolute Register Address = Module Base Address (Table 19-3) + Offset Address (Table 19-4).

Asynchronous/Synchronous Interface (ASCLIN)**Table 19-3 Registers Address Space**

Module	Base Address	End Address	Note
ASCLIN0	F0000600 _H	F00006FF _H	–
ASCLIN1	F0000700 _H	F00007FF _H	–
ASCLIN2	F0000800 _H	F00008FF _H	–
ASCLIN3	F0000900 _H	F00009FF _H	–

Asynchronous/Synchronous Interface (ASCLIN)
Table 19-4 Registers Overview

Register Short Name	Register Long Name	Offset Address	Write ¹⁾ Access	Reset Value
IOCR	Input Output Control Register ²⁾	04 _H	U, SV,P	X000 0000
ID	Module Identification Register	08 _H	BE	00C1 C0XX
TXFIFOCON	TX FIFO Configuration Register	0C _H	U, SV,P	0000 0000
RXFIFOCON	RX FIFO Configuration Register	10 _H	U, SV,P	0000 0000
BITCON	Bit Timing Configuration Reg. ²⁾	14 _H	U, SV,P	0000 0000
FRAMECON	Frame Configuration Register ²⁾	18 _H	U, SV,P	0000 0000
DATCON	Data Configuration Register	1C _H	U, SV,P	0000 0000
BRG	Baud Rate Generation Register ²⁾	20 _H	U, SV,P	0000 0000
BRD	Baud Rate Detection Register ²⁾	24 _H	U, SV,P	0000 0000
LINCON	LIN Configuration Register ²⁾	28 _H	U, SV,P	0000 0000
LINBTIMER	LIN Break Timer Register ²⁾	2C _H	U, SV,P	0000 0000
LINHTIMER	LIN Header Timer Register ²⁾	30 _H	U, SV,P	0000 0000
FLAGS	Flags Register	34 _H	BE	0000 0000
FLAGSSET	Flags Set Register	38 _H	U, SV,P	0000 0000
FLAGSCLEAR	Flags Clear Register	3C _H	U, SV,P	0000 0000
FLAGSENABLE	Flags Enable Register	40 _H	U, SV,P	0000 0000
TXDATA	Transmit Data Register	44 _H	U, SV,P	0000 0000
RXDATA	Receive Data Register	48 _H	BE	0000 0000
CSR	Clock Selection Register	4C _H	U, SV,P	0000 0000
RXDATAD	Receive Data Debug Register	50 _H	BE	0000 0000
BPI Registers				
CLC	Clock Control Register	00 _H	SV, E, P	0000 0003
OCS	OCDS Control and Status Reg.	E8 _H	SV, P	0000 0000
KRSTCLR	Reset Status Clear Register	EC _H	SV, E, P	0000 0000
KRST1	Reset Control Register 1	F0 _H	SV, E, P	0000 0000
KRST0	Reset Control Register 0	F4 _H	SV, E, P	0000 0000
ACCEN1	Access Enable Register 1	F8 _H	SV, SE	0000 0000
ACCEN0	Access Enable Register 0	FC _H	SV, SE	FFFF FFFF

Asynchronous/Synchronous Interface (ASCLIN)

- 1) All registers have read access mode U, SV. Read access from reserved addresses delivers bus error (BE). All registers belong to application (class 3) reset except OCS, which belongs to debug (class 1) reset.
- 2) This register is writable only if **CSR.CLKSEL=0**.

List of Access Protection Abbreviations

U	- User Mode
SV	- Supervisor Mode
BE	- Bus Error
nBE	- no Bus Error
P	- Access Protection, as defined by the ACCEN Register
E	- ENDINIT
SE	- Safety ENDINIT

Asynchronous/Synchronous Interface (ASCLIN)

19.18.1 Kernel Registers

IOCR

The Input and Output Control Register IOCRx determines several properties of the RX and TX signal path:

for the RX signal:

- the alternate input
- the filter depth

for the TX signal

- the trigger source

(>> [Table 19-4](#) register overview)

(>> [Baud Rate Generation](#))

(>> [Bit Timing Properties](#))

IOCR

Input and Output Control Register (04) Reset Value: X000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXM	RXM	CTS EN	LB	SPO L	CPO L	RC POL	0				CTS				
rh	rh	rw	rw	rw	rw	rw	r				rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				DEPTH				0		ALTI					
r				rw				r		rw					

Field	Bits	Type	Description
ALTI	[2:0]	rw	Alternate Input Select Selects the alternate input for the RX signal: 000 _B Alternate Input 0 selected 001 _B Alternate Input 1 selected ... _B ... 111 _B Alternate Input 7 selected

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
DEPTH	[9:4]	rw	Digital Glitch Filter Depth DEPTH determines the number of port input samples clocked with microticks that are taken into account for the calculation of the floating average. The higher the DEPTH is chosen to be, the longer the glitches that are suppressed and the longer the delay of the input signal introduced by this filter. 000000 _B off, default 000001 _B 1 000010 _B 2 000011 _B 3 ... _B ... 111111 _B 63
CTS	[17:16]	rw	CTS Select Selects the CTS input pin out of maximum four possible. 00 _B 0 01 _B 1 10 _B 2 11 _B 3
RCPOL	25	rw	RTS CTS Polarity RCPOL defines the active level or the RTS and CTS signals. Active means ready/clear to send. 0 _B active high 1 _B active low
CPOL	26	rw	Clock Polarity in Synchronous Mode CPOL defines the idle level of the clock signal if the module is set in the SPI mode. The idle level is the level outside the data transmission time intervals. Default is low level. 0 _B Idle low 1 _B Idle high
SPOL	27	rw	Slave Polarity in Synchronous Mode Defines the idle level of the SLSO signal, which is the level outside the data transmission, leading and trailing time intervals. 0 _B Idle low 1 _B Idle high

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
LB	28	rw	<p>Loop Back Mode Enables the in module connection of the transmit signal to receive signal. If Loop-back is enabled, the module can be run and tested without an external connection, in ASC and SPI modes. In LIN mode, loopback should not be used, because the module can be either master or slave.</p> <p>0_B Disabled 1_B Enabled</p>
CTSEN	29	rw	<p>Input Signal CTS Enable Enables the sensitivity of the module to the external CTS signal. If disabled, the CTS signal is considered being permanently active.</p> <p>0_B Disabled 1_B Enabled</p>
RXM	30	rh	<p>Receive Monitor Shows the status of the receive signal.</p> <p>0_B Current signal is low. 1_B Current signal is high.</p>
TXM	31	rh	<p>Transmit Monitor Shows the status of the transmit signal.</p> <p>0_B Current signal is low. 1_B Current signal is high.</p>
0	3, [15:10] , [24:18]	r	<p>Reserved Read as 0; should be written with 0.</p>

Asynchronous/Synchronous Interface (ASCLIN)
TXFIFOCON

 (>> [Table 19-4](#) register overview)

TXFIFOCON
TX FIFO Configuration Register
(0C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											FILL				
r											rh				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		INTLEVEL				INW		0			ENO		FLU SH		
r		rw				rw		r			rw		w		

Field	Bits	Type	Description
FLUSH	0	w	Flush the transmit FIFO Write of 1 brings the Tx FIFO in empty state. Write of 0 has no effect. 0 _B No effect 1 _B Empty the Tx FIFO
ENO	1	rw	Transmit FIFO Outlet Enable Enables the TxFIFO outlet. In SPI and ASC modes, data transmission starts immediately when the data is available, whereas in LIN case the transmission start is controlled by the protocol engine. 0 _B Disabled. In LIN case, if the protocol engine tries to fetch data. 1 _B Enabled. In LIN case, no data is moved to the shift register until it is fetched by the protocol engine.
INW	[7:6]	rw	Transmit FIFO Inlet Width Defines the number of bytes written to the Tx FIFO with one FPI bus write. 00 _B 0 01 _B 1 10 _B 2 11 _B 4

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
INTLEVEL	[11:8]	rw	FIFO Interrupt Level Defines the filling level that triggers an re-fill interrupt or DMA access. An interrupt is generated when the filling level falls to INTLEVEL or below, each time when a data byte is taken out of the FIFO 0000 _B 0 0001 _B 1 0010 _B 2 ... _B ... 1111 _B 15
FILL	[20:16]	rh	FIFO Filling Level Read only bit-field containing the current filling level of the FIFO. 00000 _B 0 00001 _B 1 ... _B ... 10000 _B 16 10001 _B reserved 10010 _B reserved 10011 _B reserved 10100 _B reserved 10101 _B reserved 10110 _B reserved 10111 _B reserved 11000 _B reserved 11001 _B reserved 11010 _B reserved 11011 _B reserved 11100 _B reserved 11101 _B reserved 11110 _B reserved 11111 _B reserved
0	[5:2], [15:12], [31:21]	r	Reserved

Asynchronous/Synchronous Interface (ASCLIN)
RXFIFOCON

 (>> [Table 19-4](#) register overview)

RXFIFOCON
RX FIFO Configuration Register
(10_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUF		0								FILL					
rw		r								rh					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				INTLEVEL				OUTW		0				ENI	FLU SH
r				rw				rw		r				rwh w	

Field	Bits	Type	Description
FLUSH	0	w	Flush the receive FIFO Write of 1 brings the Rx FIFO in empty state. Write of 0 has no effect. 0 _B No effect 1 _B Empty the Rx FIFO
ENI	1	rwh	Receive FIFO Inlet Enable Enables the receiver and the filling of the Rx FIFO through the shift register. In LIN slave mode, this bit is set by hardware after the correct reception of the sync byte. The software can clear this bit after reception of an foreign ID in order to suppress the reception of the following response. 0 _B Disabled 1 _B Enabled
OUTW	[7:6]	rw	Receive FIFO Outlet Width Defines the number of bytes read to the Rx FIFO with one FPI bus read. 00 _B 0 01 _B 1 10 _B 2 11 _B 4

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
INTLEVEL	[11:8]	rw	FIFO Interrupt Level Defines the filling level that triggers a drain interrupt or DMA access. An interrupt is generated when the filling level rises to INTLEVEL or beyond, each time when a data byte is delivered to the FIFO. 0000 _B 1 0001 _B 2 ... _B ... 1111 _B 16
FILL	[20:16]	rh	FIFO Filling Level Read only bit-field containing the current filling level of the FIFO. 00000 _B 0 00001 _B 1 ... _B ... 10000 _B 16 10001 _B reserved 10010 _B reserved 10011 _B reserved 10100 _B reserved 10101 _B reserved 10110 _B reserved 10111 _B reserved 11000 _B reserved 11001 _B reserved 11010 _B reserved 11011 _B reserved 11100 _B reserved 11101 _B reserved 11110 _B reserved 11111 _B reserved

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
BUF	31	rw	Receive Buffer Mode If this bit is zero, then the RXFIFO behaves normally as described in the ITS If this bit is set, the RXFIFO behaves as simple 32-bit one stage RX buffer, which is overwritten with each new received data. The received bits appear in the RXDATA register on the lowest bit locations. The upper locations are padded with zeros. 0 _B RXFIFO 1 _B Single Stage RX Buffer
0	[5:2], [15:12], [30:21]	r	Reserved

Asynchronous/Synchronous Interface (ASCLIN)

BITCON

The BITCON Register defines the integer timer parameters in the baud rate generation block.

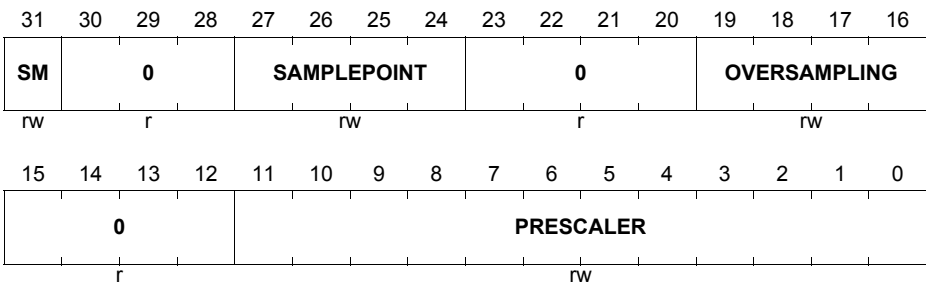
(>> [Table 19-4](#) register overview)

(>> [Baud Rate Generation](#))

(>> [Bit Timing Properties](#))

BITCON

Bit Configuration Register (14_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
PRE SCALER	[11:0]	rw	Prescaling of the Fractional Divider Prescaler in the range of 1 to 4096. Used also as a microtick generator for the input digital filter.
OVER SAMPLING	[19:16]	rw	Oversampling Factor Defines the bit length in ticks in the range of 1 to 16. The lengths of 1 to 3 are not allowed. The position of the sampling points is shown in Figure 19-16 . 0000 _B 1 (not allowed) 0001 _B 2 (not allowed) 0010 _B 3 (not allowed) 0011 _B 4 ... _B ... 1111 _B 16

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
SAMPLE POINT	[27:24]	rw	<p>Sample Point Position Programmable in the range of 0 to 15 according to the Figure 19-16. For example, if three sample points at position 7, 8, 9 are required, this bit field would contain 9.</p> <p>0000_B 0 (not allowed) 0001_B 1 ..._B ... 1111_B 15</p> <p>In SPI mode, this bit field + 1 defines the length of the first SCLK half period in ticks. Values equal or higher then the OVERSAMPLING value are forbidden.</p>
SM	31	rw	<p>Sample Mode Number of samples per bit.</p> <p>0_B 1 1_B 3</p>
0	[15:12], [23:20], [30:28]	r	Reserved

Asynchronous/Synchronous Interface (ASCLIN)

FRAMECON

The parameters regarding the properties of the message frame of the ASC communication are controlled by the Frame Control Register FRAMECON.

(>> [Table 19-4](#) register overview)

FRAMECON

Frame Control Register (18_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ODD	PEN	CEN	MSB	0										MODE	
r/w	r/w	r/w	r/w											r/w	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	LEAD		STOP		IDLE		0								
r	r/w		r/w		r/w		r								

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
IDLE	[8:6]	rw	<p>Duration of the IDLE delay Defines the duration of the IDLE delay in bit times. If more characters are available in the TXFIFO, this is the pause inserted between the characters. In the SPI mode, this is the idle time between the frames. In the ASC and LIN mode, this is the pause inserted between transmission of bytes. Idle also applies to the pause between the header and the response (response space).</p> <p><i>Note: The collision detection runs in parallel to the idle delay and in LIN mode may extend the time between two bytes for one bit length. This effect may occur if the the round trip delay including the digital filter delay is longer than the idle delay.</i></p> <p>000_B 0 001_B 1 ..._B ... 111_B 7</p>
STOP	[11:9]	rw	<p>Number of Stop Bits Defines the number of stop bits in ASC and LIN mode, or the trailing delay in SPI mode. In ASC mode, standard values are 1 and 2. In LIN mode, standard value is 1. In SPI mode there is no standard value. Nevertheless, all settings are possible in all modes.</p> <p>000_B 0 (not allowed in ASC and LIN modes) 001_B 1 ..._B ... 111_B 7</p>
LEAD	[14:12]	rw	<p>Duration of the Leading Delay Defines the leading delay in bit times in SPI mode. Has no meaning in the ASC mode. In LIN mode, this is a delay inserted between the end of the break and the start of the sync character.</p> <p>000_B 0 (not allowed in LIN mode) 001_B 1 ..._B ... 111_B 7</p>

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
MODE	[17:16]	rw	<p>Mode Selection</p> <p>This bit field defines the basic operating mode of the module.</p> <p>In initialize mode, all outputs are at inactive level, and the module does not respond to the input signals. Changing the mode of the module must be done by switching first to initialize mode, and then to the other mode.</p> <p>The SCLK signal generated by the module is active only in the SPI mode.</p> <p>The CTS output generated by the module is active only in the ASC mode.</p> <p>00_B Initialize mode 01_B ASC mode 10_B SPI mode 11_B LIN mode</p>
MSB	28	rw	<p>Shift Direction</p> <p>Defines the shift direction of the shift register. Relevant for the SPI mode. In ASC and LIN modes, should be set to zero. Parity bit is shifted out last independently of the shift direction.</p> <p>0_B LSB first 1_B MSB first</p>
CEN	29	rw	<p>Collision Detection Enable</p> <p>Enables the collision detection mechanism.</p> <p>0_B Disabled 1_B Enabled</p>
PEN	30	rw	<p>Parity Enable</p> <p>Enables the parity bit attached to the data bits. Parity bit can be used for ASC and SPI protocols. The standard LIN bytes do not use this parity bit.</p> <p>0_B Disabled 1_B Enabled</p>

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
ODD	31	rw	Parity Type Defines the type of parity bit attached to the data bits. This setting is valid for all modes of operation (ASC, LIN, SPI). 0 _B Even 1 _B Odd
0	[5:0], 15, [27:18]	r	Reserved Read as 0; should be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)
DATCON

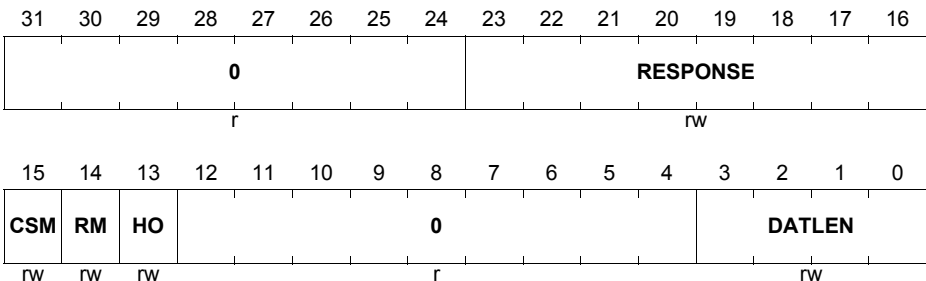
DATCON register defines the number of bits in the ASC (affecting the LIN protocol, if used) and SPI frames, data length or the number of data bytes in a LIN response (if any) and the checksum mode. It additionally defines the time window for the LIN response.

In case the whole LIN response does not fit in this time window, an error interrupt is generated. The measurement starts at the end of the last bit of the header, and ends at the end of the last bit of the response.

(>> [Table 19-4](#))

DATCON

Data Configuration Register (1C_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RESPONSE	[23:16]	rw	Response Timeout Threshold Value Defines the timer limit in the range of 1 to 256 bit times.
HO	13	rw	Header Only Defines if the LIN frame shall consist of a header and response or of a header only. 0 _B Header and response expected 1 _B Header only expected, response ignored
RM	14	rw	Response Mode Defines if the RESPONSE bit field defines a LIN Response or LIN Frame timeout threshold. See Figure 19-23 . 0 _B Frame 1 _B Response

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
CSM	15	rw	Checksum Mode Defines if the classic or the enhanced checksum will be calculated by the checksum block. 0 _B Classic 1 _B Enhanced
DATLEN	[3:0]	rw	Data Length Defines the number of bits in a character. In the ASC mode, standard length is 7, 8, or 9 bits. In the SPI mode, there is no standard length. In ASC and SPI modes, any length from 2 to 16 bits is possible, although not standard for some protocols. In LIN mode, standard length is 8 bits per character. Therefore, this field defines the number of data bytes of the response. 0000 _B 1 0001 _B 2 0010 _B 3 0011 _B 4 ... _B ... 1111 _B 16
0	[31:24], [12:4]	r	Reserved

Asynchronous/Synchronous Interface (ASCLIN)

BRD

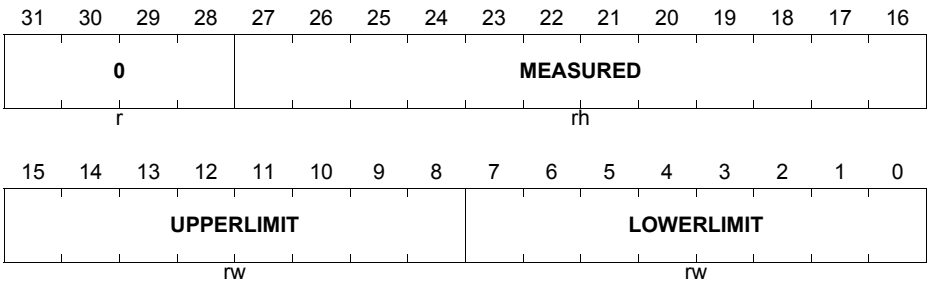
The BRD defines the properties specific for the automatic baud rate detection when the module operates as a LIN slave.

(>> [Table 19-4](#) register overview)

(>> [Auto Baud Rate Detection](#))

BRD

Baud Rate Detection Register (24_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
LOWER LIMIT	[7:0]	rw	Lower Limit This field defines the 8 most significant bits of the 12 bit compare value. The lower four bits are 1000 _B . See Auto Baud Rate Detection .
UPPER LIMIT	[15:8]	rw	Upper Limit This field defines the 8 most significant bits of the 12 bit compare value. The lower four bits are 1000 _B . See Auto Baud Rate Detection .
MEASURED	[27:16]	rh	Measured Value of the Denominator This bit field contains the measured value of the duration of 8-bits form the sync field of the LIN header in microtics. It is automatically loaded in the denominator of the fractional divider, in case of LIN slave operation with autobaud detection.
0	[31:28]	r	Reserved

Asynchronous/Synchronous Interface (ASCLIN)

LINCON

LINCON contains bits that control LIN specific features of the module.

(>> [Table 19-4](#) register overview)

(>> [LIN Protocol Control](#) chapter)

(>> [LIN Master Sequences](#), [LIN Slave Sequences](#))

LINCON

LIN Control Register (28_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0			ABD	MS	CS EN	0	CSI	0								
r			rw		rw	rw	r	rw	r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0																
r																

Field	Bits	Type	Description
CSI	23	rw	Checksum Injection Defines if the received checksum byte is written into the RXFIFO or not. See LIN Protocol Control . 0 _B Not written 1 _B Written
CSEN	25	rw	Hardware Checksum Enable Enables the hardware checksum generation and checking. 0 _B Disabled 1 _B Enabled
MS	26	rw	Master Slave Mode Configures if the module in LIN mode operates as master or as slave. 0 _B Slave 1 _B Master

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
ABD	27	rw	<p>Autobaud Detection</p> <p>Enables the autobaud detection feature in LIN slave mode. In all other operating modes of the module (LIN master, ASC, SPI) not effective.</p> <p>If the autobaud detection is disabled (the oscillator precision of the slave is sufficient), the measurement is still active and the FLAGS.LA will be set when the value is outside the BRD.LOWERLIMIT and .UPPERLIMIT range. Additionally the stop bit of the sync field is checked if correct. If not correct, a framing error is triggered.</p> <p>0_B Disabled 1_B Enabled</p>
0	24, [22:0], [31:28]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Asynchronous/Synchronous Interface (ASCLIN)

LINBTIMER

This register defines

- break detection limit if the module operates as LIN slave or
- length of the generated break pulse if the module operates as LIN master.

The break timer, if enabled, monitors the bus continuously.

(>> [Table 19-4](#) register overview)

(>> [LIN Break, Wake, Stuck Handling](#))

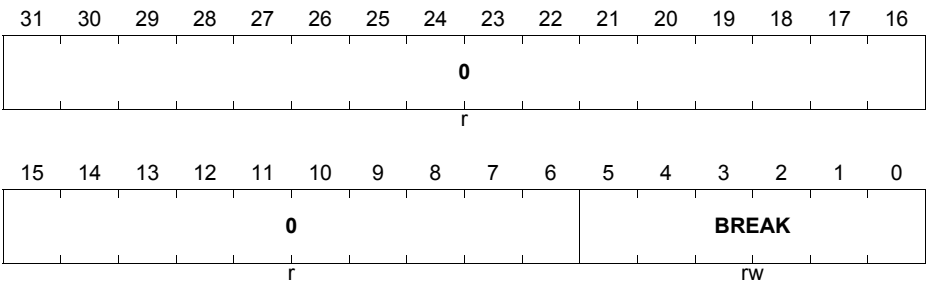
(>> [LIN Master Sequences, LIN Slave Sequences](#))

LINBTIMER

LIN Break Timer Register

(2C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
BREAK	[5:0]	rw	Break Pulse Generation and Detection In LIN slave mode, this bit field defines the duration of the detection threshold for the break pulse. In LIN master mode, this bit field defines the duration of the transmitted break pulse. The time unit is bit time.
0	[31:6]	r	Reserved Read as 0; should be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)

LINHTIMER

LINHTIMER register defines the time windows for the header of a LIN frame.

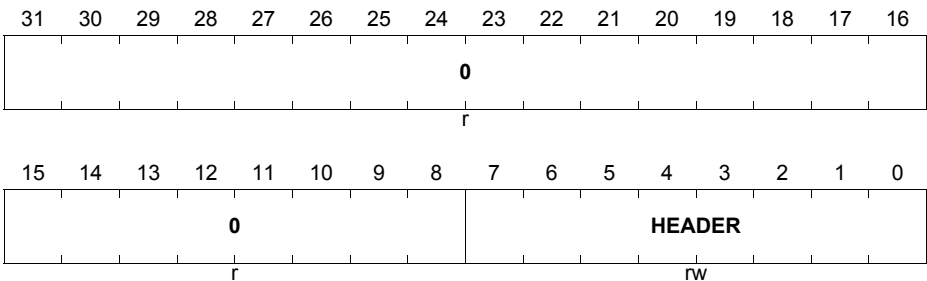
In master mode, the timer starts counting at the falling edge of the break pulse, and stops counting when the last bit of the header (including the stop bits) has been transmitted. If the predefined time in the HEADER bit field is violated, an error interrupt is generated (if enabled).

In slave mode, the timer starts counting when the break pulse has been detected, after a low time of 10 or 11 bit times, and stops counting when the last bit of the header has been received. If the predefined time in the HEADER bit field is violated, an error interrupt is generated (if enabled).

(>> [Table 19-4](#) register overview)

LINHTIMER

LIN Header Timer Register (30_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
HEADER	[7:0]	rw	Header Timeout Threshold Value Defines the timer limit in the range of 1 to 256 bit times.
0	[31:8]	r	Reserved Read as 0; should be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)
FLAGS

The FLAGS register contains all the flag bits of the ASCLIN module: the LIN phase flags (header and response transmit and receive), the overflow flags of the LIN timers, and the standard ASC error flags.

A corresponding interrupt triggering can be enabled using the **FLAGSENABLE** register.

(>> **Table 19-4** register overview)

(>> **LIN Master Sequences, LIN Slave Sequences**)

FLAGS
Flags Register
(34_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
TFL	TFO	0	RFL	RFU	RFO	CE	LC	LA	LP	BD	RT	HT	FE	TC	PE	
rh	rh	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TR RQ	TH RQ	TW RQ				0				RED	FED	0	RR	RH	TR	TH
rh	rh	rh				r				rh	rh	r	rh	rh	rh	rh

Field	Bits	Type	Description
TH	0	rh	Transmit Header End Flag Signals the HEADER_TX_END event. Set by hardware, clear by software. If enabled, a transmit interrupt is triggered. 0 _B No HEADER_TX_END event since the last clear by software 1 _B New HEADER_TX_END event since the last clear by software
TR	1	rh	Transmit Response End Flag Signals that RESPONSE_TX_END event. Set by hardware, clear by software. If enabled, a transmit interrupt is triggered. 0 _B No RESPONSE_TX_END event since the last clear by software 1 _B New RESPONSE_TX_END event since the last clear by software

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
RH	2	rh	Receive Header End Flag Signals that HEADER_RX_END event. Set by hardware, clear by software. If enabled, a receive interrupt is triggered. 0 _B No HEADER_RX_END event since the last clear by software 1 _B New HEADER_RX_END event since the last clear by software
RR	3	rh	Receive Response End Flag Signals that RESPONSE_RX_END event. Set by hardware, clear by software. If enabled, a receive interrupt is triggered. 0 _B No RESPONSE_RX_END event since the last clear by software 1 _B New RX_RESPOSE_END event since the last clear by software
FED	5	rh	Falling Edge from Level 1 to Level 0 Detected This bit is set by hardware when a falling edge is detected on the RX line. 0 _B No falling edge detected 1 _B Faling edge detected
RED	6	rh	Rising Edge from Level 0 to Level 1 Detected This bit is set by hardware when a rising edge is detected on the RX line. 0 _B No rising edge detected 1 _B Rising edge detected
TWRQ	13	rh	Transmit Wake Request Flag Signals that transmission of wake has been requested. No interrupt triggered. As soon as the wake pulse transmission starts, the bit is cleared by the hardware. 0 _B No Transmit Wake Request pending 1 _B Transmit Wake Request pending.
THRQ	14	rh	Transmit Header Request Flag Signals that transmission of header has been requested. No interrupt triggered. As soon as the header transmission starts, the bit is cleared by the hardware. 0 _B No Transmit Header Request pending 1 _B Transmit Header Request pending.

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
TRRQ	15	rh	<p>Transmit Response Request Flag Signals that transmission of response has been requested. No interrupt triggered. As soon as the response transmission starts, the bit is cleared by the hardware.</p> <p>0_B No Transmit Response Request pending 1_B Transmit Response Request pending.</p>
PE	16	rh	<p>Parity Error Flag Signals parity error. If enabled, an error interrupt is triggered. Parity error occurs if the internally calculated parity bit is not equal to the received parity bit.</p> <p>0_B Last message received error free 1_B Last message received with parity error</p>
TC	17	rh	<p>Transmission Completed Flag Signals an end of an ASC or SSC frame. This bit is set after the last stop bit transmission in ASC mode, or after the trailing delay in case of SPI mode. If enabled, an EX interrupt is triggered. Should be cleared by software.</p> <p>0_B No end of frame event pending 1_B End of frame event pending</p>
FE	18	rh	<p>Framing Error Flag Signals framing error. If enabled, an error interrupt is triggered. Framing error occurs if "0" is received at a stop bit position. If autobaud detection is deactivated, then the sync field is checked for framing error.</p> <p>0_B Last message received error free 1_B Last message received with error</p>
HT	19	rh	<p>Header Timeout Flag Signals violation of the header duration limit. If enabled, an error interrupt is triggered.</p> <p>0_B No HEADER_OVERFLOW event since the last clear by software 1_B New HEADER_OVERFLOW event since the last clear by software</p>

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
RT	20	rh	<p>Response Timeout Flag Signals violation of the response or frame duration limit as defined in DATCON.RM bit. If enabled, an error interrupt is triggered.</p> <p>0_B No timeout event since the last clear by software 1_B New timeout event since the last clear by software</p>
BD	21	rh	<p>Break Detected Flag Signals a detection of a break pulse. If enabled, an error interrupt is triggered. Slave mode only.</p> <p>0_B No BWS_OVERFLOW event since the last clear by software 1_B New BWS_OVERFLOW event since the last clear by software</p>
LP	22	rh	<p>LIN Parity Error Flag Signals parity error in the LIN identifier. If enabled, an error interrupt is triggered. Applies to LIN mode only. LIN parity error occurs if the internally calculated parity bits are not equal to the received parity bits.</p> <p>0_B Last ID received error free 1_B Last ID received with parity error</p>
LA	23	rh	<p>LIN Autobaud Detection Error Flag Signals baudrate outside the range defined by BRD.LOWERLIMIT and BRD.UPPERLIMIT.</p> <p>0_B No autobaud detection error 1_B Autobaud detection error</p>
LC	24	rh	<p>LIN Checksum Error Flag Signals checksum error when receiving response, if the internally calculated checksum is different than the received checksum. If enabled, an error interrupt is triggered.</p> <p>0_B Last checksum error free 1_B Last checksum shows an error</p>

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
CE	25	rh	Collision Detection Error Flag When transmitting, signals if the transmitted data differs from the received data. If enabled, an error interrupt is triggered in case of a mismatch. Collision detection is mandatory only when supporting LIN version 2.1. 0 _B No mismatch 1 _B Mismatch detected
RFO	26	rh	Receive FIFO Overflow Flag Signals an overflow error. If enabled, an error interrupt is triggered. 0 _B No overflow error pending 1 _B Overflow error pending
RFU	27	rh	Receive FIFO Underflow Flag Signals an underflow error. If enabled, an error interrupt is triggered. See also RxFIFO Overview . 0 _B No underflow error pending 1 _B Underflow error pending
RFL	28	rh	Receive FIFO Level Flag An interrupt (if enabled) is generated when the RXFIFO filling level rises to INTLEVEL or above, each time when a data byte is delivered to the RXFIFO. This flag signals that the event from above occurred. 0 _B No receive interrupt pending 1 _B Receive interrupt pending
TFO	30	rh	Transmit FIFO Overflow Flag Signals an overflow error. If enabled, an error interrupt is triggered. 0 _B No overflow error pending 1 _B Overflow error pending
TFL	31	rh	Transmit FIFO Level Flag A TXFIFO refill interrupt (if enabled) is generated when the filling level falls to INTLEVEL or below, each time when a data byte is taken out of the TXFIFO. This flag signals that the event from above occurred. 0 _B No transmit interrupt pending 1 _B Transmit interrupt pending

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
0	4, [12:7], 29	r	Reserved Read as 0; should be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)
FLAGSSET

The FLAGSSET register contains the write only bits used to set the corresponding bits in the FLAGS register by software. Setting a flag bit triggers an interrupt, if the corresponding interrupt enable bit is set.

(>> [Table 19-4](#) register overview)

(>> [LIN Master Sequences](#), [LIN Slave Sequences](#))

(>> [FLAGS](#) register)

FLAGSSET
Flags Set Register
(38_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
TF LS	TF OS	0	RF LS	RF US	RF OS	CES	LCS	LAS	LPS	BDS	RTS	HTS	FES	TCS	PES	
w	w	r	w	w	w	w	w	w	w	w	w	w	w	w	w	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TR RQS	TH RQS	TW RQS				0				RE DS	FE DS	0	RRS	RHS	TRS	THS
w	w	w				r				w	w	r	w	w	w	w

Field	Bits	Type	Description
THS	0	w	Flag Set Bit Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
TRS	1	w	Flag Set Bit Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
RHS	2	w	Flag Set Bit Write of “1” in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
RRS	3	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
FEDS	5	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
REDS	6	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
TWRQS	13	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
THRQS	14	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
TRRQS	15	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
PES	16	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
TCS	17	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
FES	18	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
HTS	19	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
RTS	20	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
BDS	21	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
LPS	22	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
LAS	23	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
LCS	24	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
CES	25	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
RFOS	26	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
RFUS	27	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
RFLS	28	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
TFOS	30	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
TFLS	31	w	Flag Set Bit Write of "1" in this bit sets the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Set the corresponding flag
0	4, [12:7], 29	r	Reserved Read as 0; should be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)
FLAGSCLEAR

The FLAGSCLEAR register contains the write only bits used to clear the corresponding bits in the FLAGS register.

(>> [Table 19-4](#) register overview)

(>> [FLAGS](#) register)

FLAGSCLEAR
Flags Clear Register
(3C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
TF LC	TF OC	0	RF LC	RF UC	RF OC	CEC	LCC	LAC	LPC	BDC	RTC	HTC	FEC	TCC	PEC	
w	w	r	w	w	w	w	w	w	w	w	w	w	w	w	w	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
TR RQC	TH RQC	TW RQC				0				RE DC	FE DC	0	RRC	RHC	TRC	THC
w	w	w				r				w	w	r	w	w	w	w

Field	Bits	Type	Description
THC	0	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clear the corresponding flag
TRC	1	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clear the corresponding flag
RHC	2	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clear the corresponding flag

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
RRC	3	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clear the corresponding flag
FEDC	5	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clear the corresponding flag
REDC	6	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clear the corresponding flag
TWRQC	13	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clear the corresponding flag
THRQC	14	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clear the corresponding flag
TRRQC	15	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clear the corresponding flag
PEC	16	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
TCC	17	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
FEC	18	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
HTC	19	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
RTC	20	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
BDC	21	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
LPC	22	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
LAC	23	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
LCC	24	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
CEC	25	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
RFOC	26	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
RFUC	27	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
RFLC	28	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
TFOC	30	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
TFLC	31	w	Flag Clear Bit Write of "1" in this bit clears the bit at the corresponding position in the FLAGS register. 0 _B No action 1 _B Clears the interrupt for the corresponding flag
0	4, [12:7], 29	r	Reserved Read as 0; should be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)
FLAGSENABLE

The FLAGSENABLE register contains the read write bits that enable the error interrupt in case the corresponding event has occurred.

(>> [Table 19-4](#) register overview)

(>> [LIN Master Sequences](#), [LIN Slave Sequences](#))

(>> [FLAGS](#) register)

FLAGSENABLE
Flags Enable Register
(40_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TF LE	TF OE	0	RF LE	RF UE	RF OE	CEE	LCE	LAE	LPE	BDE	RTE	HTE	FEE	TCE	PEE
rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				0					RE DE	FE DE	0	RRE	RHE	TRE	THE
				r					rw	rw	r	rw	rw	rw	rw

Field	Bits	Type	Description
THE	0	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
TRE	1	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
RHE	2	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
RRE	3	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
FEDE	5	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
REDE	6	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
PEE	16	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
TCE	17	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
FEE	18	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
HTE	19	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
RTE	20	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
BDE	21	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
LPE	22	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
ABE	23	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
LCE	24	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
CEE	25	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
RFOE	26	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
RFUE	27	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
RFLE	28	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
TFOE	30	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
TFLE	31	rw	Flag Enable Bit This bit enables the interrupt for the flag at the corresponding position in the FLAGS register. 0 _B Disabled 1 _B Enabled
0	4, [15:7], 29	r	Reserved Read as 0; should be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)

TXDATA

Writing data to this register enters the data to the TXFIFO.

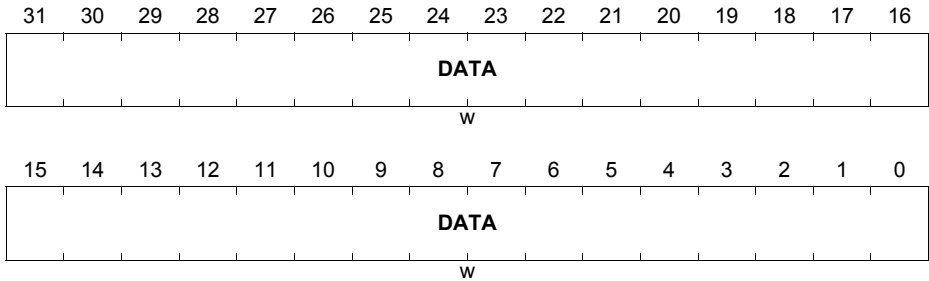
(>> [Table 19-4](#))

TXDATA

Transmit Data Register

(44_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
DATA	[31:0]	w	Data Writing to this bit field writes the content to the TXFIFO, depending on the write width - 8, 16 or 32 bit. read from this register returns 0.

Asynchronous/Synchronous Interface (ASCLIN)

RXDATA

Reading data from this register takes data from the RXFIFO.

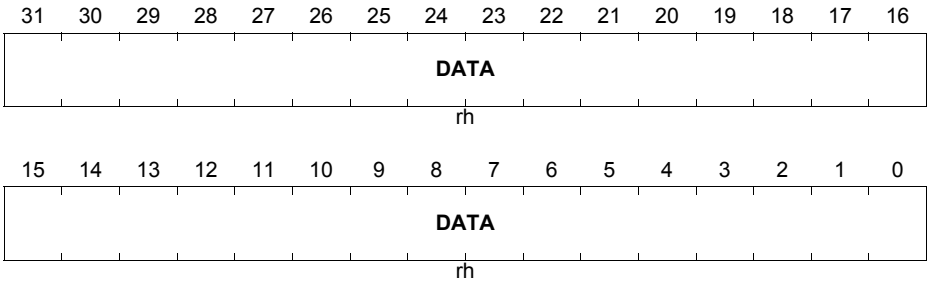
(>> [Table 19-4](#))

RXDATA

Receive Data Register

(48_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
DATA	[31:0]	rh	Data Reading from this bit field takes content from the RXFIFO, depending on the read width - 8, 16 or 32 bit. Write to this register has no effect.

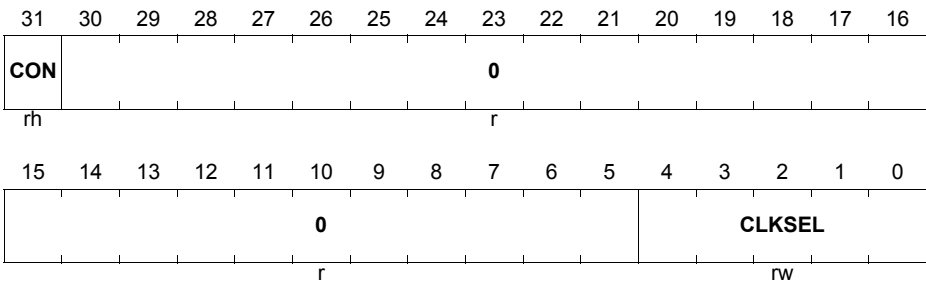
Asynchronous/Synchronous Interface (ASCLIN)
CSR

This register is used to select the clock source for the baud rate generation, detection, timeouts and the SPI delays.

(>> [Table 19-4](#))

CSR

Clock Selection Register (4C_H) **Reset Value: 0000 0000_H**

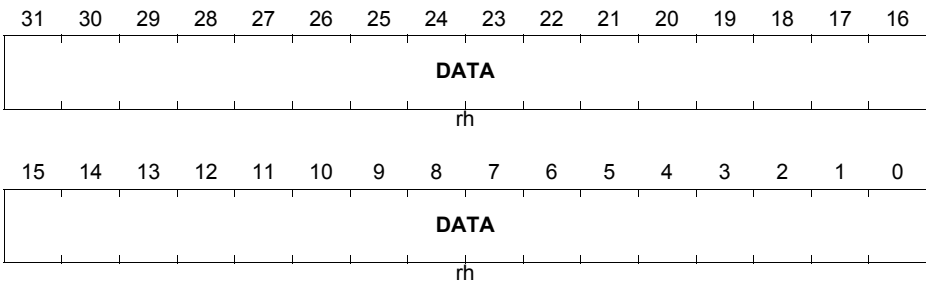


Field	Bits	Type	Description
CLKSEL	[4:0]	rw	Baud Rate Logic Clock Select 00000 _B No clock supplied 00001 _B f_{CLC} 00010 _B XTAL Oscillator Clock f_{OSCO} 00100 _B f_{ERAY} 01000 _B $f_{ASCLINF}$ 10000 _B $f_{ASCLINS}$... _B not allowed
CON	31	rh	Clock On Flag Shows if the clock in the bit time domain is switched on or off. Many configuration registers can be written only if this bit shows 0 (see Table 19-4). 0 _B clock is off 1 _B clock is on
0	[30:5]	r	Reserved Read as 0; should be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)
RXDATAD

Reading data from this register takes data from the RXFIFO, but does not influence the read pointer. This virtual register provides non-destructive read to the RXFIFO.

(>> [Table 19-4](#))

RXDATAD
Receive Data Debug Register
(50_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
DATA	[31:0]	rh	Data Reading from this bit field takes content from the RXFIFO, depending on the read width - 8, 16 or 32 bit, but does not influence the read pointer of the RXFIFO. Write to this register has no effect.

Clock Reconfiguration

The reconfiguration of the clock source has to be done by using two writes: first a write of zero to the CLKSEL bitfield, and then a second write defining the new clock source. Between the first and the second write a delay of minimum $T_W \geq 4 * (1/f_A) + 2 * (1/f_{CLC})$ must be inserted by software, where f_A is the frequency being switched off with the first write. Exception: in case that the f_{CLC} is selected as a baud rate and bit timing logic clock (CSR.CLKSEL = 1), no delay cycles between the writes are necessary. In both cases, simply using one write defining the new clock source is not allowed.

Additionally, always activate the asynchronous clock for at least two f_A cycles:

- after entering the INIT state and
- after device reset

This is an example of a correct sequence:

- CSR.CLKSEL = No_Clock (equals the reset state)
- Wait T_W or poll for CSR.CON = 0

Asynchronous/Synchronous Interface (ASCLIN)

- FRAMECON.MODE = INIT
- CSR.CLKSEL = Clock_On
- Wait T_W or poll for CSR.CON = 1
- CSR.CLKSEL = No_Clock
- Wait T_W or poll for CSR.CON = 0
- FRAMECON.MODE = ASC (or SPI or LIN)
- CSR.CLKSEL = Clock_On
- Wait T_W or poll for CSR.CON = 1

See also [LIN Master Sequences](#)

See also [LIN Slave Sequences](#)

Abort Sequence

If a header has been transmitted in ASCLIN master mode, and the corresponding response does not come, the waiting for the response can be aborted by first entering the INIT mode and then entering the LIN mode. The sequence is identical with the example sequence above.

Asynchronous/Synchronous Interface (ASCLIN)

19.19 Implementation

This section describes the product specific configuration of the ASCLIN module and its interconnection with the rest of the system. The TC26x B-Step contains 4 interfaces, ASCLIN0 to ASCLIN3.

19.19.1 BPI_FPI Module Registers

19.19.1.1 System Registers

Figure 19-33 shows all registers associated with the BPI_FPI module, configured for one kernel.

BPI_FPI Registers Overview

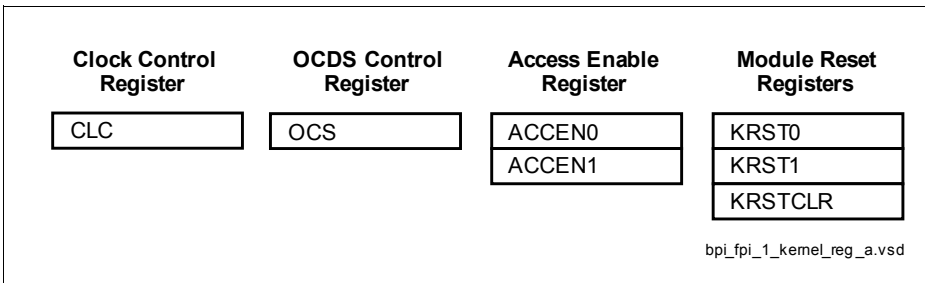


Figure 19-33 BPI_FPI Registers

Clock Control Register (CLC)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI_FPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the f_A module clock signal, sleep mode and disable mode for the module.

(>> Table 19-4 register overview)

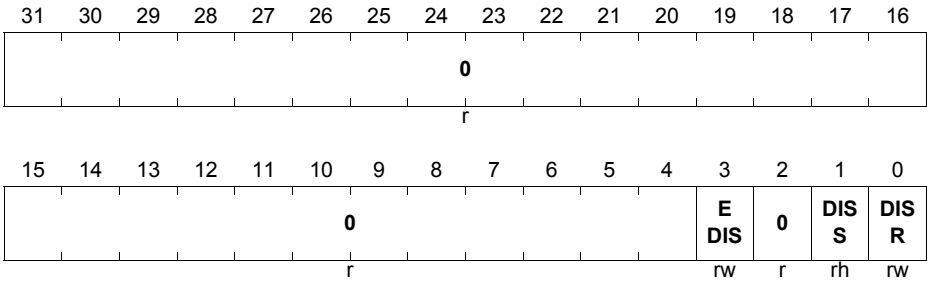
Asynchronous/Synchronous Interface (ASCLIN)

CLC

Clock Control Register

(00_H)

Reset Value: 0000 0003_H



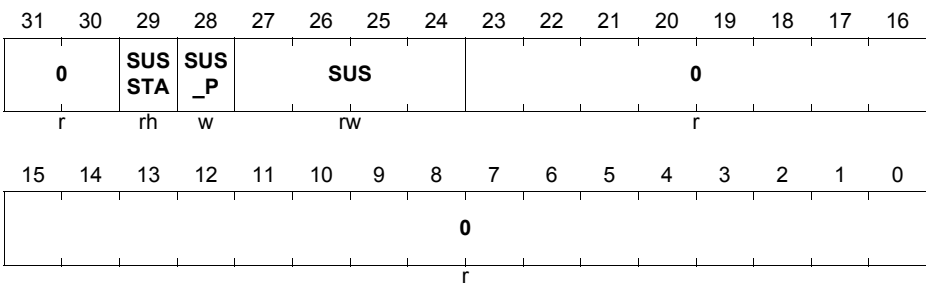
Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
EDIS	3	rw	Sleep Mode Enable Control Used to control module's sleep mode, that is if the sensitivity of the module to the sleep signal is enabled to react to it or disabled to ignore it. 0 _B Enabled 1 _B Disabled
0	[31:4], 2	r	Reserved Read as 0; must be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)
OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32 bit wide only and requires Supervisor Mode

(>> [Table 19-4](#) register overview)

OCS
OCDS Control and Status
(E8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately. 2 _H Soft suspend others , reserved,
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:0], [31:30]	r	Reserved Read as 0; must be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)
Access Enable Register (ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ... , EN31 -> TAG ID 011111_B.

(>> [Table 19-4](#) register overview)

ACCEN0
Access Enable Register 0
(FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
ENn (n = 0-31)	n	rW	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

Asynchronous/Synchronous Interface (ASCLIN)

Access Enable Register (ACCEN1)

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 10000_B to 11111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 10000_B, EN1 -> TAG ID 10001_B, ... , EN31 -> TAG ID 11111_B.

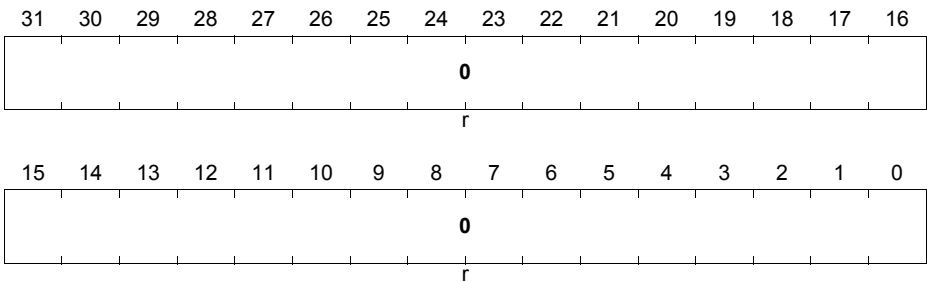
(>> [Table 19-4](#) register overview)

ACCEN1

Access Enable Register 1

(F8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
RSTSTAT	1	rh	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits. 0_B No kernel reset was executed 1_B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

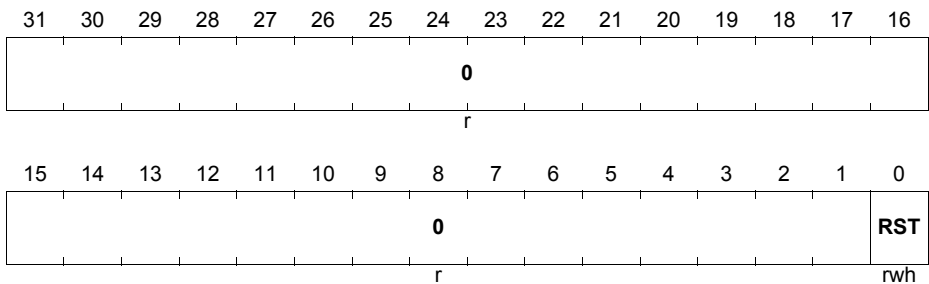
(>> [Table 19-4](#) register overview)

KRST1

Kernel Reset Register 1

(F0_H)

Reset Value: 0000 0000_H



Asynchronous/Synchronous Interface (ASCLIN)

Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. 0_B No kernel reset was requested 1_B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

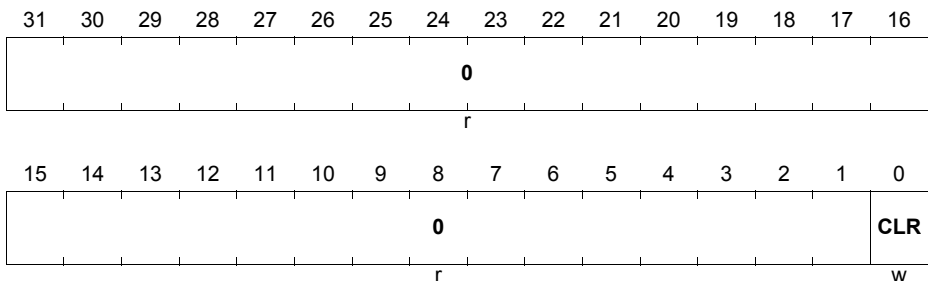
Kernel Reset Status Clear Register (KRSTCLR)

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

(>> [Table 19-4](#) register overview)

KRSTCLR

Kernel Reset Status Clear Register (EC_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0_B No action 1_B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Asynchronous/Synchronous Interface (ASCLIN)
19.20 On-Chip Connections

This section describes the on-chip connections of the ASCLINx module instances.

Port/Pin Connections

The connections of the ASCLIN modules to the pins / ports is described in the chapters regarding the pinning and the ports.

Input signals not connected to ports/pins are connected to the following voltage levels:

- The unconnected ARX signals in the lower half range (RHA to RXD) are connected to (active) low level, “0”
- The unconnected ARX signals in the upper half range (RXE to RXH) are connected to (inactive) high level, “1”
- The unconnected ACTS signals are connected to low level “0”, which is after reset the inactive level (but disabled via **IOCR**.CTSEN bit field, meaning don't care)

Table 19-5 shows an overview for TC26x how bits and bit fields must be programmed for the required I/O functionality of the ASCLIN I/O lines.

Table 19-5 ASCLIN I/O Control Selection and Setup

ASCLIN Module	Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
ASC0	ARX0A / P14.1	ASC_IOCR.ALT1 = 000 _B	P14_IOCR0.PC1 = 0XXXX _B	I
	ARX0B / P15.3	ASC_IOCR.ALT1 = 001 _B	P15_IOCR0.PC3 = 0XXXX _B	I
	ACTS0A / P14.9	ASC_IOCR.CTS = 00 _B	P14_IOCR8.PC9 = 0XXXX _B	I
	ATX0 / P14.0	-	P14_IOCR0.PC0 = 1X010 _B	O
	ATX0 / P14.1	-	P14_IOCR0.PC1 = 1X010 _B	O
	ATX0 / P15.2	-	P15_IOCR0.PC2 = 1X010 _B	O
	ATX0 / P15.3	-	P15_IOCR0.PC3 = 1X010 _B	O
	ARTS0 / P14.7	-	P14_IOCR4.PC7 = 1X010 _B	O
	ASCLK0 / P14.0	-	P14_IOCR0.PC0 = 1X110 _B	O

Asynchronous/Synchronous Interface (ASCLIN)
Table 19-5 ASCLIN I/O Control Selection and Setup (cont'd)

ASCLIN Module	Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
ASC0	ASCLK0 / P15.2	-	P15_IOCR0.PC2 = 1X110 _B	O
ASC1	ARX1A / P15.1	ASC_IOCR.ALT1 = 000 _B	P15_IOCR0.PC1 = 0XXXX _B	I
	ARX1B / P15.5	ASC_IOCR.ALT1 = 001 _B	P15_IOCR4.PC5 = 0XXXX _B	I
	ARX1C / P20.9	ASC_IOCR.ALT1 = 010 _B	P20_IOCR8.PC9 = 0XXXX _B	I
	ARX1D / P14.8	ASC_IOCR.ALT1 = 011 _B	P14_IOCR8.PC8 = 0XXXX _B	I
	ARX1E / P11.10	ASC_IOCR.ALT1 = 100 _B	P11_IOCR8.PC10 = 0XXXX _B	I
	ARX1F / P33.13	ASC_IOCR.ALT1 = 101 _B	P33_IOCR12.PC13 = 0XXXX _B	I
	ARX1G / P02.3	ASC_IOCR.ALT1 = 110 _B	P02_IOCR0.PC3 = 0XXXX _B	I
	ACTS1A / P20.7	ASC_IOCR.CTS = 00 _B	P20_IOCR4.PC7 = 0XXXX _B	I
	ACTS1B / P32.4	ASC_IOCR.CTS = 01 _B	P32_IOCR4.PC4 = 0XXXX _B	I
	ATX1 / P02.2	-	P14_IOCR0.PC0 = 1X010 _B	O
	ATX1 / P11.12	-	P11_IOCR12.PC12 = 1X010 _B	O
	ATX1 / P14.10	-	P14_IOCR8.PC10 = 1X100 _B	O
	ATX1 / P15.0	-	P15_IOCR0.PC0 = 1X010 _B	O
	ATX1 / P15.1	-	P15_IOCR0.PC1 = 1X010 _B	O
	ATX1 / P15.4	-	P15_IOCR4.PC4 = 1X010 _B	O
	ATX1 / P15.5	-	P15_IOCR4.PC5 = 1X010 _B	O

Asynchronous/Synchronous Interface (ASCLIN)
Table 19-5 ASCLIN I/O Control Selection and Setup (cont'd)

ASCLIN Module	Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
ASC1	ATX1 / P20.10	-	P20_IOCR8.PC10 = 1X010 _B	O
	ATX1 / P33.12	-	P33_IOCR12.PC12 = 1X010 _B	O
	ATX1 / P33.13	-	P33_IOCR12.PC13 = 1X010 _B	O
	ARTS1 / P20.6	-	P20_IOCR4.PC6 = 1X010 _B	O
	ARTS1 / P23.1	-	P23_IOCR0.PC1 = 1X010 _B	O
	ASCLK1 / P15.0	-	P15_IOCR0.PC0 = 1X110 _B	O
	ASCLK1 / P20.10	-	P20_IOCR8.PC10 = 1X110 _B	O
	ASCLK1 / P33.11	-	P33_IOCR8.PC11 = 1X010 _B	O
	ASCLK1 / P33.12	-	P33_IOCR12.PC12 = 1X100 _B	O
	ASLSO1 / P14.3	-	P14_IOCR0.PC3 = 1X100 _B	O
	ASLSO1 / P20.8	-	P20_IOCR8.PC8 = 1X010 _B	O
ASLSO1 / P33.10	-	P33_IOCR8.PC10 = 1X100 _B	O	
ASC2	ARX2A / P14.3	ASC_IOCR.ALT1 = 000 _B	P14_IOCR0.PC3 = 0XXXX _B	I
	ARX2B / P02.1	ASC_IOCR.ALT1 = 001 _B	P02_IOCR0.PC1 = 0XXXX _B	I
	ARX2D / P10.6	ASC_IOCR.ALT1 = 011 _B	P10_IOCR4.PC6 = 0XXXX _B	I
	ARX2E / P33.8	ASC_IOCR.ALT1 = 100 _B	P33_IOCR8.PC8 = 0XXXX _B	I
	ARX2G / P02.0	ASC_IOCR.ALT1 = 110 _B	P02_IOCR0.PC0 = 0XXXX _B	I

Asynchronous/Synchronous Interface (ASCLIN)
Table 19-5 ASCLIN I/O Control Selection and Setup (cont'd)

ASCLIN Module	Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
ASC2	ACTS2A / P10.7	ASC_IOC.R.CTS = 00 _B	P10_IOC.R4.PC7 = 0XXXX _B	I
	ACTS2B / P33.5	ASC_IOC.R.CTS = 01 _B	P33_IOC.R4.PC5 = 0XXXX _B	I
	ATX2 / P02.0	-	P02_IOC.R0.PC0 = 1X010 _B	O
	ATX2 / P10.5	-	P10_IOC.R4.PC5 = 1X010 _B	O
	ATX2 / P14.2	-	P14_IOC.R0.PC2 = 1X010 _B	O
	ATX2 / P14.3	-	P14_IOC.R0.PC3 = 1X010 _B	O
	ATX2 / P33.8	-	P33_IOC.R8.PC8 = 1X010 _B	O
	ATX2 / P33.9	-	P33_IOC.R8.PC9 = 1X010 _B	O
	ARTS2 / P10.8	-	P10_IOC.R8.PC8 = 1X010 _B	O
	ARTS2 / P33.4	-	P33_IOC.R4.PC4 = 1X010 _B	O
	ASCLK2 / P02.4	-	P02_IOC.R4.PC4 = 1X010 _B	O
	ASCLK2 / P10.6	-	P10_IOC.R4.PC6 = 1X010 _B	O
	ASCLK2 / P14.2	-	P14_IOC.R0.PC2 = 1X110 _B	O
	ASCLK2 / P33.7	-	P33_IOC.R4.PC7 = 1X010 _B	O
	ASCLK2 / P33.9	-	P33_IOC.R8.PC9 = 1X100 _B	O
	ASLSO2 / P02.3	-	P02_IOC.R0.PC3 = 1X010 _B	O
	ASLSO2 / P10.5	-	P10_IOC.R4.PC5 = 1X110 _B	O

Asynchronous/Synchronous Interface (ASCLIN)
Table 19-5 ASCLIN I/O Control Selection and Setup (cont'd)

ASCLIN Module	Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
ASC2	ASLSO2 / P33.6	-	P33_IOCR4.PC6 = 1X010 _B	O
ASC3	ARX3A / P15.7	ASC_IOCR.ALT1 = 000 _B	P15_IOCR4.PC7 = 0XXXX _B	I
	ARX3C / P20.3	ASC_IOCR.ALT1 = 010 _B	P20_IOCR0.PC3 = 0XXXX _B	I
	ARX3D / P32.2	ASC_IOCR.ALT1 = 011 _B	P32_IOCR0.PC2 = 0XXXX _B	I
	ARX3E / P0.1	ASC_IOCR.ALT1 = 100 _B	P00_IOCR0.PC1 = 0XXXX _B	I
	ARX3F / P21.6	ASC_IOCR.ALT1 = 101 _B	P21_IOCR4.PC6 = 0XXXX _B	I
	ACTS3A / P0.12	ASC_IOCR.CTS = 00 _B	P00_IOCR12.PC12 = 0XXXX _B	I
	ATX3 / P0.0	-	P00_IOCR0.PC0 = 1X011 _B	O
	ATX3 / P0.1	-	P00_IOCR0.PC1 = 1X010 _B	O
	ATX3 / P15.6	-	P15_IOCR4.PC6 = 1X010 _B	O
	ATX3 / P15.7	-	P15_IOCR4.PC7 = 1X010 _B	O
	ATX3 / P20.0	-	P20_IOCR0.PC0 = 1X010 _B	O
	ATX3 / P20.3	-	P20_IOCR0.PC3 = 1X010 _B	O
	ATX3 / P21.7	-	P21_IOCR4.PC7 = 1X010 _B	O
	ATX3 / P32.2	-	P32_IOCR0.PC2 = 1X010 _B	O
	ATX3 / P32.3	-	P32_IOCR0.PC3 = 1X010 _B	O
	ARTS3 / P0.9	-	P00_IOCR8.PC9 = 1X011 _B	O

Asynchronous/Synchronous Interface (ASCLIN)
Table 19-5 ASCLIN I/O Control Selection and Setup (cont'd)

ASCLIN Module	Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
ASC3	ASCLK3 / P0.0	-	P00_IOCR0.PC0 = 1X010 _B	O
	ASCLK3 / P0.2	-	P00_IOCR0.PC2 = 1X010 _B	O
	ASCLK3 / P15.6	-	P15_IOCR4.PC6 = 1X110 _B	O
	ASCLK3 / P15.8	-	P15_IOCR8.PC8 = 1X110 _B	O
	ASCLK3 / P20.0	-	P20_IOCR0.PC0 = 1X011 _B	O
	ASCLK3 / P21.5	-	P21_IOCR4.PC5 = 1X010 _B	O
	ASCLK3 / P21.7	-	P21_IOCR4.PC7 = 1X011 _B	O
	ASCLK3 / P32.3	-	P32_IOCR0.PC3 = 1X100 _B	O
	ASCLK3 / P33.2	-	P33_IOCR0.PC2 = 1X010 _B	O
	ASLSO3 / P0.3	-	P00_IOCR0.PC3 = 1X010 _B	O
	ASLSO3 / P14.3	-	P14_IOCR0.PC3 = 1X101 _B	O
	ASLSO3 / P21.2	-	P21_IOCR0.PC2 = 1X010 _B	O
	ASLSO3 / P21.6	-	P21_IOCR4.PC6 = 1X010 _B	O
	ASLSO3 / P33.1	-	P33_IOCR0.PC1 = 1X010 _B	O

ASCLIN Connection to Itself

the following connection is defined for each ASCLIN instance:

ARTS -> ACTSD

This connection can be useful in the SPI mode, where frequently the transmission and the reception of data are performed in parallel. If this connection is selected by using

Asynchronous/Synchronous Interface (ASCLIN)

IOCR.CTS, the TXFIFO will deliver data for transmission only if there is a free space in the RXFIFO. If not, the TXFIFO will wait for emptying of the RXFIFO by software and then automatically continue the transmission.

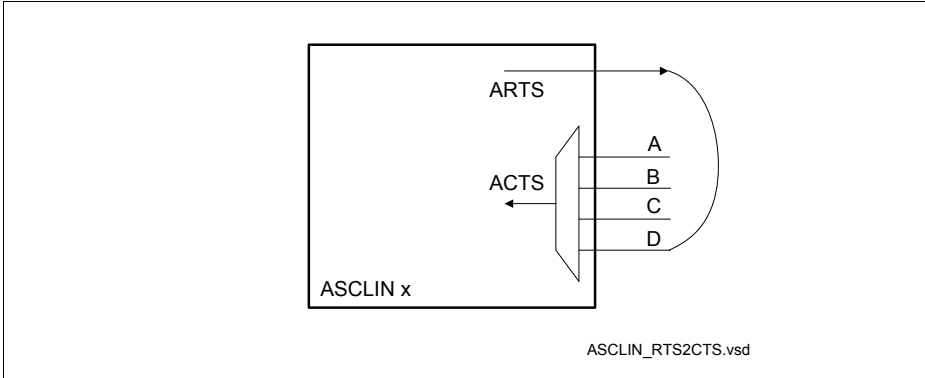


Figure 19-34 ASCLIN ARTS to ACTS Connection

19.21 ASC at CAN Support

The ASC Tx and Rx signals are overlaid with CAN Rx and Tx signal. See the Ports chapter and the Pinning chapter for the particular pin assignments.

A virgin device with a completely erased flash memory can be booted via ASC at CAN pins. See the Boot chapter for details on the implementation.

Queued Synchronous Peripheral Interface (QSPI)

20 Queued Synchronous Peripheral Interface (QSPI)

The main purpose of the QSPI module is to provide synchronous serial communication with external devices using clock, data-in, data-out and slave select signals. The focus of the module is set to fast and flexible communication: either point-to-point or master-to-many slaves communication.

Parallel requests from on chip bus masters to a module will be executed sequentially via the on chip bus system. Read-modify-write feature provides an atomic read/write sequence where no other master can access the module in between. Module hardware semaphores are not supported.

>> [Registers Overview](#)

>> [BPI_FPI Module Registers](#)

20.1 Feature List

This section describes the features of the QSPI module.

- Master and Slave Mode operation
 - Full-duplex operation
 - Half-duplex operation
 - Automatic slave select control
 - Four-wire and three-wire type of connection
- Flexible data format
 - Programmable number of data bits: 2 to 32 data bits (plus parity: 3 to 33 bits)
 - 4 to 32 data bits possible for 50 Mbit/s
 - Programmable shift direction: LSB or MSB shift first
 - Programmable clock polarity: Idle low or idle high state for the shift clock
 - Programmable clock phase: data shift with leading or trailing edge of the shift clock
- Baud rate generation
 - Flexible baud rate and delays (leading, trailing, idle) generation
- Interrupt generation
 - On a transmitter FIFO event
 - On a receiver FIFO event
 - On an error condition (receive, baud rate, transmit error, parity error)
 - On a phase transition (start of frame, end of frame ...)
- QSPI supports control and data handling by the DMA controller
- Flexible QSPI pin configuration
- Hardware supported parity mode
 - Odd / even / no parity
- Seven slave select inputs SLSIB..H in Slave Mode
- Sixteen programmable slave select outputs SLISO[15:0] in Master Mode
 - Automatic SLISO generation with programmable timing
 - Programmable active level and enable control

Queued Synchronous Peripheral Interface (QSPI)

- External demultiplexing of the slave select outputs support
- Several module reset options
 - State machine reset per software (only the state machine)
 - Module reset per software (both FIFOs, all registers and the state machine)
 - Automatic stop option of the state machine in slave mode after baud rate error
- Loop-Back mode
- Interoperability with SSC and USIC modules of Infineon microcontroller families, and with popular (Q)SPI interfaces of multiple suppliers
- Communication stop on RxFIFO full
 - Shift register full and RxFIFO full can pause the communication
 - Interrupt generation
- High Speed Input Capture (HSIC)
 - Provides input capture functionality for ADAS applications
 - 15-bit counter with resolution of $f_{\text{BAUD}2}$

Queued Synchronous Peripheral Interface (QSPI)

20.2 Overview

This section gives a top-level overview of the QSPI.

20.2.1 External Signals

The communication between two devices using QSPI generally uses four signals:

- serial clock SCLK
- data in master to slave direction MTSR (Master Transmit Slave Receive)
- data in slave to master direction MRST (Master Receive Slave Transmit)
- slave select signal SLS

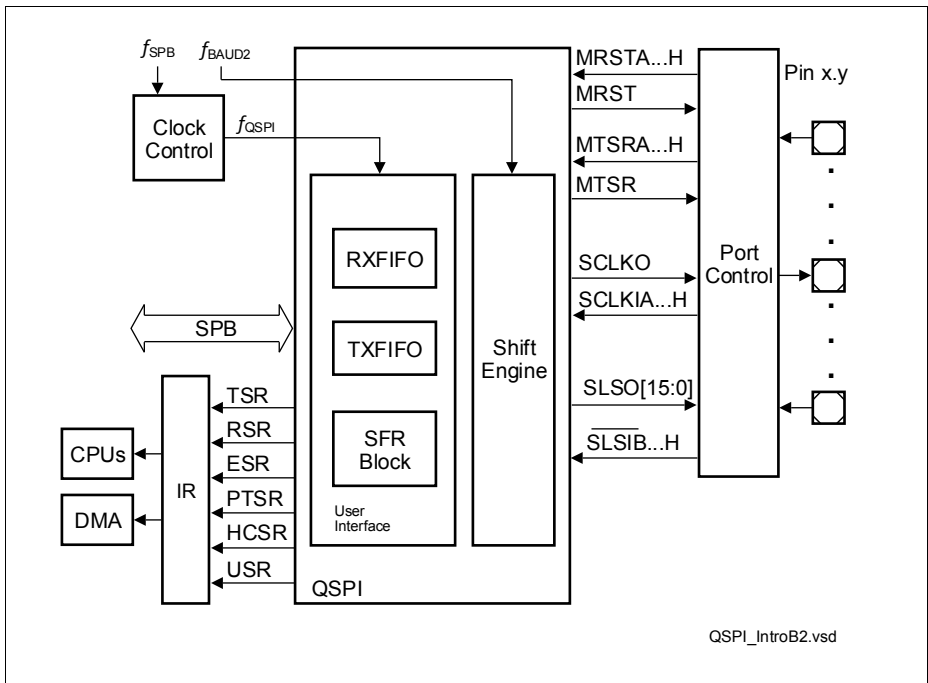


Figure 20-1 External Signals of the QSPI Module

Queued Synchronous Peripheral Interface (QSPI)

20.2.2 Operating Modes

The QSPI operates in one of two modes regarding the generation of the serial clock and slave select signals: master or slave mode. The master generates and drives the clock and select signals, the slave receives these signals.

The QSPI operates in one of three modes regarding the direction of the communication and depending whether the transmission and the reception appear simultaneously or not: duplex, half-duplex and simplex mode.

This gives six possible combinations for the operating mode of the QSPI: master duplex, half-duplex, and simplex; slave duplex, half-duplex, and simplex.

Note: The half-duplex mode can be implemented either by short-cut connection between two different pins on the pcb, one data output and one data input pin, or by using single pins mapped to both data input and data output signals. See the pinning and port chapters for the pinning definition of a particular product. The module itself does not differentiate between the full-duplex, half-duplex or simplex connection.

Queued Synchronous Peripheral Interface (QSPI)

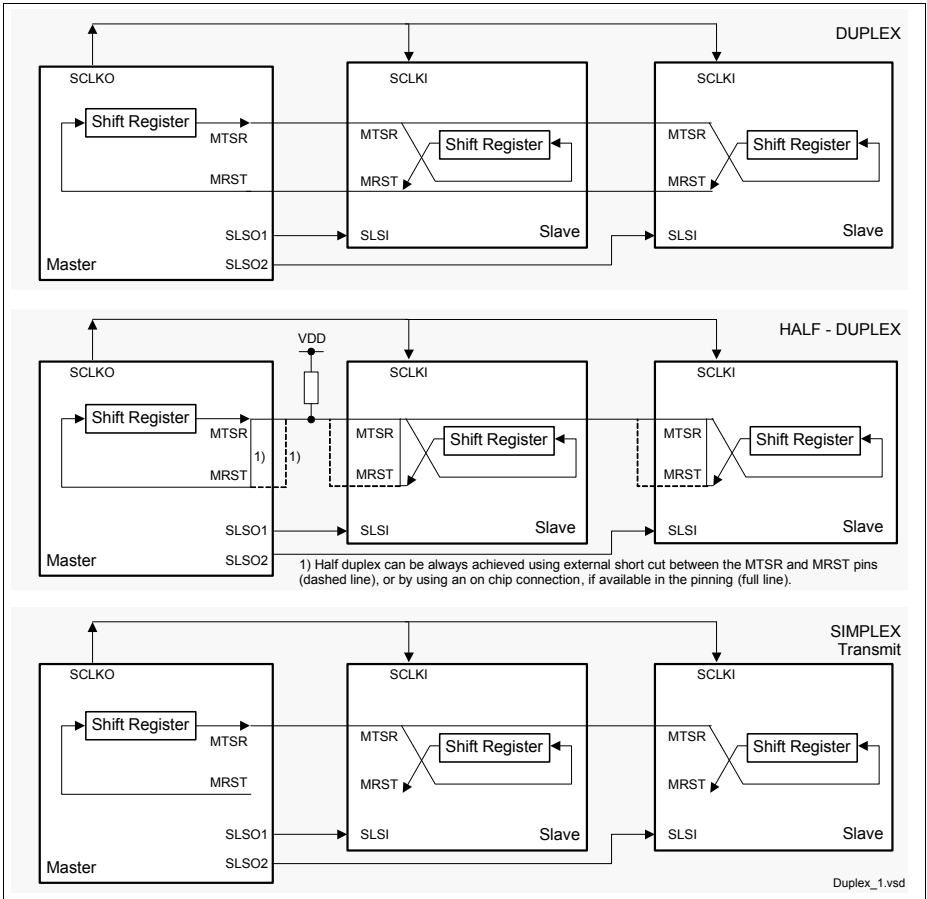


Figure 20-2 Operating Modes and Types of Connections

Queued Synchronous Peripheral Interface (QSPI)

20.2.3 Queue Support Overview

The term "Queue Support" is used in this context to describe the functionality implemented for comfortable switching of the timing configuration of the QSPI frames, depending on the slave select signal which is to be activated. The main feature of the module is the possibility to take both the configuration and data to the TXFIFO, and to track down which TXFIFO entry is configuration, and which data. The QSPI module expects 32 basic configuration bits to be moved with one move (for example DMA move) from some on-chip general purpose RAM to the TXFIFO. These 32 configuration bits from the TXFIFO and the configuration bits from the 8 configuration extension registers **ECONz** contained in the QSPI module, define together the full configuration of the module (see **Figure 20-3**). One extension register is used for two slave selects: ECON0 for SLSO0 and SLSO8, ECON1 for SLSO1 and SLSO9 ...

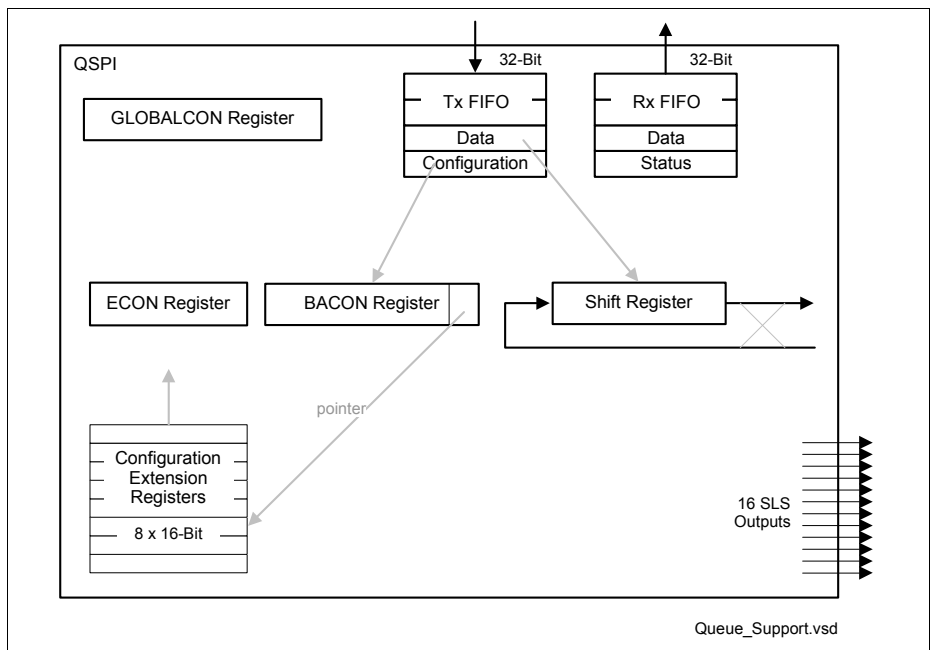


Figure 20-3 Queue Support Overview

Queued Synchronous Peripheral Interface (QSPI)

20.2.4 Architecture Overview

The **Figure 20-4** shows the main blocks of the QSPI module. The Tx FIFO and Rx FIFO provide the user interface. The Shift Register and the “Miscellaneous Logic” block build the state machine of the module. The “Configuration Extensions” block provides comprehensive capabilities for configuring the QSPI frames.

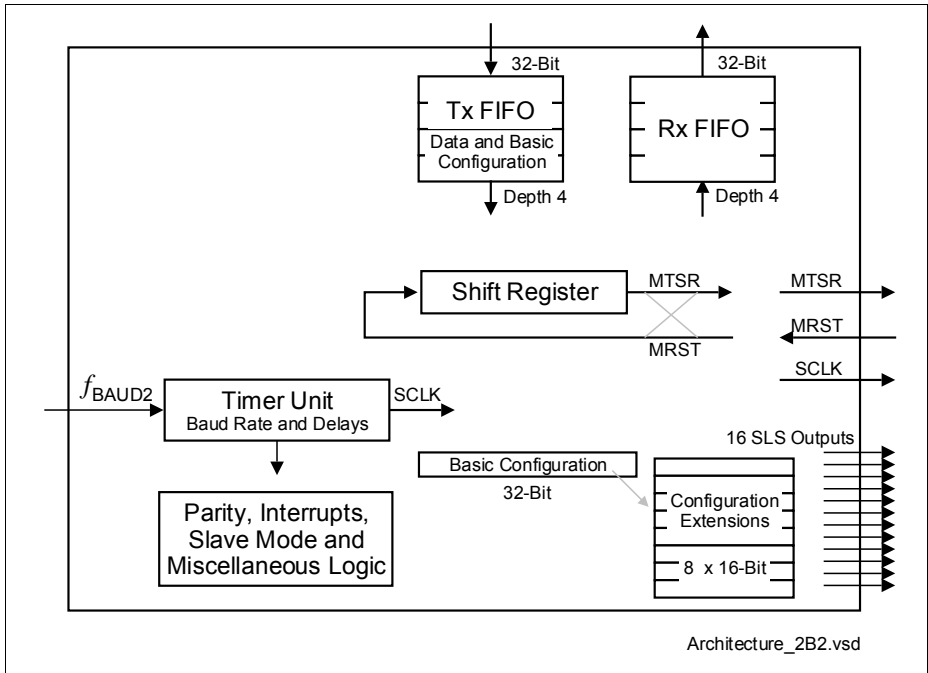


Figure 20-4 QSPI - Architecture Overview

Queued Synchronous Peripheral Interface (QSPI)

20.2.5 Three Wire Connection

The term "Three Wire Connection" is used in this context to describe a connection without slave select signal. This connection relies solely on counting bits for determining the end of the current frame and resetting the shift register state machine, instead of using the slave select signal for this purpose. This way of communication is less robust than communication with slave select, but it saves a pin on both master and slave device.

The name "three wire connection" is true only in case of full-duplex connection, where exactly three signals (clock, data-in, data-out) are needed. In case of half-duplex and simplex connections only two signals (clock, data-in/data-out common data line) are necessary.

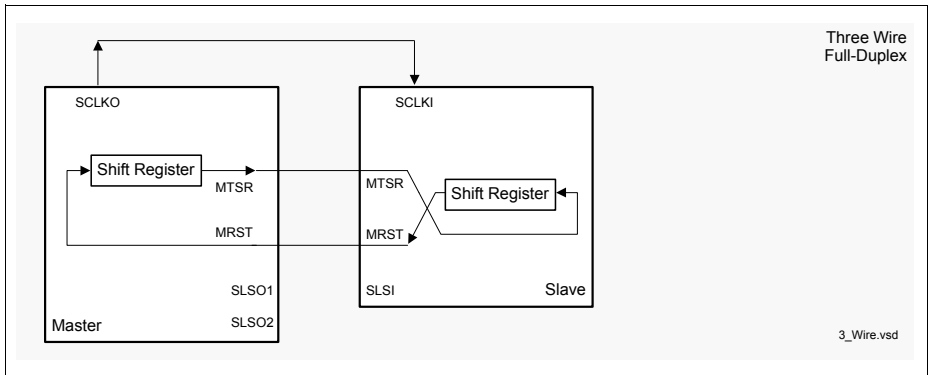


Figure 20-5 Three Wire Connection

Queued Synchronous Peripheral Interface (QSPI)

20.3 Abstract Overview

An abstract overview of the QSPI module is shown in **Figure 20-6**. This document describes the QSPI module according to this view: the State Machine with its configuration and status capabilities, the User Interface, and the Interrupts (in this order).

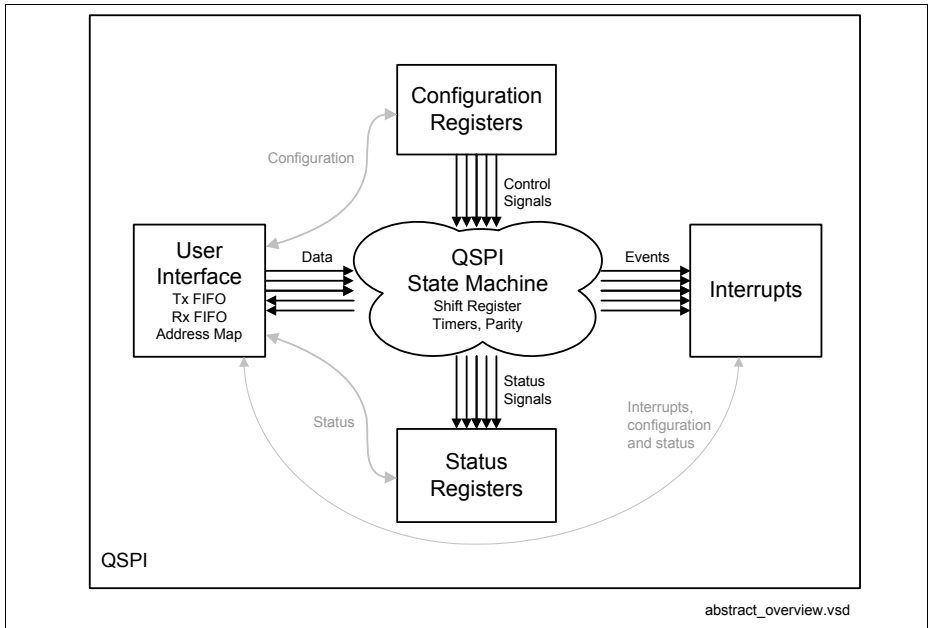


Figure 20-6 QSPI - Abstract Overview

Queued Synchronous Peripheral Interface (QSPI)

20.4 Frequency Domains

The state machine of the QSPI is placed in a separate frequency domain operating with a frequency f_{BAUD2} which can be:

- integer multiple of the SPB bus frequency
- equal to the SPB frequency
- lower than the SPB frequency, derived by integer division

The baud rate is determined by the f_{BAUD2} frequency divided by an integer equal or greater than four.

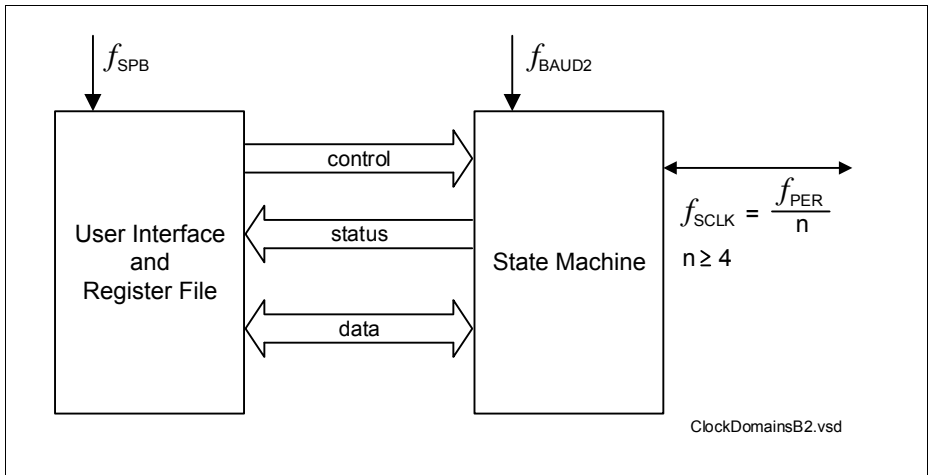


Figure 20-7 QSPI - Frequency Domains

Queued Synchronous Peripheral Interface (QSPI)

20.5 Master Mode State Machine

In master mode, the QSPI module generates the timings, the serial clock, and the slave select signals.

20.5.1 Phases of one Communication Cycle

This section describes the possibilities available for configuring the length of the phases of the QSPI communication: timing delays, data length, duty cycle and data sampling.

A QSPI frame starts with activating a slave select signal SLSO (at transition from idle to leading delay phase), and ends with its deactivation (at transition from trailing delay to wait or idle phase). It is a sequence of five phases: idle delay, leading delay, data phase, trailing delay and an optional wait phase. The idle phase is subdivided in two phases of equal length: idle A and idle B.

Figure 20-8 shows a full and a compressed view of a QSPI frame and its phases. The full view shows all four signals needed for a QSPI connection. The compressed view represents the phases in a single row, in a way suitable to discuss their properties.

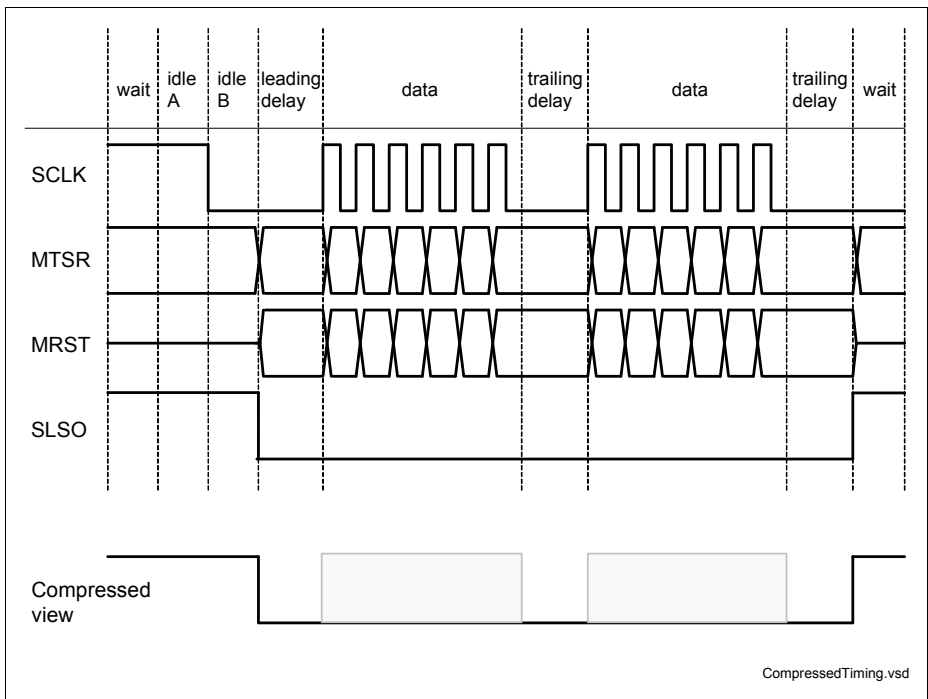


Figure 20-8 Phases of a QSPI Frame (Example for Data Length of 5 Bits)

Queued Synchronous Peripheral Interface (QSPI)

The flexible timing control of the QSPI allows to program each phase in a very wide time range, with sufficient precision. Some examples of QSPI frames, in compressed view, with different lengths of the phases, are shown in **Figure 20-9**.

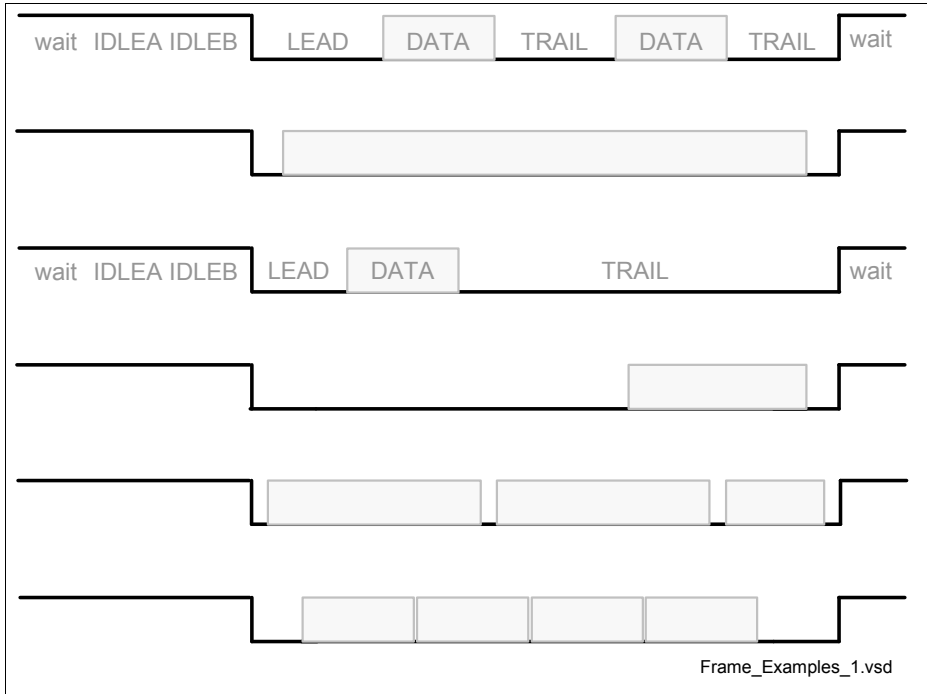


Figure 20-9 Examples of QSPI Frames in Compressed View

The length of the phases is programmed using the corresponding bit fields in the register **BACON**. There are four main bit fields defining the length of the phases in t_Q units and several modifier bit fields defining their range and granularity:

- IDLE
- LEAD
- DATA
- TRAIL

The WAIT phase is simply a loop waiting for a write to the Tx FIFO, without predefined duration, and consequently without a defining bit field.

The IDLE bit field defines two sub-phases with the same length IDLEA and IDLEB. At the transition between them the polarity of the serial clock signal for the next slave is set / changed.

Queued Synchronous Peripheral Interface (QSPI)

In addition, the flexible timing control allows to program the duty cycle and the sampling point properties of the serial clock. This allows to improve to a certain extent the achievable baud rate taking into account the clock asymmetries and the loop-delay.

The **ECON** bit fields A, B and C define the length of the corresponding bit segments.

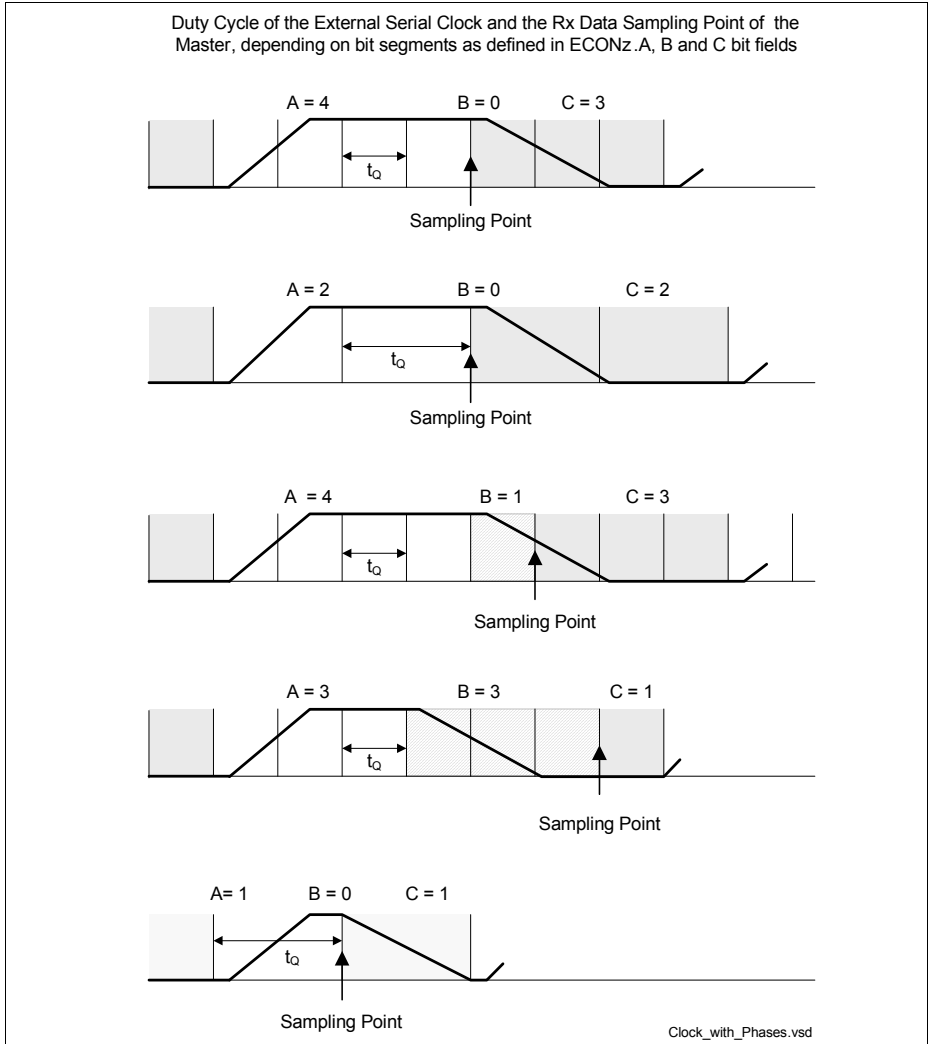


Figure 20-10 Control of the Duty Cycle and the Sampling Point of the Serial Clock

Queued Synchronous Peripheral Interface (QSPI)

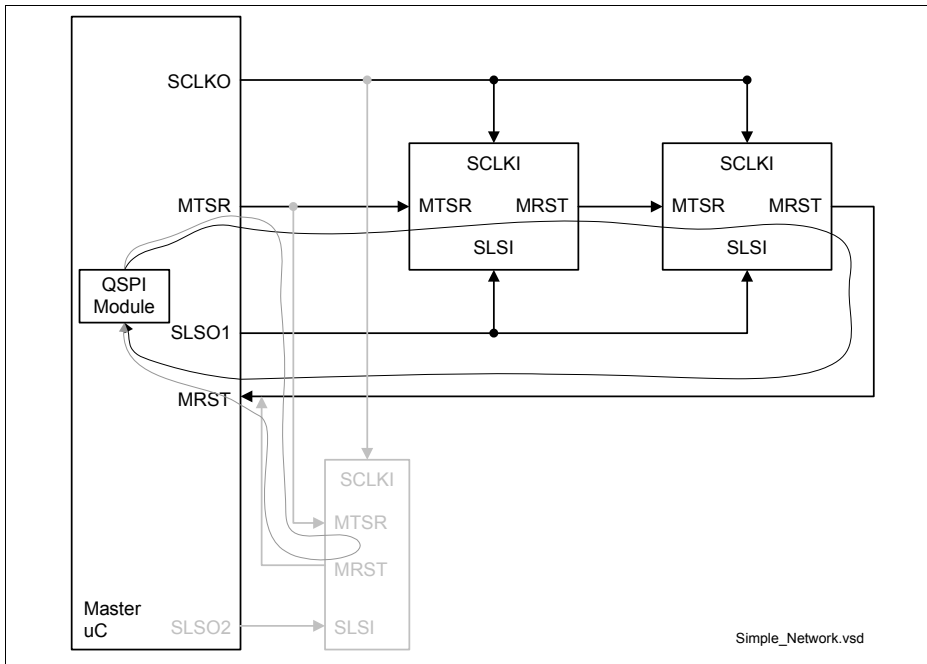


Figure 20-11 Closed-loop delays in a simple network consisting of one daisy-chain and one point to point connection in parallel

The purpose of the duty cycle control is:

- to compensate the influence of the path and pad asymmetry on the duty cycle
- to compensate the considerable asymmetry of the edges in case of half-duplex connection implemented as open-drain driver with pull-up resistor
- to adjust the duty cycle optimally to the timing properties of the master and slave, and to the properties of the connection (distance and capacitive load)
- to help achieve optimal baud rate, especially at higher baud rates which need an odd division factor (for example $5 = 3 \text{ high time} + 2 \text{ low time}$ or $5 = 2 \text{ high time} + 3 \text{ low time}$)

The purpose of the sampling point control is to allow reliable reception of the data transmitted by the slave, at as high as possible baud rates.

In master receive direction the slave transmitted data reaches the master considerably late relative to the shift edge of the serial clock. This is caused by the loop delay consisting of:

- the traveling time of the shift clock edge from the master to the slave
- the reaction time of the slave
- the traveling time of the data from the slave to the master

Queued Synchronous Peripheral Interface (QSPI)

Last but not least, the flexible timing control allows to control the phase and the polarity of the shift clock.

The clock polarity is configured by bit field **ECONz.CPOL**. This bit field sets the clock idle polarity to low or high.

The clock phase is configured by bit field **ECONz.CPH**. This bit field sets the initial clock delay to 0 or Bit Segment A, as defined by the bit field **ECONz.A**.

The CPOL and CPH settings control the master mode. The slave mode timing behavior is fixed to CPOL=0 and CPH=1, and these values must be programmed in the **ECON** register pointed by **BACON.CS**.

The **Figure 20-12** shows the QSPI timings for clock phase CPH = 0.

ECON.CPH = 0 is used when communicating with a slave that delivers a valid value of the first bit immediately after being selected with an SLS signal. In such a case the first clock edge in a frame is used to latch the first bit. The second edge delivers the second bit value, and so on. The last edge in a frame delivers a bit with don't care value.

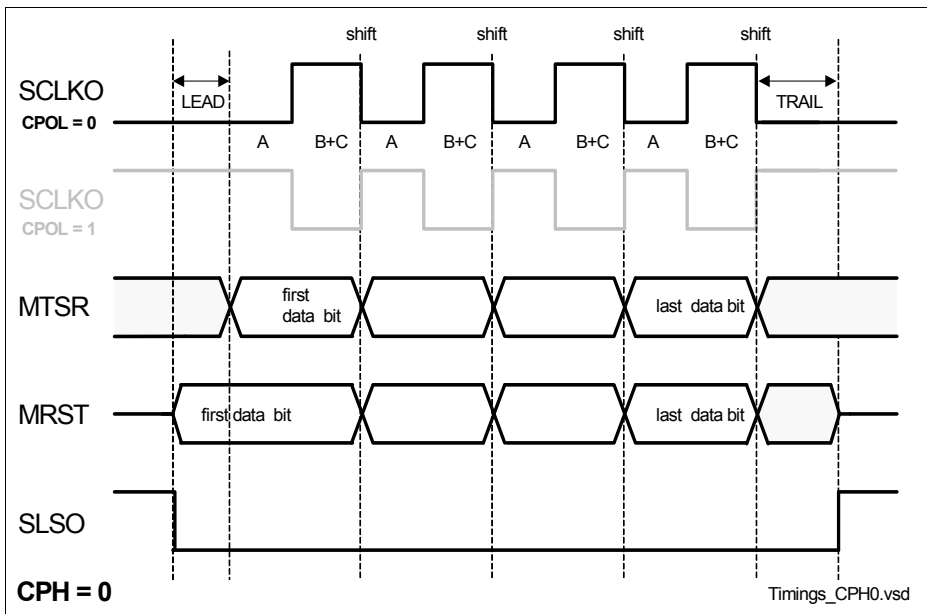


Figure 20-12 Timing of a transfer with CPH = 0 (Example for Data Length of 4 Bits)

The trailing delay starts after the last shift clock period of a data block and is followed either by the deactivating edge of SLSO, or a new data block in continuous mode.

Attention: If CPH = 0, then the total delay between the SLSO activating edge and the first edge of the serial clock SCLKO is LEAD + **ECONz.A**

Queued Synchronous Peripheral Interface (QSPI)

Figure 20-13 shows the QSPI timings for clock phase CPH = 1.

ECON.CPH = 1 is used when communicating with a slave that delivers a bit with a random or don't care value immediately after being selected with the SLSO signal. In such a case the first clock edge in a frame is used by the slave to shift out the first valid data bit. The second edge latches this bit value, and so on. The last edge in a frame is used to latch the last valid data bit, which normally remains driven until the end of the frame.

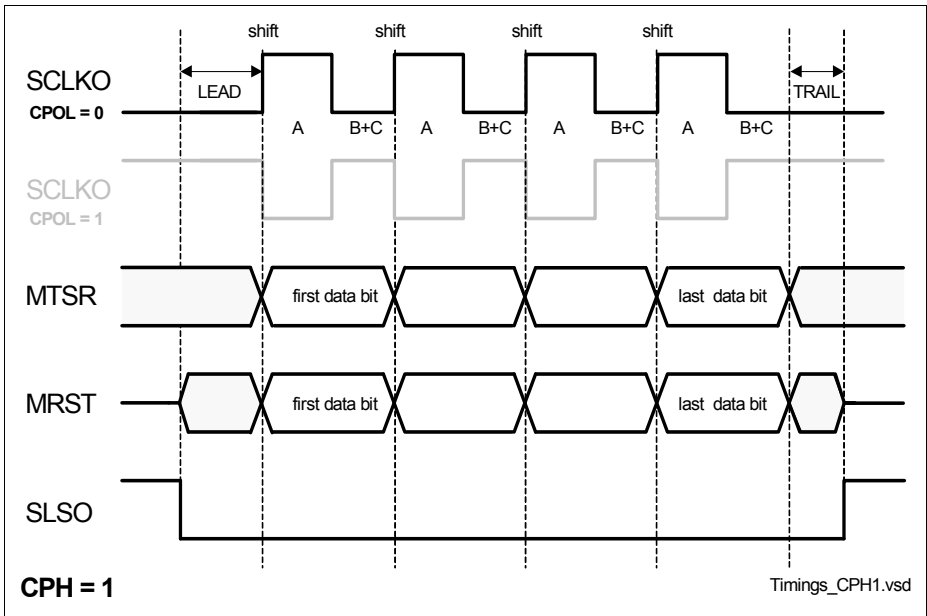


Figure 20-13 Timing of a transfer with CPH = 1 (Example for Data Length of 4 Bits)

The leading delay lasts between the active edge of the SLSO and the first shift clock edge.

Attention: If CPH = 1, then the total delay between the last edge of the SCLKO and the SLSO deactivating edge is $TRAIL + ECONz.B + ECONz.C$

Queued Synchronous Peripheral Interface (QSPI)

20.5.2 Configuration Extensions

This section describes the possibilities available for configuring the phases of the QSPI communication: timing delays, data lengths, their ranges and granularity.

The QSPI module controls 16 communication channels, which are individually programmable. Each channel is associated to one slave select output and to the corresponding timing and other properties (baud rate, delays, data width, parity...).

The 16 slave select channels of the QSPI module are subdivided in two groups of 8 slave select channels:

- Channels 0 to 7
- Channels 8 to 15, channel 8 having the same **ECON** settings as channel 0, channel 9 as channel 1, and so on.

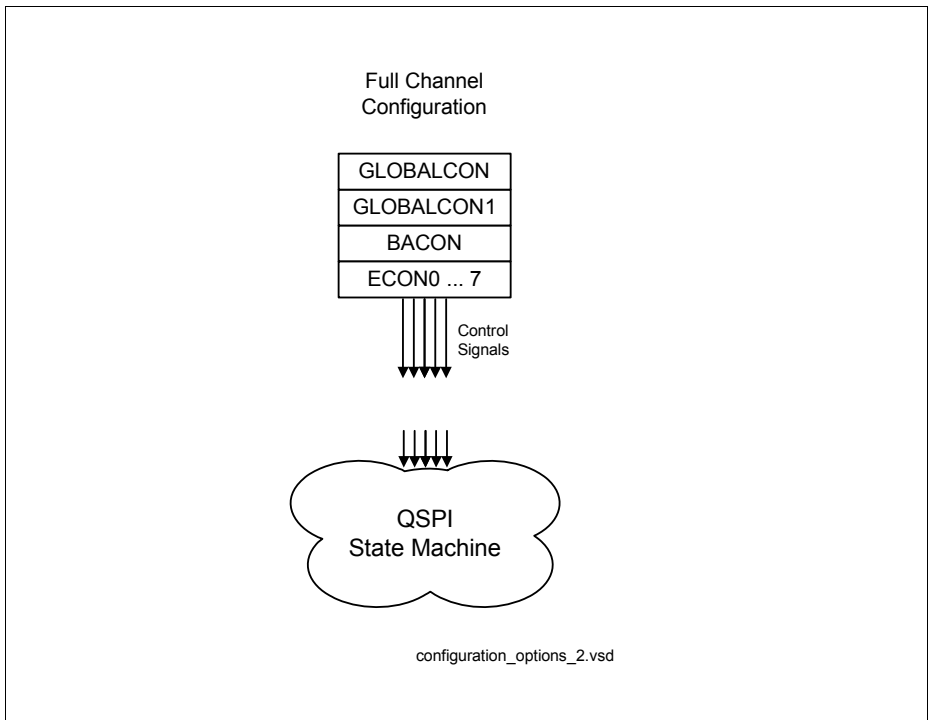


Figure 20-14 Flexible Configuration Concept

Queued Synchronous Peripheral Interface (QSPI)

20.5.3 Details of the Baud Rate and Phase Duration Control

The QSPI phases correspond to states of a simple state machine defining a sequence of simple states and loops. Due to the strict sequential nature of the communication cycle one timer is enough to define the length of each phase. The length of each phase is defined in time quanta, and the length of a time quantum is defined by a time quantum timer. Each QSPI slave select channel has its own time quantum length, based on the module time quantum length:

T_{BAUD2} clock period → one module time quantum → one channel time quantum

Each QSPI module generates a module time quantum which is further downscaled to a channel quantum. This affects both the delays and the baud rates. The assumption is that the pad strength setting, the capacitive load and the corresponding QSPI network layout properties do not allow extreme differences in the baud rates for the different slaves connected to one module. They all have similar speed range: high speed, or medium speed, or low speed. Mixing very high speed and very low speed slaves does not make much sense. Many slaves produce high accumulated capacitive load on the clock and data outputs, which influences the possible baud rates for all slaves. Variations in the baud rates of the slaves of one module in the range of 6:1 is possible by varying the bit segment phases in each channel. Additional two-bit counter per slave allows for additional scaling of 1, 2, 4, or 8, making a total bit-time variation of 48:1 between the channels in one module possible.

Each QSPI slave select channel has its own set of phase lengths, depending on $T_{\text{BAUD2}} = 1 / f_{\text{BAUD2}}$

- $T_{\text{SCLKz}} = T_{\text{BAUD2}} * \text{GLOBALCON.TQ} * \text{ECONz.Q} * (A + B + C)$
- $T_{\text{LEAD}} = T_{\text{BAUD2}} * \text{BACON.LPRE} * \text{BACON.LEAD}$
- $T_{\text{TRAIL}} = T_{\text{BAUD2}} * \text{BACON.TPRE} * \text{BACON.TRAIL}$
- $T_{\text{IDLEA,B}} = T_{\text{BAUD2}} * \text{BACON.IPRE} * \text{BACON.IDLE}$
- $T_{\text{STROBE}} = T_{\text{BAUD2}} * \text{GLOBALCON.TQ} * \text{ECONz.Q} * \text{GLOBALCON.STROBE}$

Note: The equations above contain the representative decimal values of the bit-fields in corresponding units, not their actual binary bit-value. For the formulas using the actual binary values of the bit-fields, see [Figure 20-16](#) and [Figure 20-17](#).

Queued Synchronous Peripheral Interface (QSPI)

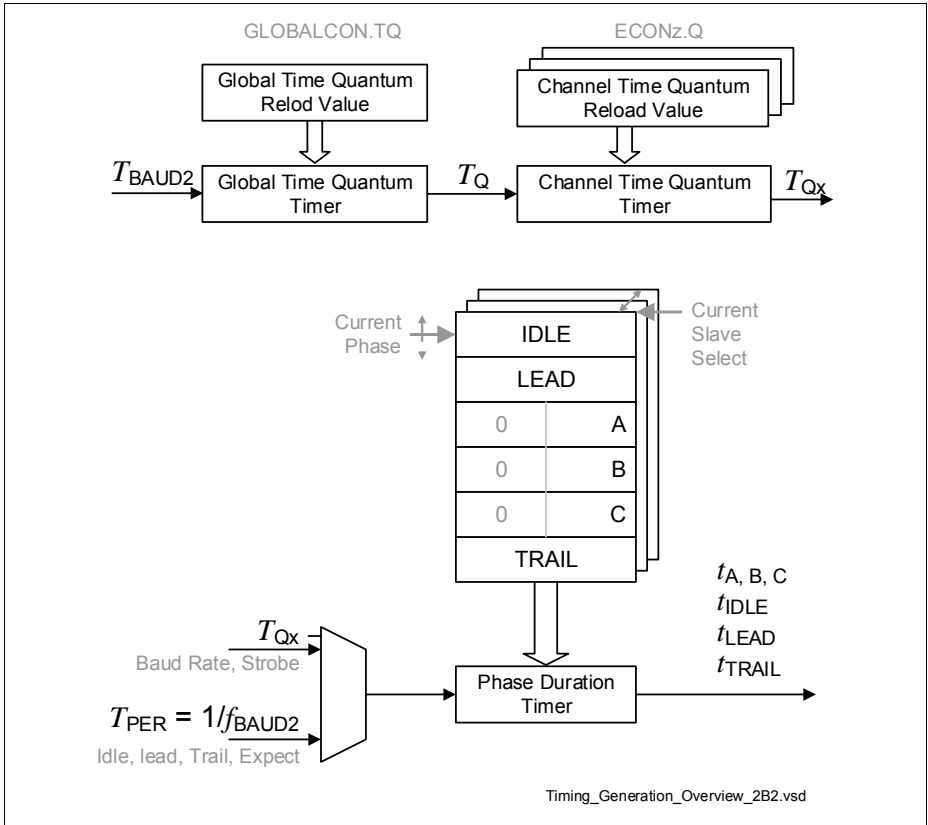


Figure 20-15 Phase Duration Control, Overview

Queued Synchronous Peripheral Interface (QSPI)

20.5.4 Calculation of the Baud Rates and the Delays

This section gives another alternative view to the calculation of the baud rate and the duration of the delays.

The baud rate generation chain for a channel starts with a common TQ divider. The resulting global time quantum is used by the dedicated Q and A,B,C dividers to generate the baud rates per channel (slave select). See [Figure 20-16](#).

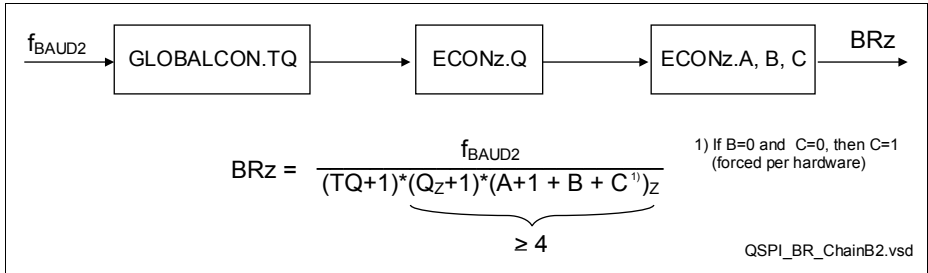


Figure 20-16 Baud Rate Calculation

Attention: The following condition must be satisfied: $(Q+1) \cdot (A+1 + B + C) \geq 4$. Or alternatively formulated: bit length must be at least $4 \cdot T_Q$.

Each delay of an active channel (slave select) has a separate and independent divider chain starting with a power-of-four prescaler and an n-divider. See [Figure 20-17](#).

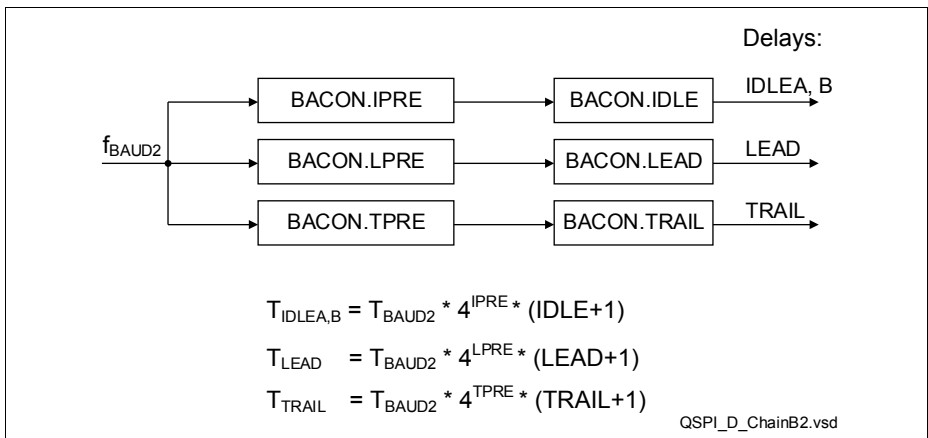


Figure 20-17 Calculation of the delays

Queued Synchronous Peripheral Interface (QSPI)

20.5.5 State Diagram of Standard Communication Cycle

Figure 20-18 shows the top view of the standard communication cycle of the QSPI state machine in a hierarchical way. The “Data” state consists of a loop of several “Bit” states, where the number of repetitions is defined with the data width bit field. Each “Bit” state consists of a simple sequence of “A”, “B”, and “C” states.

A complete (or standard) communication cycle containing all phases, starting with activating the slave select and ending with its deactivation, is executed only if the bit **BACON.LAST = 1**. Otherwise, the slave select remains active and the module waits for more FIFO data entries.

The Phase Transition Interrupt PT (events PTI1 and PTI2) signal one out of all phase transitions which are shown in **Figure 20-18**, selected by the bit fields **GLOBALCON.PT1** and **PT2**.

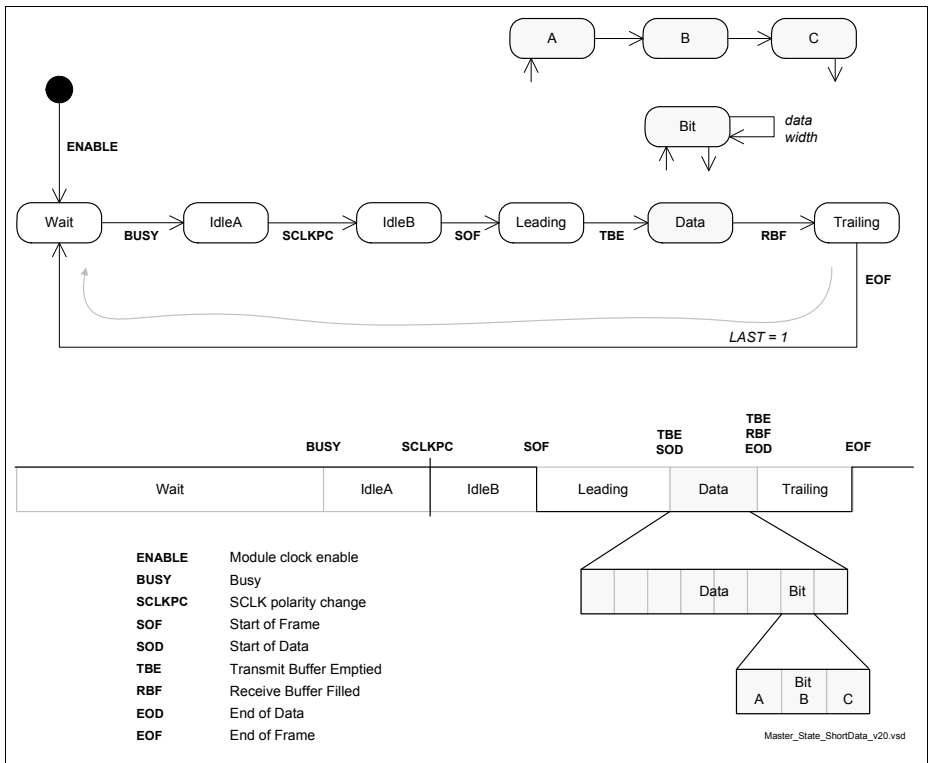


Figure 20-18 Standard Communication Cycle of the State Machine (Short Data)

(>>Interrupts)

Queued Synchronous Peripheral Interface (QSPI)

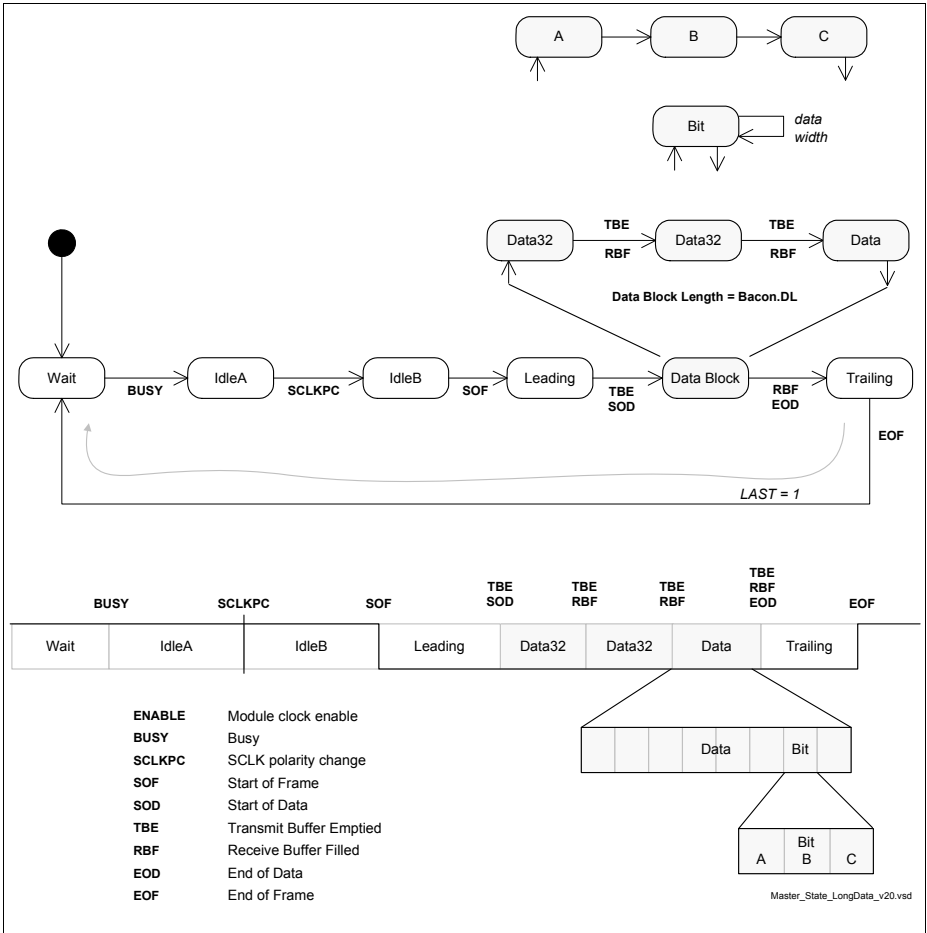


Figure 20-19 Standard Communication Cycle of the State Machine (Long Data)

Queued Synchronous Peripheral Interface (QSPI)

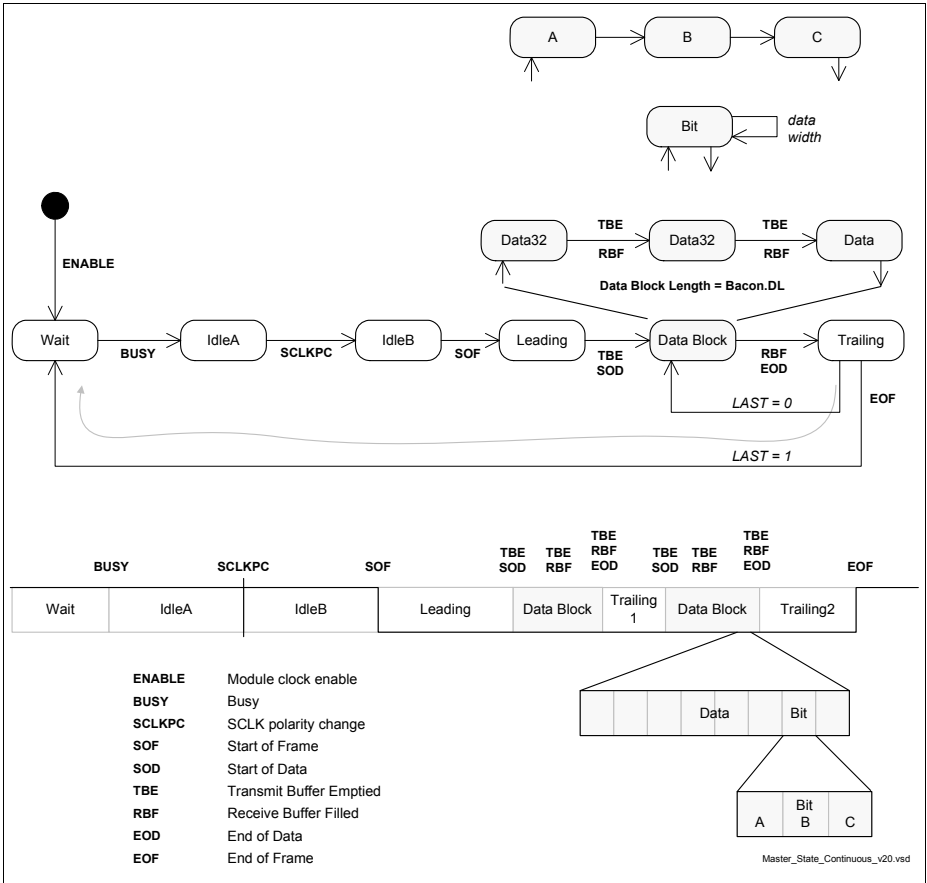


Figure 20-20 Standard Communication Cycle of the State Machine (Continuous)

Queued Synchronous Peripheral Interface (QSPI)

20.5.6 Expect Phase

Figure 20-21 shows the EXPECT state in the QSPI state machine diagram. This state can be entered in the long data mode and in the continuous mode, where one frame consists of more than one TXFIFO entries. Its purpose is to provide monitoring of the latencies on the FPI bus. In this case, after one FIFO entry has been shifted out and the module expects more bits to shift, according to the BACON.DL, a time-out counting starts. If the expected FIFO entry does not come in time, a time-out occurs.

The duration of the EXPECT state is zero if at the end of the current DATA state the new data is already available in the FIFO.

The duration of the EXPECT state can be anywhere between zero and the programmed upper limit, if the next data is written into the TXFIFO any time within the allowed time window. In this case the current frame continues with the SLSO signal remaining active.

The duration of the EXPECT state is exactly maximum if the data did not come in time. Then the current frame is stopped, but the SLSO is not deactivated. A TO (Time - Out) interrupt is raised. The state machine continues the EXPECT state in a loop, waiting for the software intervention, generating an interrupt signal after each expect time interval.

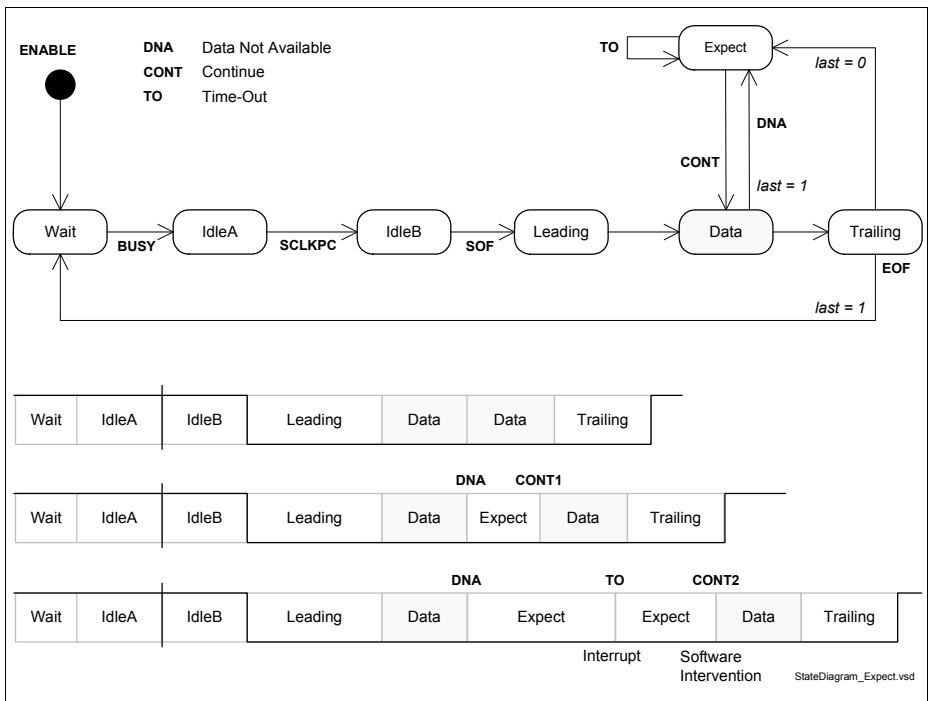


Figure 20-21 EXPECT Cycles of the QSPI State Machine

Queued Synchronous Peripheral Interface (QSPI)

20.5.7 External Slave Select Expansion

In this mode the bit field **BACON**.CS drives directly the SLSO1 to SLSO4 signals, not demultiplexed by the QSPI, but chip-externally. **SSOC**.AOL bits can invert the SLSO0...4 signal levels individually; **SSOC**.OEN bits enable them. See also **GLOBALCON**.DEL0.

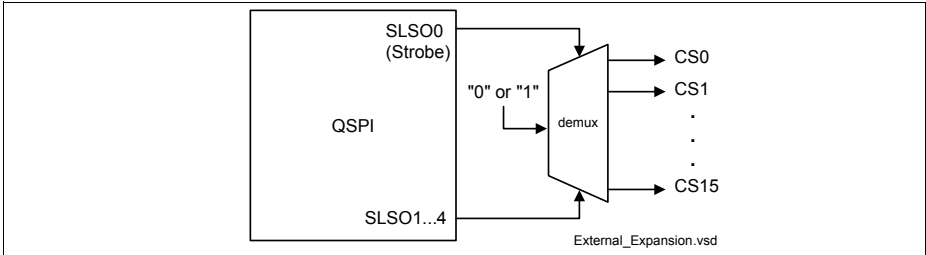


Figure 20-22 Demultiplexing the Slave Select Signals Externally

In order to ensure glitch free selection, a strobe signal is provided, driven at SLSO0 pin. This signal is delayed relative to the SLSO1...4 signals for LS (Lead Strobe) and TS (Trail Strobe) delays, equal in duration and configurable in the range of 1 to 16* f_{BAUD2}. The duration of the LS and TS delays is set in the **GLOBALCON**.STROBE.

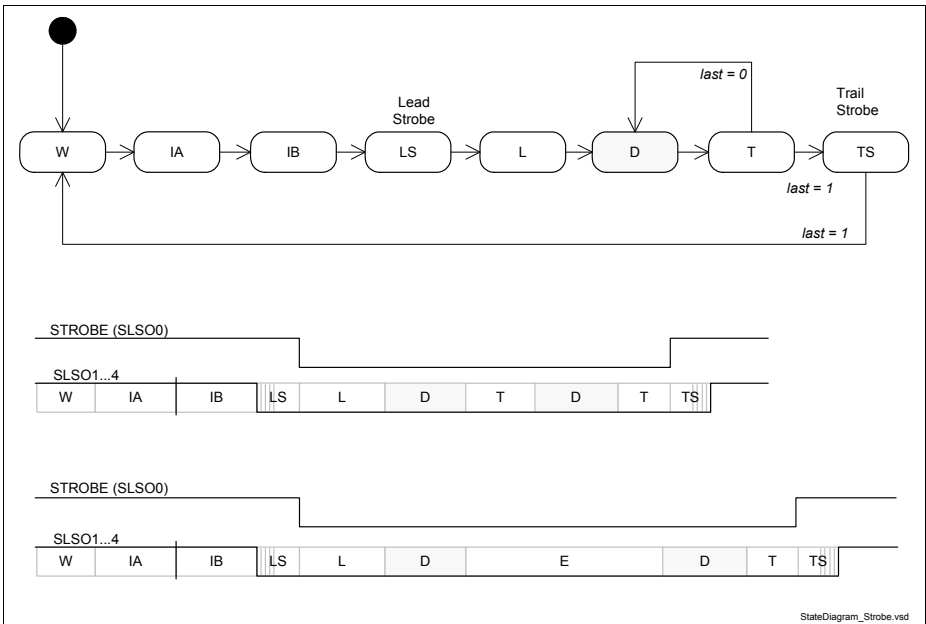


Figure 20-23 State Machine and SLS Signals in Strobed Mode

Queued Synchronous Peripheral Interface (QSPI)

20.6 User Interface

This section describes the possibilities available for transferring data between the on-chip RAM memories and the QSPI module using the transmit and receive FIFOs. The QSPI features one 4 x 32-bit Tx FIFO and one 4 x 32-bit Rx FIFO. The address ranges for writing data and configuration to the Tx FIFO and for reading data and configuration from the Rx FIFO consist of address locations with special properties:

- Tx FIFO
 - DATA_ENTRY - writes to any of these eight functionally identical locations are always interpreted as data
 - BACON_ENTRY - writes to this location are always interpreted as configuration
 - MIX_ENTRY - writes to this location are interpreted as data or configuration, based on a set of rules
- Rx FIFO
 - RX_EXIT - reads from this location deliver either data or data and status, based on a set of rules.

Queued Synchronous Peripheral Interface (QSPI)

20.6.1 Transmit and Receive FIFOs

The Tx_FIFO is filled with both configuration and data entries, which are distributed to the **BACON** and the shift register at the exit of the Tx_FIFO. The Tx_FIFO keeps track if an entry is configuration or data. The control and the data elements are automatically distributed to the **BACON** and shift register. If no data is available, the new **BACON** is pending.

If a frame is finished and there is data in the FIFO

- in short mode, a new frame starts automatically with the same configuration as the last one.
- in long mode, the extra data are ignored.

TX_FIFO error conditions:

- FPI bus write to a full TX_FIFO generates an overflow interrupt. The write is ignored.
- In case of slave mode, (**GLOBALCON.MS** = 1X), in case of an underflow, only "1" are delivered.

RX_FIFO error conditions:

- FPI bus read from an empty RX_FIFO generates an underflow interrupt, and delivers only "1" bits.
- Hardware attempt to write a full RX_FIFO with data or status generates an overflow interrupt. The write attempt is ignored by the RX_FIFO.
- If the **GLOBALCON.SRF** bit is set, the communication is stopped in case of full FIFO in master mode. In slave mode, the bit is ignored, no stop can be done.
- If the FPI bus frequency is very slow, and the baud rate very high, it can happen that some data is lost on the way between the shift register and a not full RX_FIFO. In such a case, an overflow interrupt is generated, although the RX_FIFO is not full.

The Rx_FIFO can be filled with both status and received data entries, depending on **GLOBALCON.SI**.

Note: If a 8-bit or 16-bit write is executed to the TX_FIFO entry addresses, the data is mapped 1:1 to its location within the 32-bit wide TX_FIFO entry, and the rest is padded with zeros.

Queued Synchronous Peripheral Interface (QSPI)

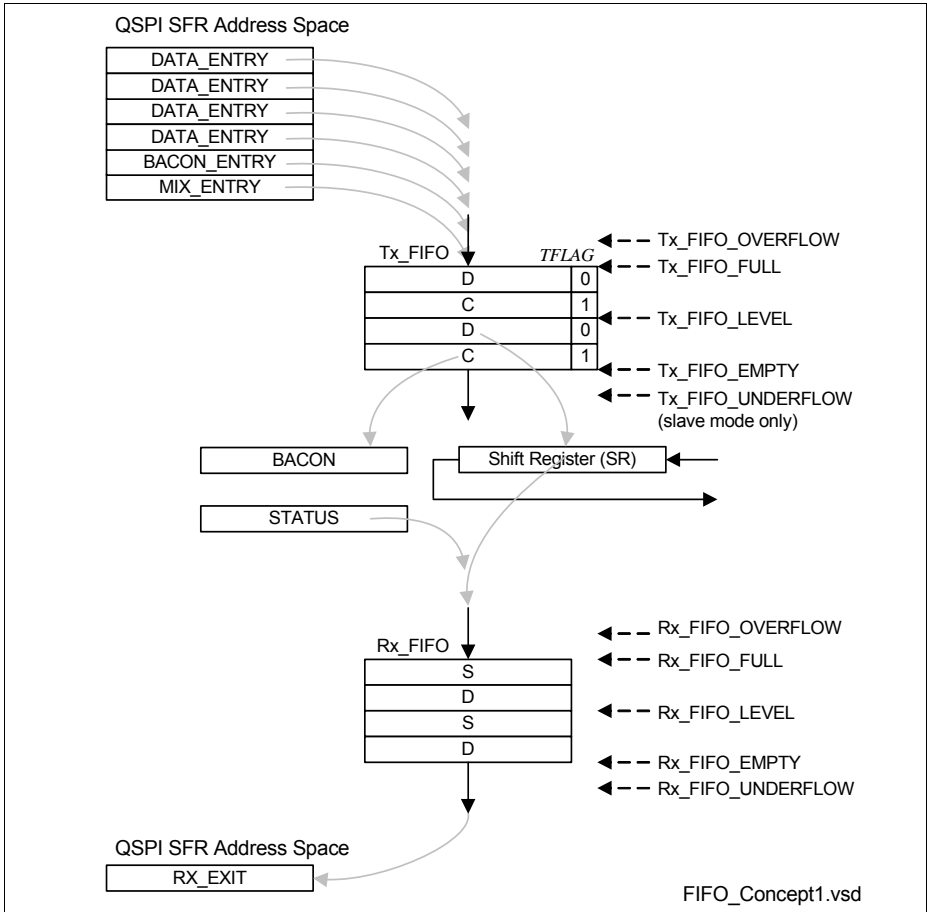


Figure 20-24 Architecture of the Tx and Rx FIFOs

(>>Interrupts)

Queued Synchronous Peripheral Interface (QSPI)

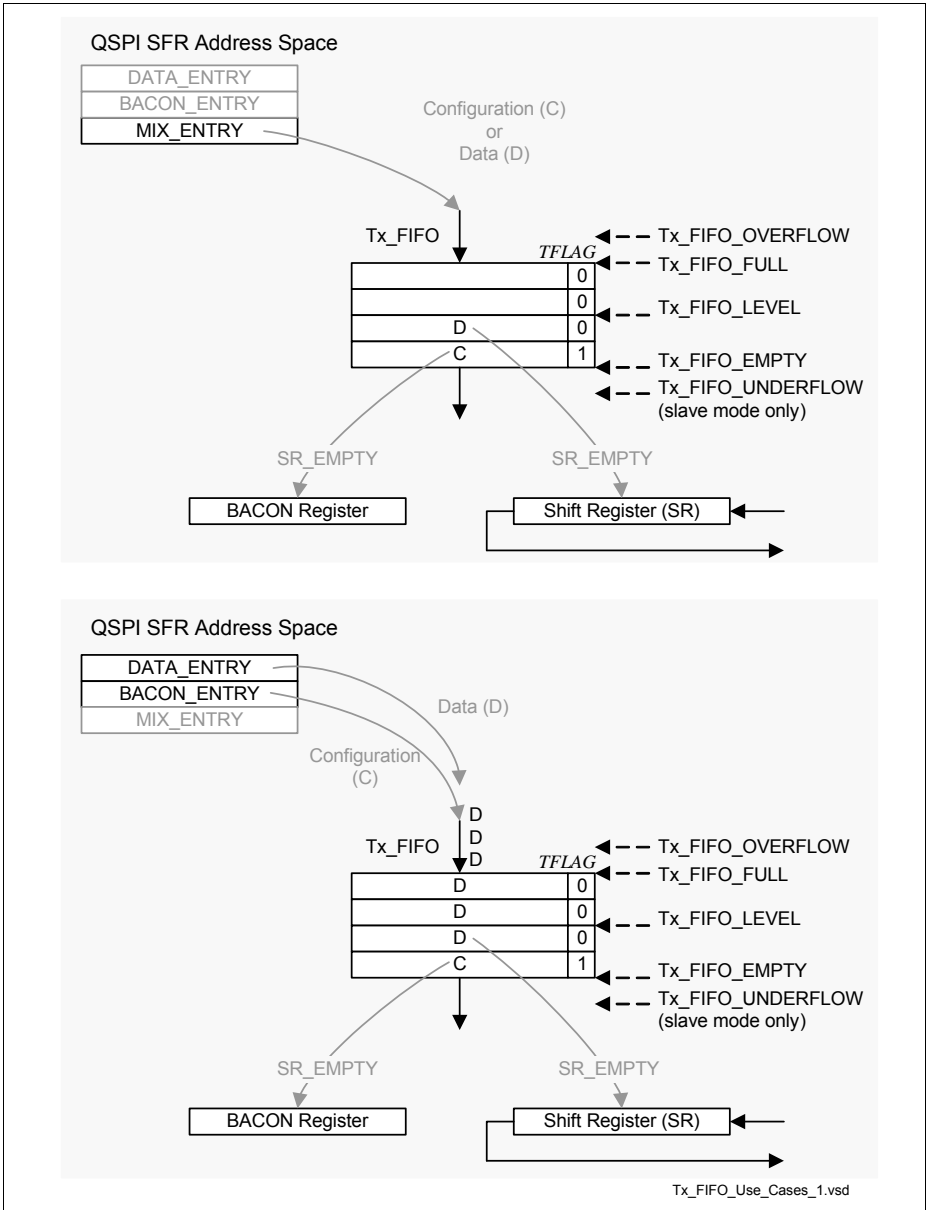


Figure 20-25 Using Tx FIFO Mix and Bacon / Data Entries

Queued Synchronous Peripheral Interface (QSPI)

20.6.1.1 Short Data Mode

In Short Data Mode, the QSPI module transmits single data with a length of 2 to 32 bits in one frame. This mode is defined by **BACON.LAST** = 1 and **BACON.BYTE** = 0.

The transfer cycle is a sequence of the following phases: W?_I_L_D_T. The symbol “?” means that the phase is optional (duration of 0 allowed).

For example, a Tx_FIFO event can be used to trigger one DMA transfer (consisting of two 32-bit moves) to transfer both the BACON and DATA entries from an on-chip RAM memory to the QSPI module.

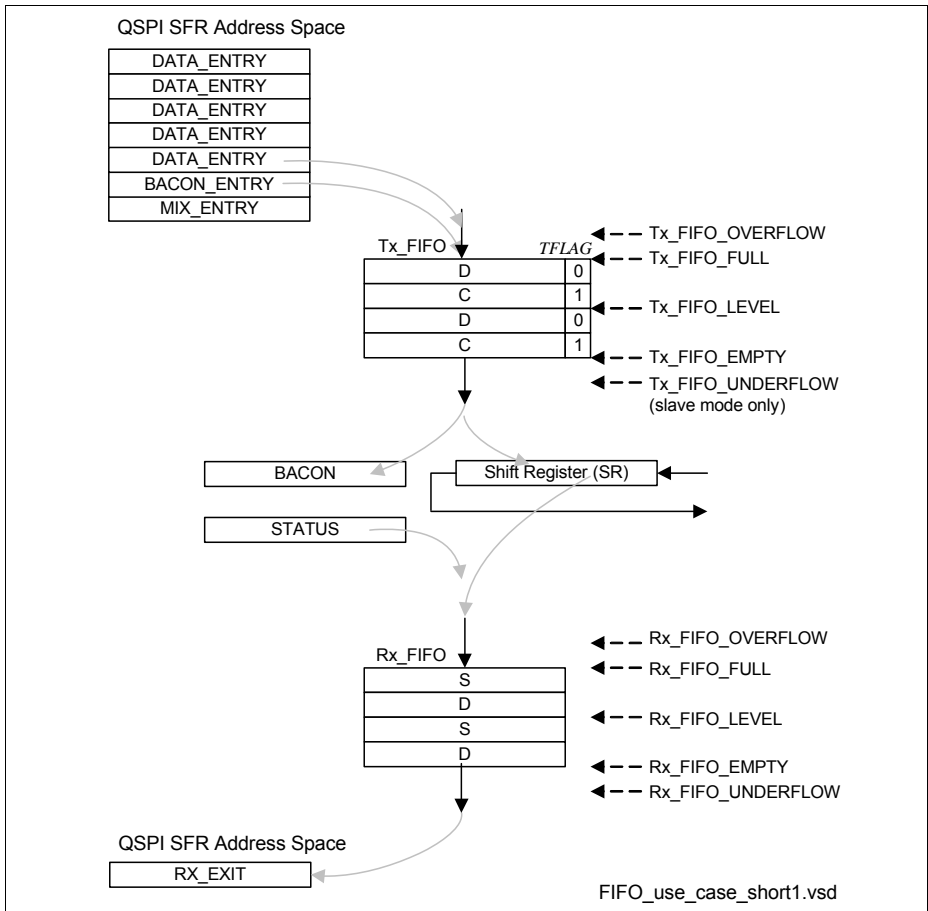


Figure 20-26 User Interface in Short Data Mode

Queued Synchronous Peripheral Interface (QSPI)

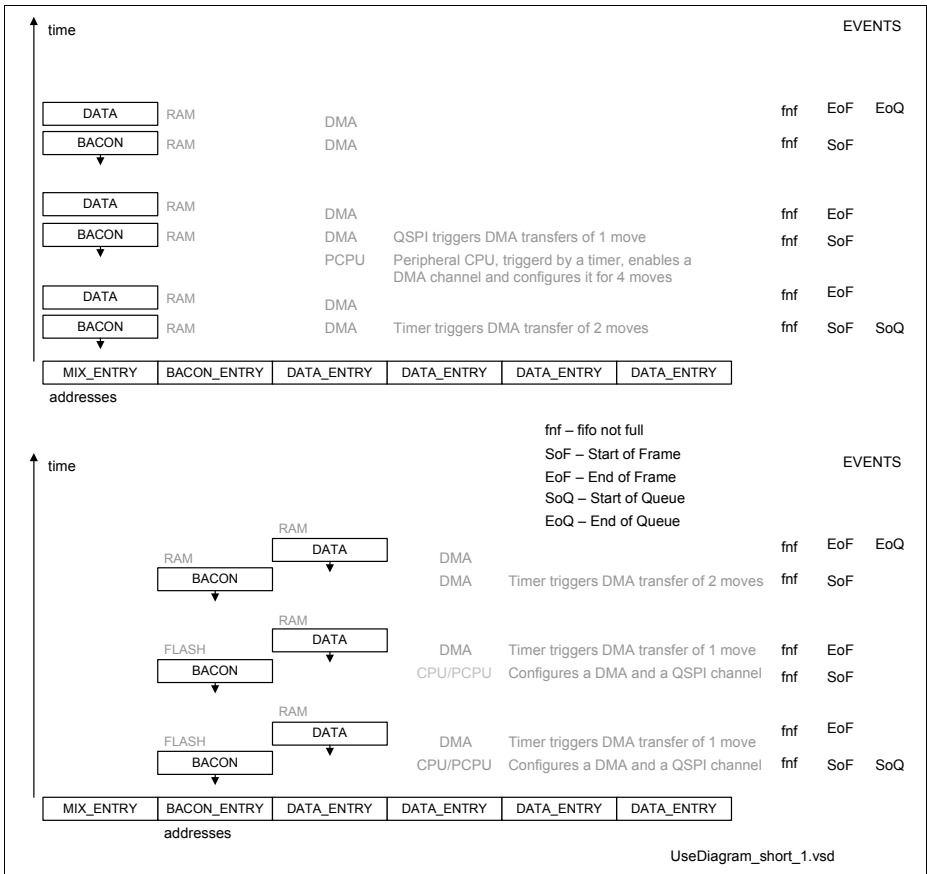


Figure 20-27 Use Diagram of the User Interface in Short Data Mode

Queued Synchronous Peripheral Interface (QSPI)

20.6.1.2 Long Data Mode

In Long Data Mode, the QSPI module transmits bursts of up to 32 bytes (256 bits) in one frame. This mode is defined by programming **BACON.LAST** = 1 and **BACON.BYTE** = 1. One transfer cycle in long data mode is a sequence of the following phases: W?_I_L_D_T. The W_I_L_D_T_I_L_D_T sequence is possible. The symbol “?” indicates that the phase is optional (duration of 0 allowed). Each “D” indicates a data length as defined with **BACON.DL** and **BACON.BYTE**, that is, up to 256 bits. One “D” is made of a number of 32-bit TXFIFO entries “d” and one rest at the end “b”, which can be written as D=dddddb.

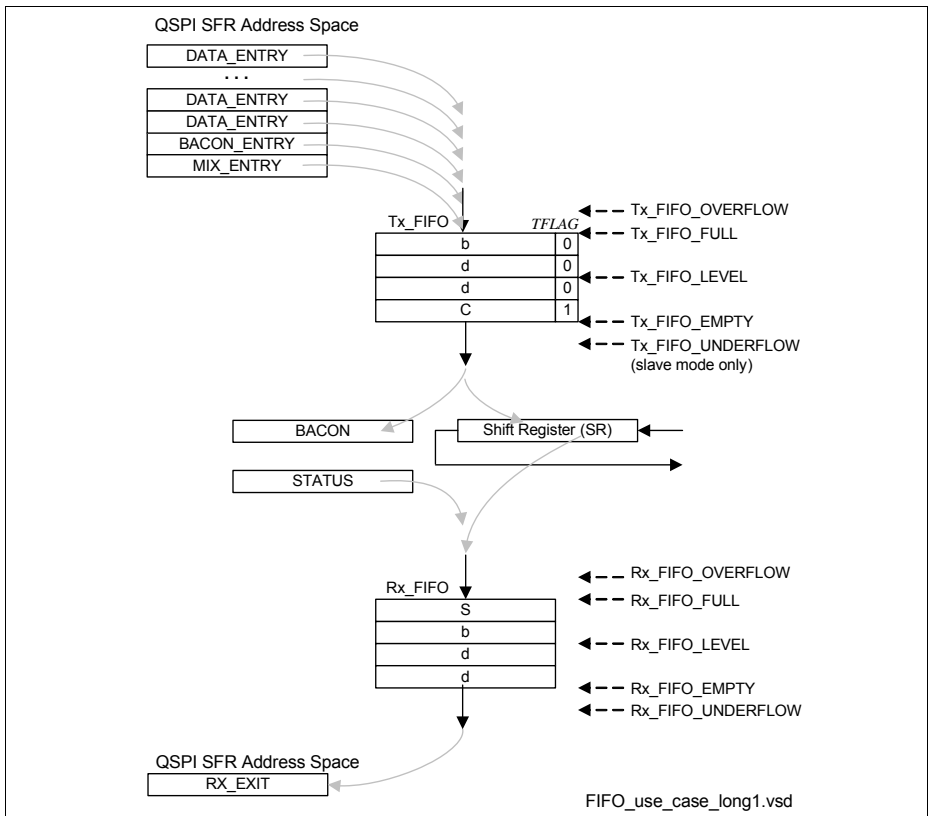


Figure 20-28 User Interface in Long Data Mode

In a DMA transfer example, a Tx_FIFO_EMPTY event can be used to trigger one DMA transfer (consisting of, for example, four 32-bit moves) to transfer the BACON and three DATA entries from RAM memory to the QSPI module. The three data words could

Queued Synchronous Peripheral Interface (QSPI)

contain 80 bits (10 bytes) of data. The three data words could also contain 40 bits (5 bytes), 64 bits (8 bytes) plus one dummy word. In this way short bursts of up to 96 bits can be transmitted.

In such 4 words DMA transfer example, the Tx_FIFO event can be used to transfer two 32-bit words per transfer.

If a configuration word is written when a data word is expected, the configuration word is ignored, and an Unexpected Configuration Error is raised.

Queued Synchronous Peripheral Interface (QSPI)

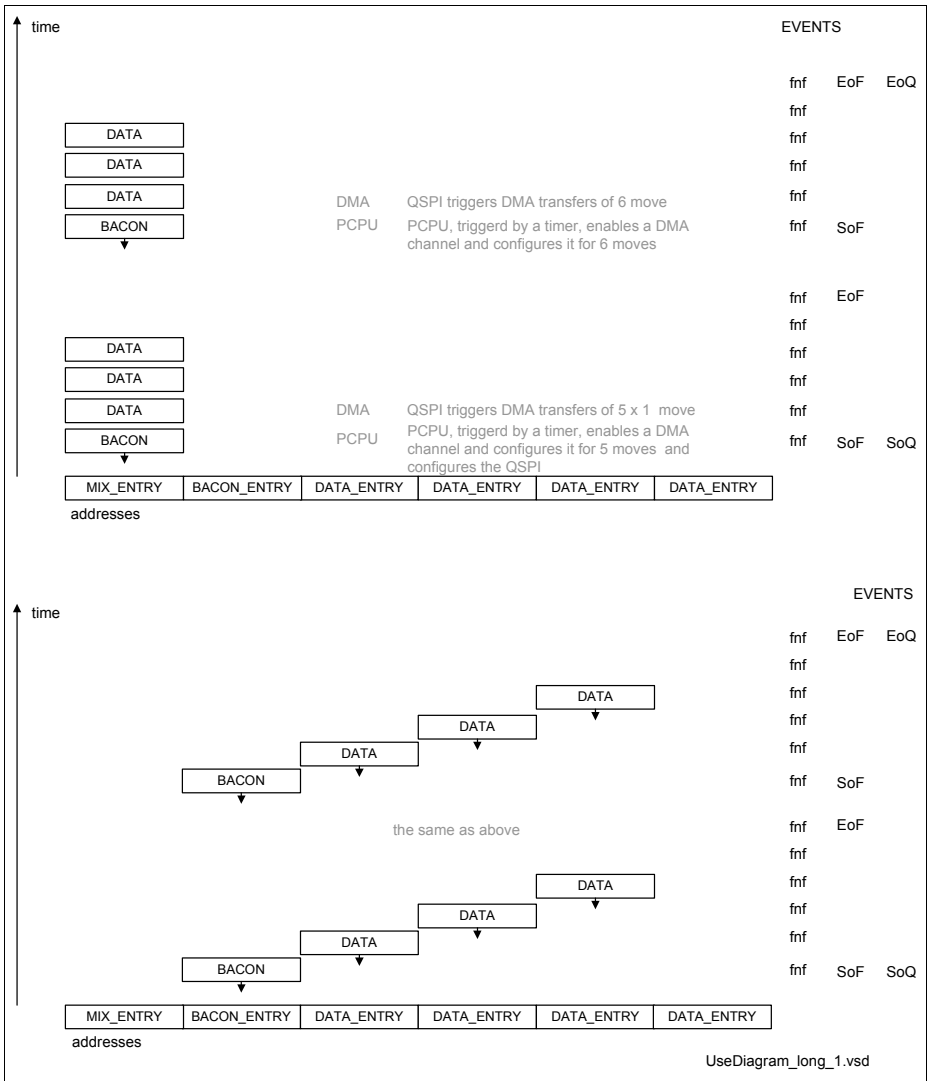


Figure 20-29 Use Diagram of the User Interface in Long Data Mode

Queued Synchronous Peripheral Interface (QSPI)

20.6.1.3 Continuous Data Mode

In Continuous Data Mode, the QSPI transmits a stream of data with an arbitrary length with an active slave select signal. It can be used with both short and long data.

This mode starts by programming a control word containing **BACON**.LAST = 0 and the first data. The communication continues with the subsequent data entries written. The mode ends with a control word containing **BACON**.LAST = 1 and the last data. If LAST = 0, then TRAIL is a delay between data blocks. The W_I_L_D_T_D_T_D_T sequence appears. Each "D" represents data block as defined with **BACON**.DL and **BACON**.BYTE (up to 256 bits). If LAST = 1 then TRAIL is trailing delay, see [Figure 20-9](#).

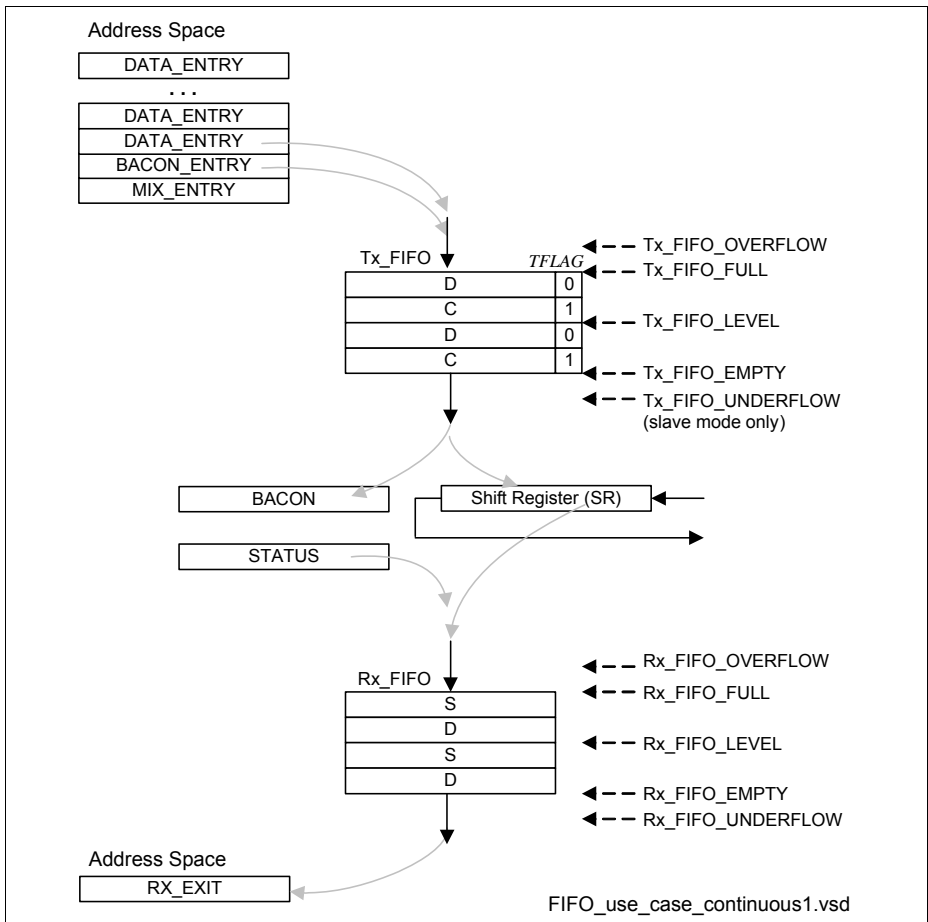


Figure 20-30 User Interface in Continuous Data Mode

Queued Synchronous Peripheral Interface (QSPI)

The **Figure 20-31** shows an example of continuous data mode using short data. Here, the CPU starts the stream by writing BACON to the BACON_ENTRY address, and afterwards a DMA transfers the whole stream except the last BACON and DATA.

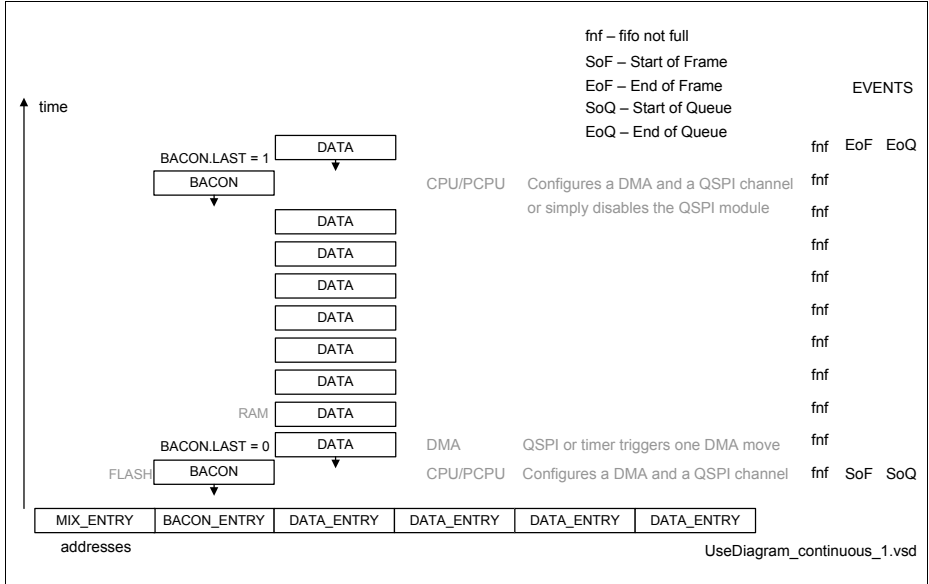


Figure 20-31 Use Diagram of Continuous Data Mode with Short Data

Queued Synchronous Peripheral Interface (QSPI)

The **Figure 20-32** shows an example of continuous data mode using long data. Here, the first BACON must be repeated after each data block, but on the other hand, the whole transfer can be done using only the MIX entry and one single DMA channel, without CPU involvement. The loss of efficiency because of the unnecessary extra BACONS can be minimized by using long data blocks.

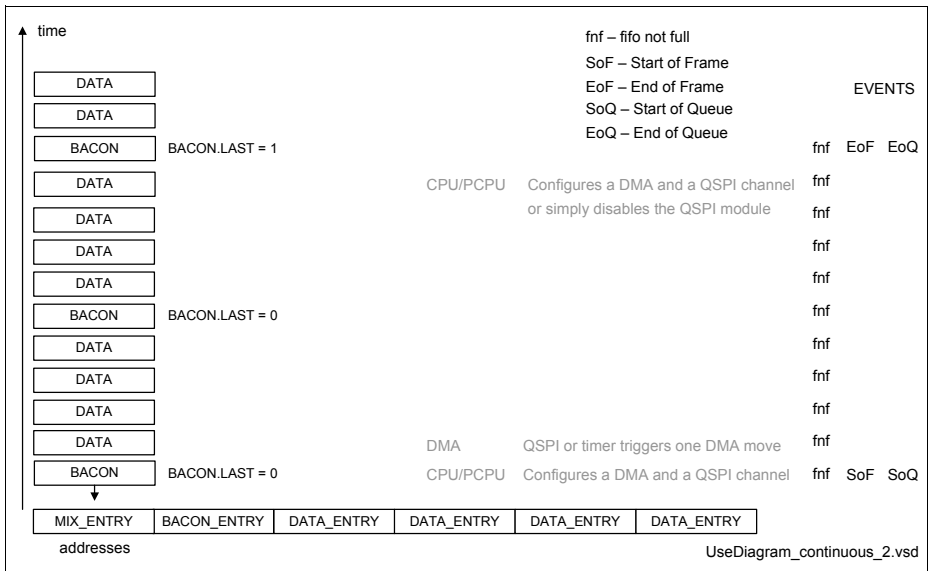


Figure 20-32 Use Diagram of Continuous Data Mode with Long Data

Queued Synchronous Peripheral Interface (QSPI)

20.6.1.4 Single Configuration - Multiple Frames Behavior

If longer sequences of data with length of up to 16 bit have to be sent to single slave in separate frames, then the DMA move for the repeating configuration word can be saved. An example of a TxFIFO contents and the resulting message sequence is shown in Figure 20-33.

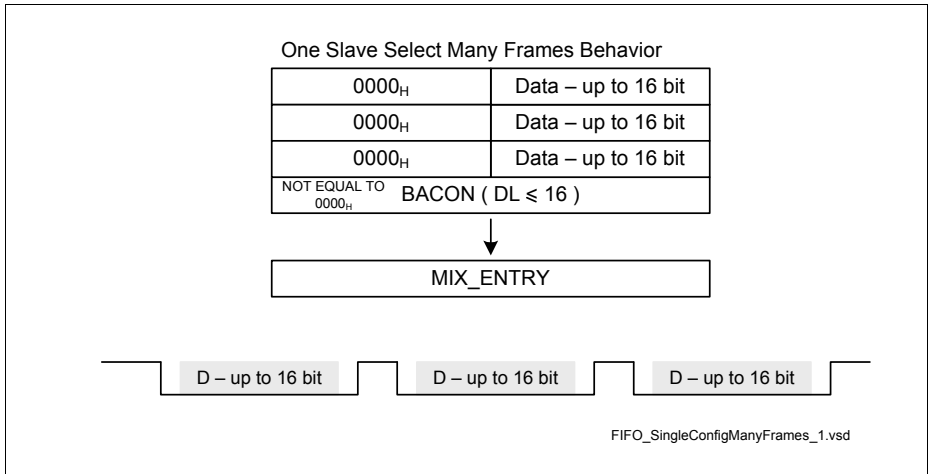


Figure 20-33 Multiple Frames to One Channel (Example of up to 16 Bit Data)

In order to support sending short multiple frames to single channel without having to send the same configuration each time, the TXFIFO follows this rule:

- If **BACON.DL** less or equal 15 (up to 16 bit data length)
- if **MIX_ENTRY** address is used
- then if the upper 16 bit of a 32-bit **MIX_ENTRY** are 0, the entry is considered data, else configuration. Note that there is no valid configuration word with upper 16 bits equal to 0.
- In such a case the TXFIFO marks the entry as data.

*Note: In case the **BACON.LAST** = 0, the slave select remains active between the frames. The behavior is equal to the continuous mode. The disadvantage of this method is that only up to 16 bits data can be transferred with one FPI bus move. The advantage is that continuous mode is possible using the mix entry.*

20.6.1.5 Big Endian Data Format

The bit-field **ECON.BE** activates the permutation of the data bytes written to the shift register and read from it to the big endian format and back.

ECON.BE allows for switching between big and little endian per slave (SLSO signal).

Queued Synchronous Peripheral Interface (QSPI)

See [Figure 20-34](#) and [Figure 20-35](#).

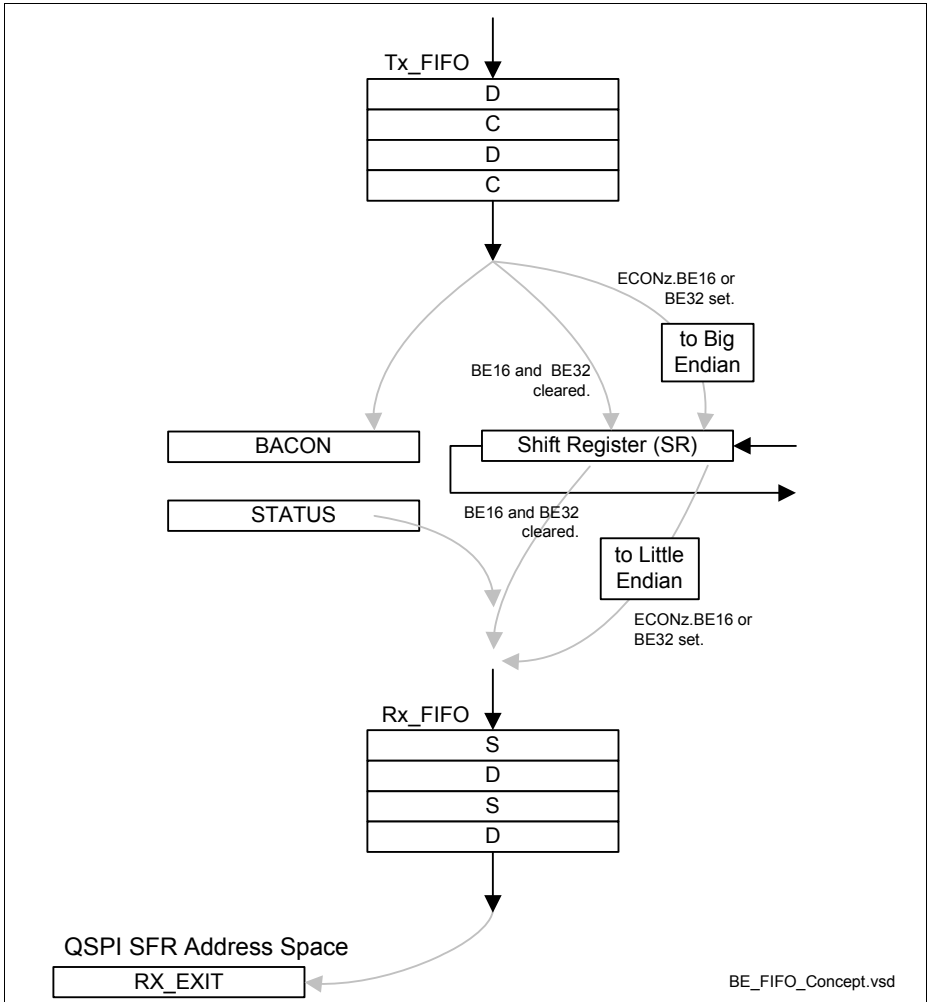


Figure 20-34 Big Endian Data Path

Converting data from little to big endian and other way around requires one and the same permutation of the data bytes, as shown in [Figure 20-34](#). One permutation block is located before and one after the shift register, so that only data can be permuted, and not **BACON** or **STATUS**.

Queued Synchronous Peripheral Interface (QSPI)

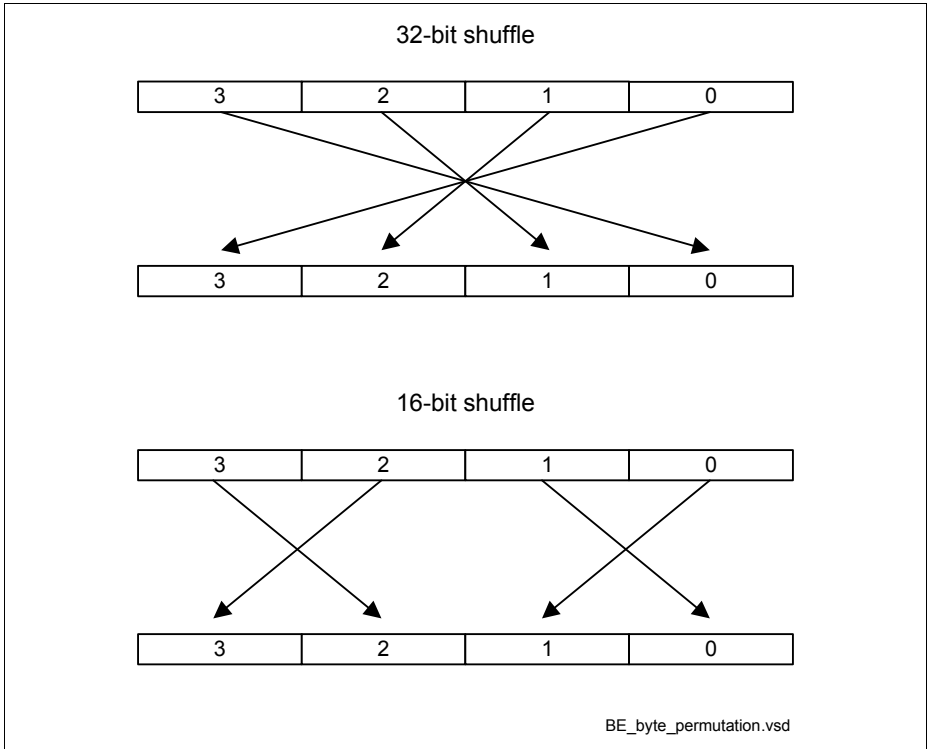


Figure 20-35 Endianness Byte Permutation

The endianness byte permutation should be used only for 16-bit and 32-bit data (and multiples of it), and not for other data lengths.

20.6.2 Loop-Back Mode

Loop Back mode is a master mode test feature used for establishing a module internal connection between the transmit and the receive signal, the same as if the transmit and receive pins would be short connected. It is activated by setting the **GLOBALCON.LB** bit.

Queued Synchronous Peripheral Interface (QSPI)

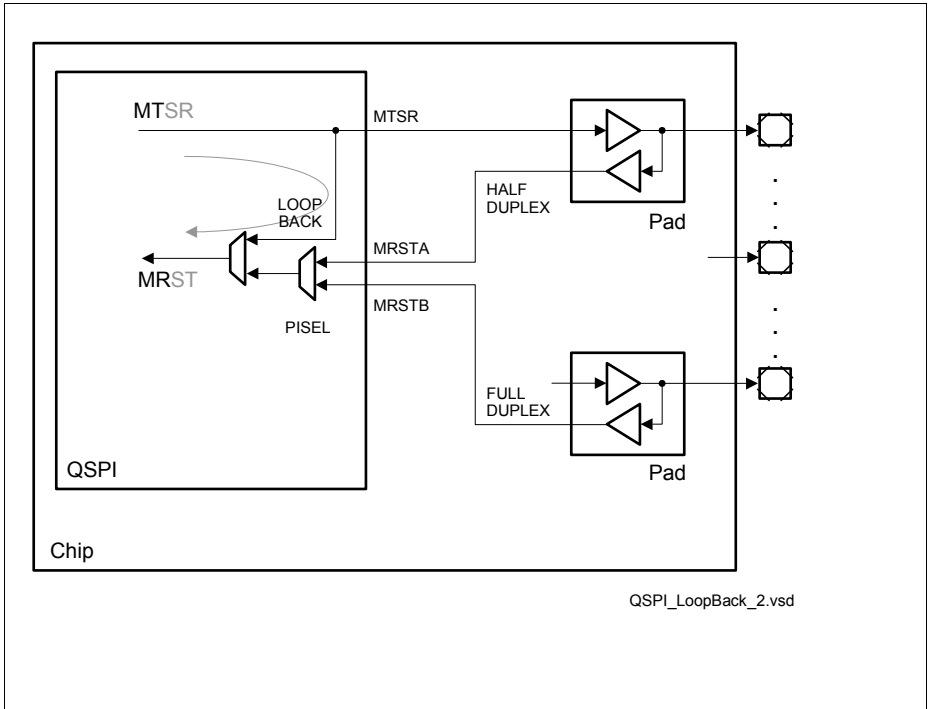


Figure 20-36 Loop-Back Mode

Queued Synchronous Peripheral Interface (QSPI)

20.7 Interrupts

There are several types of interrupts:

- Interrupts related to the user interface
 - FIFO related interrupts
- Interrupts related to the shift engine (state machine)
 - error interrupts
 - phase transition interrupts

According to this classification, here follows the complete list of all the interrupts:

- FIFO related interrupts
 - TX - Transmit FIFO Interrupt, requests feeding of the TXFIFO, but does not signal the error events of overflow and underflow (see [Figure 20-24](#))
 - RX - Receive FIFO Interrupt, requests emptying of the RxFIFO, but does not signal the error events of overflow and underflow (see [Figure 20-24](#))
- ERR Error Interrupt
 - signals every error condition
 - signals TX_FIFO_OVERFLOW, TX_FIFO_UNDERFLOW
 - signals RX_FIFO_OVERFLOW, RX_FIFO_UNDERFLOW
- PT - Phase transition interrupt, signalling two out of all phase transitions in the QSPI communication cycle (see [Figure 20-18](#) and [Figure 20-21](#))
 - PT1 event
 - PT2 event
- U - User Interrupt
 - triggered by the PT1 event during the time [BACON.UINT](#) bit keeps this event enabled.

Queued Synchronous Peripheral Interface (QSPI)

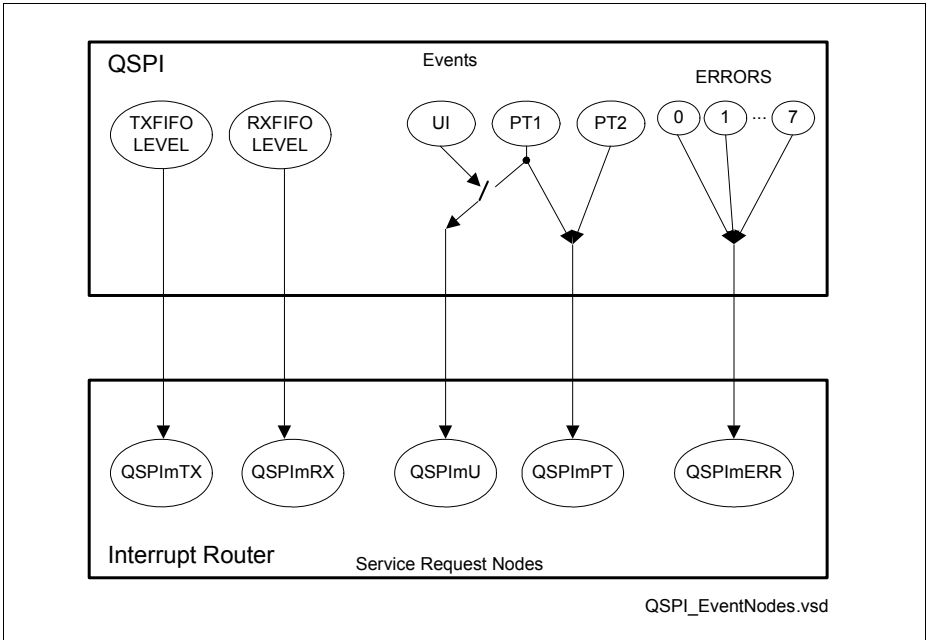


Figure 20-37 Events and Interrupts Overview

20.7.1 Slave Mode SLSI Interrupt

In slave mode, deactivating the SLSI signal (SLSI rising edge) triggers the PT interrupt, if PT2 enabled.

Queued Synchronous Peripheral Interface (QSPI)

20.7.2 Interrupt Flags Behavior

An state of an event flag (low or high) does not influence the flow of its event signal. If an event occurs recurrently, and the event flag remains set (if not cleared by the software interrupt handler), the interrupt flag (which is cleared by hardware, by an ICU, when the interrupt wins an arbitration round) would receive recurrent triggers, and the flag would be constantly set all the time.

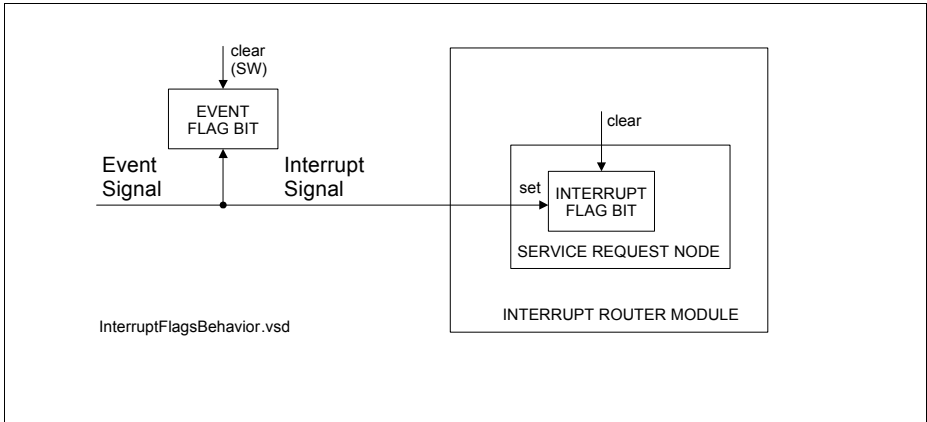


Figure 20-38 Events and Interrupts Overview

Queued Synchronous Peripheral Interface (QSPI)

20.7.3 TXFIFO Interrupt Generation

The TXFIFO provides two interrupt generation modes, selected by the bit-field **GLOBALCON1.TXFM**:

- Single Move Mode
- Batch Move Mode
- Combined Mode

The RXFIFO and the TXFIFO can use the interrupt generation modes independently of each other.

Single Move Mode

The purpose of the Single Move Mode is to keep the TXFIFO as full as possible, refilling the TXFIFO by writing to it as soon as there is a free element.

The single move mode supports primarily a DMA operation using single move per TXFIFO interrupt. In this mode the DMA keeps the TXFIFO full. A DMA request is triggered each time a write to the TXFIFO is performed, and there is at least one empty element available. If the write makes the TXFIFO full (no more empty elements), an interrupt is not generated in order to prevent overflow. It is generated later when the shift register (or BACON) reads one element from the TXFIFO, making it not full.

The **Figure 20-39**, shows the filling levels of the TXFIFO, and the events associated with each filling level triggering a TXFIFO refill interrupt.

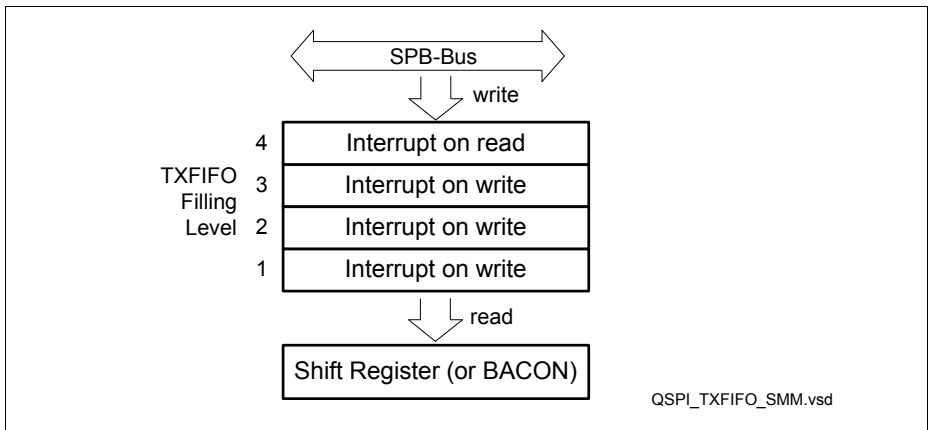


Figure 20-39 Interrupt Generation in the Single Move Mode

Queued Synchronous Peripheral Interface (QSPI)

Batch Move Mode

Batch Move Mode supports the following use case:

- CPU servicing the TXFIFO

The purpose of the Batch Move Mode is to reduce the number of interrupts by triggering an interrupt when more than one TXFIFO elements are empty. For example, a CPU can be interrupted less frequently and it can perform more than one moves per interrupt.

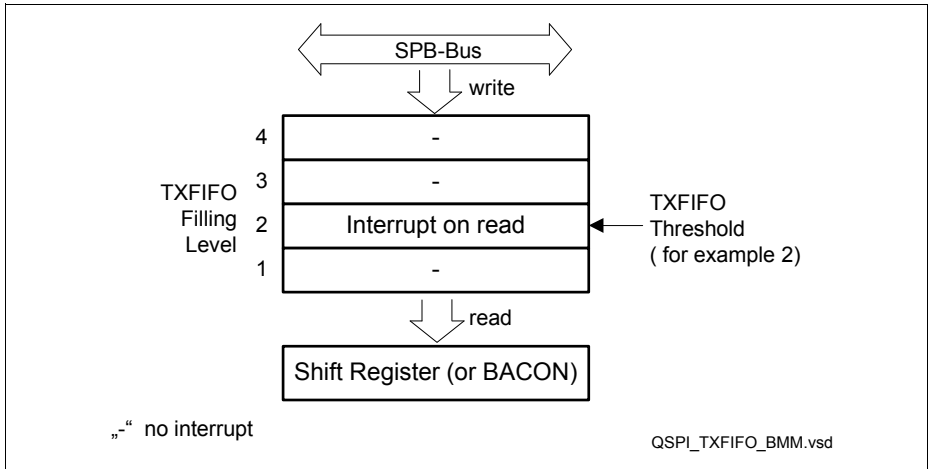


Figure 20-41 TXFIFO - Interrupt Generation in the Batch Move Mode

In Batch Move Mode, an interrupt is generated only at one point in the TXFIFO, when the filling level falls below the programmed threshold, implying that there are at least the predefined number of empty TXFIFO elements available.

Attention: *The TXFIFO must be refilled until the filling level has risen above the interrupt threshold (ensured by polling the filling level), in order to guarantee that the next interrupt will occur.*

At high baud rates and short frames, it could be possible that a couple of frames have been transmitted until the moment the TXFIFO is refilled. For example, if the threshold is set to three full elements, then the interrupt will be set when the filling level falls below three, making two elements free. If the refill moment of the TXFIFO is delayed so long that two frames have been transmitted (or frame and BACON), the TXFIFO will become empty, and refill sequence of two will not reach the level of three. So, the next interrupt would never come unless the filling level is being polled and the CPU keeps refilling in a loop until the threshold has been passed (ideally the TXFIFO would be filled completely with each batch of moves).

Queued Synchronous Peripheral Interface (QSPI)

Combined Mode

TXFIFO generates interrupt each time a data is fetched from it below the pre-programmed threshold, as defined in the **GLOBALCON1.TXFIFOINT**. Consequently, it generates two close interrupts when delivering BACON and the first following data. If the BACON TX interrupt is not serviced until the DATA1 TX interrupt comes, which will always be the case due to their time proximity, the BACON TX interrupt will be lost. Therefore, it is recommended that a DMA should always perform two moves per interrupt trigger (transfer size of two). Otherwise, the TXFIFO average filling level will go down with the time and at the end remain oscillating between zero and one (except in continuous mode). The DMA transaction loss event that may be raised by the DMA should be ignored.

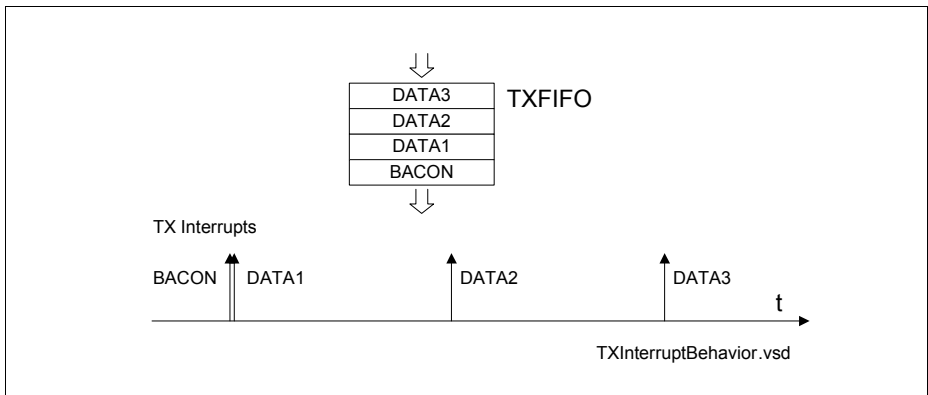


Figure 20-42 Events and Interrupts Overview

Queued Synchronous Peripheral Interface (QSPI)

20.7.4 RXFIFO Interrupt Generation

The RXFIFO provides two interrupt generation modes, selected by the bit field **GLOBALCON1.RXFM**:

- Single Move Mode
- Batch Move Mode
- Combined Mode

Single Move Mode

The purpose of the Single Move Mode is to keep the RXFIFO as empty as possible, by fetching the received elements one by one as soon as possible.

The single move mode supports primarily a DMA operation using single move per RXFIFO interrupt. In this mode the DMA keeps the RXFIFO as empty as possible. A DMA request is triggered each time a read from the RXFIFO is performed, and the RXFIFO is afterwards still not empty. If the read makes the RXFIFO empty, an interrupt is not generated in order to prevent underflow. It is generated later when the shift register (or STATUS) writes new element to the empty RXFIFO.

The **Figure 20-39**, shows the filling levels of the RXFIFO, and the events associated with each filling level triggering a RXFIFO refill interrupt.

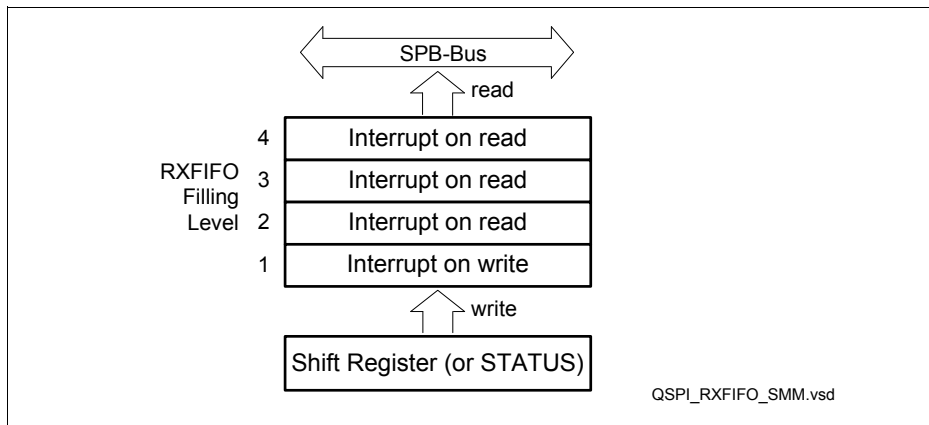


Figure 20-43 RXFIFO - Interrupt Triggering in the Single Move Mode

Queued Synchronous Peripheral Interface (QSPI)

The initial RXFIFO interrupt is triggered by the Shift Register, after it delivers the first received element. Afterwards, the DMA trigger-chain of refetch interrupts is self sustaining until the whole transaction is over. At the end of the DMA transaction, there is no service request remaining active in the service request node, due to the fact that the read of the last element in the RXFIFO does not trigger an interrupt.

Attention: *In Single Move Mode multiple software reads or block DMA moves lead to multiple interrupts and (false) transaction lost events. Therefore they should be avoided - only single moves should be used.*

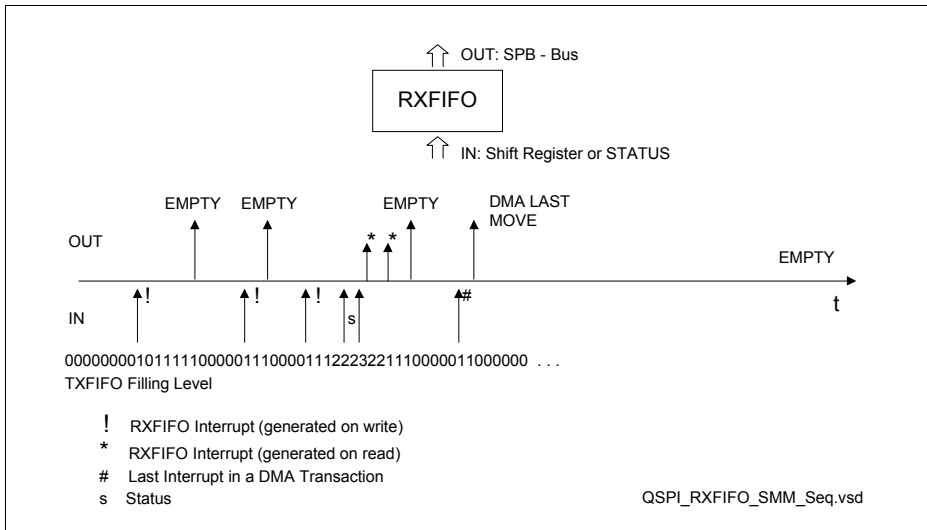


Figure 20-44 RXFIFO - Interrupt Operation in Single Move Mode

Queued Synchronous Peripheral Interface (QSPI)

Batch Move Mode

The purpose of the Batch Move Mode is to reduce the number of interrupts by triggering an interrupt when more than one RXFIFO elements are full. For example, a CPU can be interrupted less frequently and it can perform more than one move per interrupt.

Batch Move Mode supports the following use case:

- CPU servicing the RXFIFO

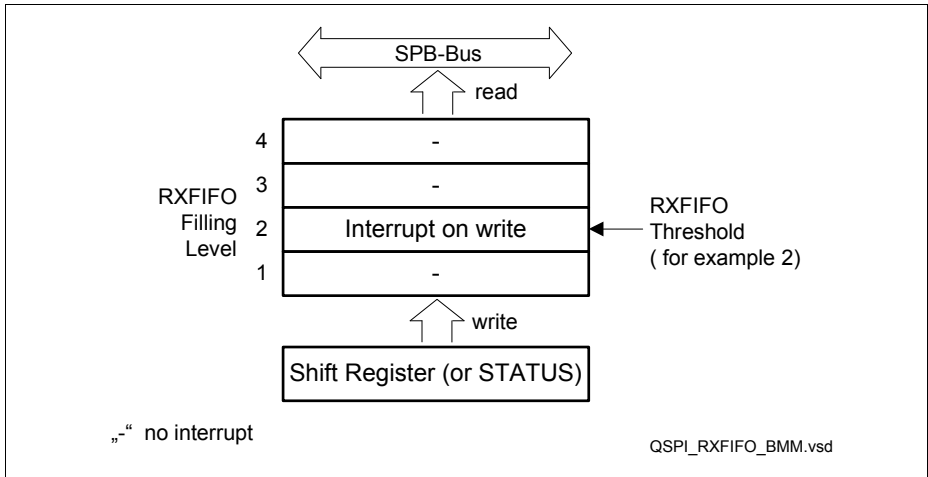


Figure 20-45 RXFIFO - Interrupt Triggering in the Batch Move Mode

In Batch Move Mode, an interrupt is generated only at one point in the RXFIFO, when the filling level rises above the programmed threshold, implying that there are at least the predefined number of full RXFIFO elements available.

Attention: It must be guaranteed that the CPU keeps emptying the RXFIFO and polling the RXFIFO filling level until the filling level has fallen below the interrupt threshold (or the RXFIFO is empty), in order to guarantee that next interrupt will occur.

At high baud rates and short frames, it could be possible that more than one frames have been received until the moment the RXFIFO is reempted. For example, if the threshold is set to two full elements, then the interrupt will be set when the filling level rises above two. If the emptying moment of the RXFIFO is delayed so long that two additional frames has been received (or a frame and STATUS), the RXFIFO will become full, and reempty sequence of two will not reach the level of two. So, the next interrupt would never come. This effect can not occur with threshold levels of 3 and 4. The threshold level of 1 is possible, but does not make much sense (use single move mode instead).

Queued Synchronous Peripheral Interface (QSPI)

Combined Mode

RXFIFO generates interrupt each time a data is written to it above the preprogrammed level, as defined in the [GLOBALCON1.RXFIFOINT](#). Consequently, it generates two close interrupts when filled with data and STATUS. If the data RX interrupt is not serviced until the STATUS RX interrupt comes, which will always be the case due to their time proximity, the STATUS RX interrupt will be lost. In such a case, it is recommended that the DMA performs two moves per interrupt trigger (transfer size of two). Otherwise, the RXFIFO average filling level will go up with the time and at the end overflow.

20.7.5 DMA Transfer Example

Transmitting one compact RAM queue using DMA could consist of the following steps:

- The user software configures one appropriate DMA channel (source address, destination address, one move per transfer, number of transfers per transaction, gating mode, single mode, interrupt at the end of the transaction etc...)
- A not full Tx FIFO from some QSPI module constantly generates pending DMA request
- A timer triggers a software routine that enables the DMA channel
- The DMA channel reads the queue elements from some on-chip general purpose RAM and writes them to the QSPI, one element at a time. One DMA transfer is one move long. One whole queue takes one transaction.
- The DMA channel generates an interrupt at the end of the transaction
- The DMA channel disables itself at the end of the transaction (single mode) and waits for further software actions.

The sequence is identical for the receive direction.

Queued Synchronous Peripheral Interface (QSPI)

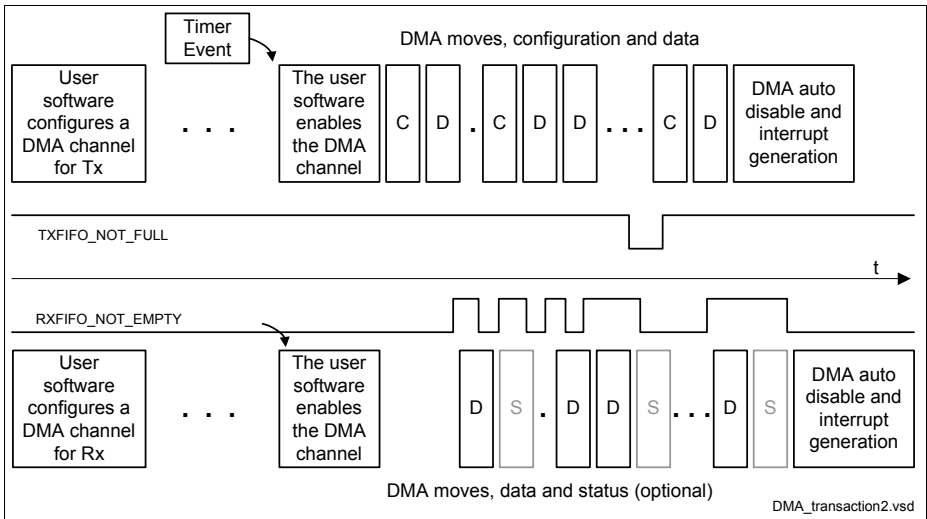


Figure 20-46 Example of transmitting a compact RAM Queue per DMA

20.8 Slave Mode

The slave mode is entered by setting the bit field GLOBALCON.MS to value 1x. Before entering the slave mode, the related parameters must be configured, including pin selection for SCLKI, SLSI and MTSR, pin input level, and pull-up/pull-down configuration. Subsequently state machine reset should be performed and afterwards the slave mode should be entered.

In slave mode data is being pushed into or out of the module through the serial pins by an external master. The RXFIFO must be regularly emptied by the system in order to avoid data loss, and the TXFIFO must be regularly fed by the system to avoid delivering all "1" data in case of empty TXFIFO.

If the slave mode relies on a slave select signal to mark the start and, in parallel to the bit counting, the end of a frame, then deactivating the slave signal by the master automatically resets the shift register state machine to wait state.

If the slave mode relies solely on bit counting for determining the frame end, then any SCLKI clock period longer than two bit times is treated as an end of a frame and simultaneously as a baud rate error.

In slave mode, the module simply immediately responds to external clock edges. Therefore, the settings for the leading, trailing, idle delays, and duty cycle and sampling point are irrelevant. From all timing settings, only the data length and the baud rate divider setting are relevant. The baud rate setting is important because of its role of watchdog timer of the incoming serial clock.

Queued Synchronous Peripheral Interface (QSPI)

Generally, the slave mode supports the same user interface as the master mode. This means the BACON-Data sequences remain the same. The SRF bit is ignored.

Limit the data length setting in the slave mode to the range of 2 to 32 bits. The byte setting **BACON.BYTE** is to be set to zero, that is, it always defines bits.

In case of receive only (simplex receive) the user software must reset the module in order to switch to transmit and receive mode (full duplex). Generally, mode reconfiguration (slave mode to master mode) requires module reset which resets the internal state machines and counters.

In case of slave transmit, the user software must write data to the TXFIFO before the first SCLKI edge of a frame arrives. If the TXFIFO is empty at this point of time, an underflow occurs, all "1" data is delivered, and underflow error interrupt is triggered if enabled. If the TXFIFO is written at the same time when the first SCLKI edge starts the shifting, the underflow occurs, interrupt will be triggered and data from the TXFIFO will be transmitted possibly corrupted - the first bit will be "1".

In case of TXFIFO / RXFIFO underflow the fifos deliver all "1" data .

In case of TXFIFO / RXFIFO overflow, the last data is lost. In such case data or optionally status in the RXFIFO can be lost.

The bit counting in the slave mode operates according to the following rules:

- The incoming bits (or shift clock shift edges) are counted and when the number defined in the **BACON.DL** is reached, the bits are transferred in the RXFIFO
- If the SLSI input is deactivated, the internal bit counter is reseted.

In slave mode, sleep requests may be enabled only when there is no pending transmit queue and no pending transmission and the TXFIFO is empty.

20.8.1 Shift Clock Phase and Polarity in Slave Mode

In slave mode the shift clock phase and polarity are fixed to CPH=1 and CPOL=0 (have to be programmed accordingly in the **ECON** register selected by **BACON.CS**) and slave select polarity is fixed to low active. The baud rate that the slave expects is defined in the **ECON** register pointed at by the **BACON.CS**.

Queued Synchronous Peripheral Interface (QSPI)

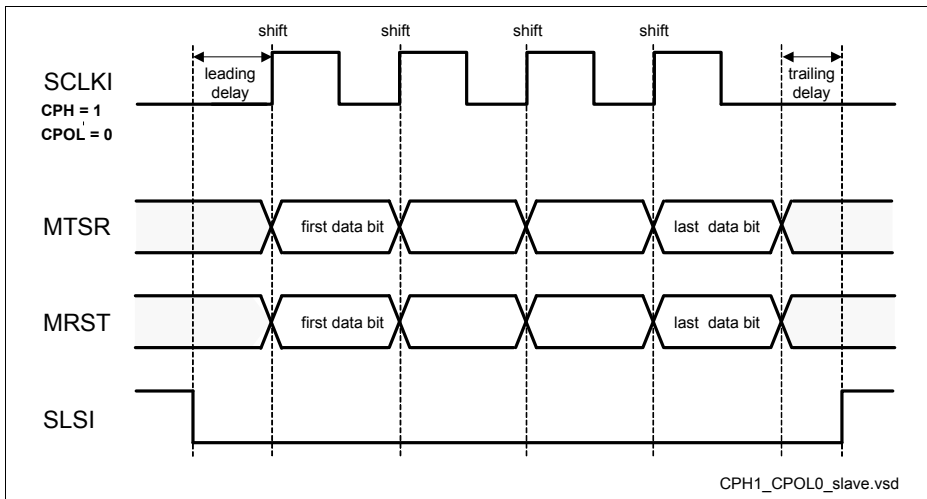


Figure 20-47 Slave transfer, CPH = 1, CPOL = 0

20.8.2 Shift Clock Monitoring

The QSPI module provides two mechanisms for monitoring the shift clock input signal in slave mode. These mechanisms monitor:

- if the receiving shift clock frequency (baud rate) lies within a certain range and
- if there are spikes on the shift clock input.

In case such disturbances are detected, the reaction is always the same:

- the corresponding error flags in **STATUS.ERRORFLAGS** are set, which can be cleared with **FLAGSCLEAR.ERRORCLEARS**
- the common error interrupt is raised if enabled in **GLOBALCON1.ERRORENS** and
- the bit **GLOBALCON.EN** is automatically cleared, if this feature is enabled by the bit **GLOBALCON.AREN**. The user software can activate a reset afterwards

In case of too high baud rate error, baud rate detection and spike detection both react, where spike detection is more effective. Spike detection reacts immediately, baud rate detection requires several bits with higher baud rate. Baud rate detection uses a window two nominal bit times wide and counts the real bit times: double baud rate error requires at least four real shift clock pulses to be detected, triple baud rate error requires six and so on. Baud rate detection does not react to all spikes smaller than one kernel clock period.

In case of too low baud rate error, baud rate detection reacts immediately, spike detection ignores this case.

Queued Synchronous Peripheral Interface (QSPI)
20.8.2.1 Baud Rate Error Detection

If the shift clock has less than half or more than double the expected baud rate, then:

- the corresponding flag is set in **STATUS.ERRORFLAGS**
- error interrupt is raised if enabled in **GLOBALCON1.ERRORENS**
- automatic clear of **GLOBALCON.EN** follows, if enabled by **GLOBALCON.AREN**.

20.8.2.2 Spike Detection

Short spikes on the clock line due to ground bouncing or clock ringing due to over/undershoots would cause one or more extra bits to be written to the front end receive fifo of the slave. This extra bit, if no measures are taken by the software, would cause all the subsequent received frames to be corrupted (shifted).

The spike detection mechanism operates up to 50MBaud and checks if a bouncing spike has occurred on the clock line near to the correct receiving clock edges.

If a spike is detected, the software should use **GLOBALCON.RESETS** bitfield, and should clear the corresponding status flags. Alternatively a complete module reset may be performed via registers **KRST0/KRST1**.

The spike detection is based on two mechanisms:

- detection of two close writes to the front end receive FIFO by detecting filling level of two
- using an inverted watchdog set to ca. 75% to 80% SCLKI clock period, which is the optimal range.

The inverted watchdog is a timer which starts to count triggered on the write event to the front-end receive fifo of the slave. During the time the watchdog counts, no edge is allowed to occur. An edge in the forbidden time raises a spike error event.

If a spike occurs in the last 25% of the SCLKI period, when the watchdog has stopped counting, the watchdog will be triggered to count by the spike. The next regular edge will violate the forbidden time and raise a spike event.

The watchdog counts **ECON.B + ECON.C** time quantas. The following constraint applies: $50\% * T_{SCLKI} < \mathbf{ECON.B} + \mathbf{ECON.C} < 100\% * T_{SCLKI}$

Queued Synchronous Peripheral Interface (QSPI)

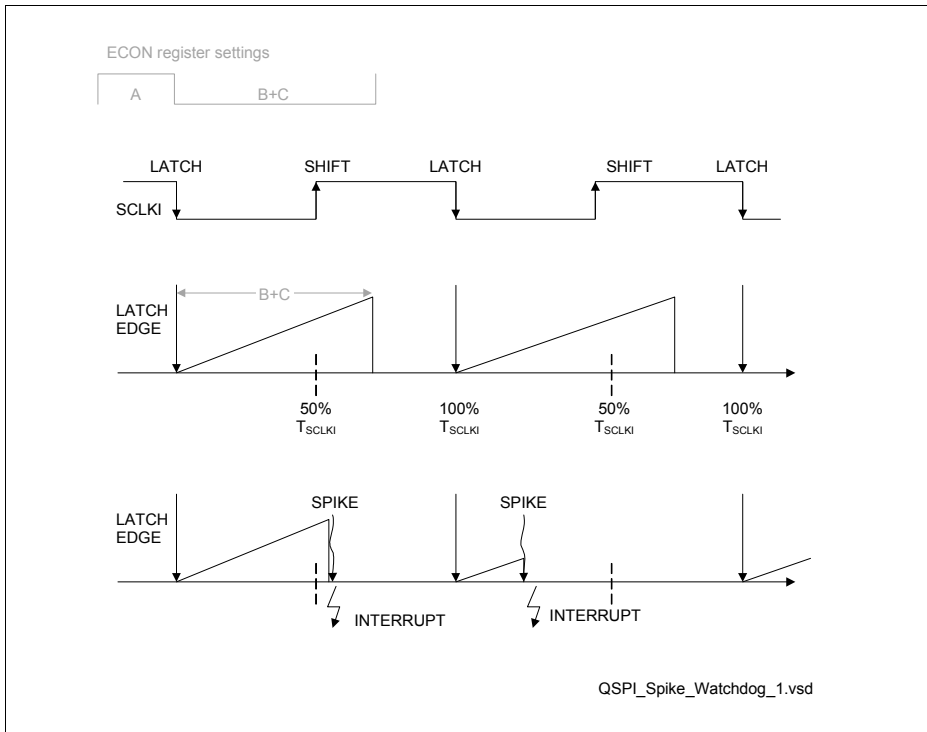


Figure 20-48 Spike Watchdog

20.8.2.3 Shift Clock Monitor Flags

The spike detection mechanism sets the flag **STATUS1.SPD** if it has detected a spike. The baud rate error detection mechanism sets the flag **STATUS1.BRD** if it has detected excessive baud rate deviation.

Writing 1 to SPD and BRD sets the bit and causes an interrupt if enabled. Writing 0 has no effect.

Both signals are ORed. This combined signal is shown in **STATUS.ERRORFLAGS** and raises an interrupt if enabled in **GLOBALCON1.ERRORENS**. Automatic clear of **GLOBALCON.EN** follows, if enabled by **GLOBALCON.AREN**.

If the **STATUS.ERRORFLAGS[2]** bit is cleared via **FLAGSCLEAR.ERRORCLEARS[2]**, the both flags SPD and BRD are automatically cleared.

Queued Synchronous Peripheral Interface (QSPI)

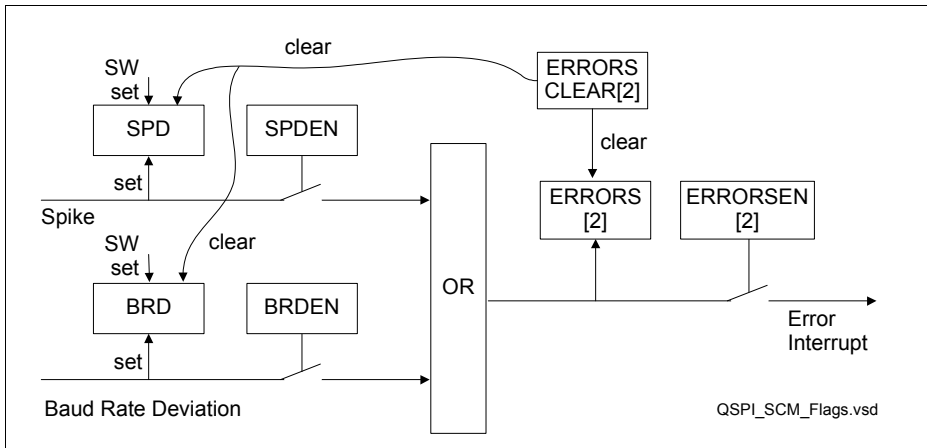


Figure 20-49 Shift Clock Monitor Flags

20.8.3 Parity

Parity settings for transmit and receive are defined by the corresponding bit fields in the **BACON** register.

The parity bit is concatenated to the data bits at transmission time. That means that a frame containing 16 bit data is 17 bits long (data + 1 parity). The transmitted parity bit can be read from the **STATUS.TPV** bit.

The parity bit is removed from the received bits at receive time automatically. The received parity bit can be read from the **STATUS.RPV** bit.

Attention: Parity is available only if the bit field **BACON.BYTE = 0**, that is, only for data of up to 32 bits length, that is, only in short mode.

Attention: The parity bit is concatenated to the data at the **LSB** location. If the bit field **BACON.MSB = 0** it is shifted out first, else it is shifted out last.

Queued Synchronous Peripheral Interface (QSPI)

20.9 Kernel Registers

This section describes the kernel registers of the QSPI module. All QSPI kernel register names described in this section will be referenced in other parts of the TC26x B-Step User’s Manual by the module name prefix “QSPI0_” for the QSPI0 interface, “QSPI1_” for the QSPI1 interface and so on.

All registers in the QSPI address spaces are reset with the application reset (definition see SCU section “Reset Operation”).

QSPI Kernel Register Overview

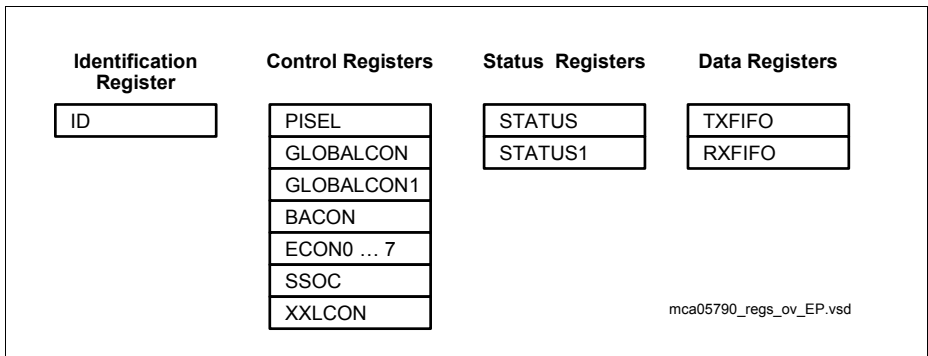


Figure 20-50 QSPI Kernel Registers

The following tables give the overview of the QSPI base addresses and registers.

Table 20-1 Registers Address Space

Module	Base Address	End Address	Note
QSPI0	F0001C00 _H	F0001CFF _H	–
QSPI1	F0001D00 _H	F0001DFF _H	–
QSPI2	F0001E00 _H	F0001EFF _H	–
QSPI3	F0001F00 _H	F0001FFF _H	–

Table 20-2 Registers Overview

Register Short Name	Register Long Name	Offset ¹⁾ Address	Write ²⁾ Access	Reset Value (hex)
PISEL	Port Input Select Register	04 _H	SV, P	0000 0000
ID	Module Identification Register	08 _H	BE	00C0 C0XX

Queued Synchronous Peripheral Interface (QSPI)
Table 20-2 Registers Overview (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾ Address	Write ²⁾ Access	Reset Value (hex)
reserved	reserved	0C _H	BE	
GLOBALCON	Global Configuration Register	10 _H	SV, P	000F 30FF
GLOBALCON1	Global Configuration Register 1	14 _H	SV, P	0005 0000
BACON	Basic Configuration Register	18 _H	BE	0F87 1C71
reserved	reserved	1C _H	BE	
ECONz (z = 0 - 7)	Configuration Extension z Reg.	20 _H ...3C _H	SV, P	0000 1450
STATUS	Status Register	40 _H	U, SV, P	0000 0000
STATUS1	Status Register 1	44 _H	U, SV, P	0000 0000
SSOC	Slave Select Output Control R.	48 _H	SV, P	0000 0000
reserved	reserved	4C _H	BE	
reserved	reserved	50 _H	BE	
FLAGSCLEAR	Flags Clear Register	54 _H	U, SV, P	0000 0000
XXLCON	Extra Large Data Config. Reg.	58 _H	U, SV, P	0000 0000
MIXENTRY	MIX_ENTRY Register	5C _H	U, SV, P	0000 0000
BACONENTRY	BACON_ENTRY Register	60 _H	U, SV, P	0000 0000
DATAENTRY_x	Data Entry Addresses, x=0 to 7	64 _H ...80 _H	U, SV, P	0000 0000
reserved	reserved	84 _H ...8C _H	BE	
RXEXIT	RX_EXIT Register	90 _H	BE	0000 0000
RXEXITD	RX_EXIT Debug Register	94 _H	BE	0000 0000
reserved	reserved	98 _H ...9C _H	BE	
CAPCON	Capture Control Register	A0 _H	U,SV,P	0000 0000
reserved	reserved	A4 _H ...E4 _H	BE	
BPI Registers				
CLC	Clock Control Register	00 _H	SV, E, P	0000 0003
OCS	OCDS Control and Status Reg.	E8 _H	SV, P	0000 0000
KRSTCLR	Reset Status Clear Register	EC _H	SV, E, P	0000 0000
KRST1	Reset Control Register 1	F0 _H	SV, E, P	0000 0000
KRST0	Reset Control Register 0	F4 _H	SV, E, P	0000 0000
ACCEN1	Access Enable Register 1	F8 _H	SV, SE	0000 0000
ACCEN0	Access Enable Register 0	FC _H	SV, SE	FFFF FFFF

Queued Synchronous Peripheral Interface (QSPI)

- 1) The absolute register address is Module Base Address ([Table 20-1](#)) + Offset Address (shown in this column)
- 2) All registers have read access mode U, SV. Read access to reserved addresses delivers bus error (BE).
All registers belong to application reset except OCS, which belongs to debug reset.

List of Access Protection Abbreviations

U	- User Mode
SV	- Supervisor Mode
BE	- Bus Error
nBE	- no Bus Error
P	- Access Protection, as defined by the ACCEN Register
E	- ENDINIT
SE	- SafetyENDINIT

Queued Synchronous Peripheral Interface (QSPI)

20.9.1 Kernel Registers

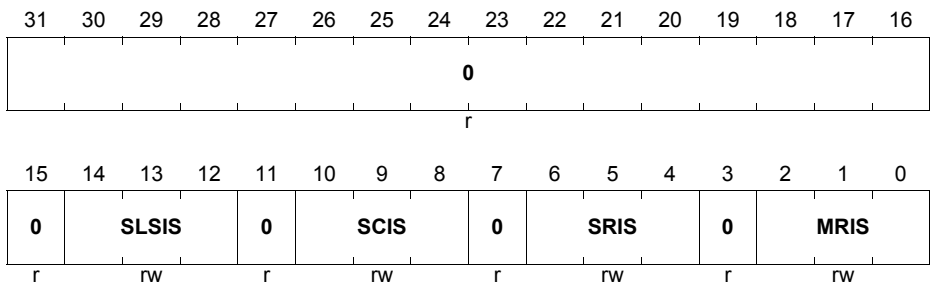
PISEL

The PISEL register controls the input signal selection of the SSC module.

(>> [Table 20-2](#) register overview)

PISEL

Port Input Select Register (04_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
MRIS	[2:0]	rw	Master Mode Receive Input Select MRIS selects one out of eight MRST receive input lines (0 to 7 correspondingly), used in Master Mode. The signal suffixes "A" to "H" correspond to the bit field values of 0 to 7: MRSTxA ->0, MRSTxB->1 ...
SRIS	[6:4]	rw	Slave Mode Receive Input Select SRIS selects one out of eight MTSR receive input lines (0 to 7 correspondingly), used in Slave Mode. The signal suffixes "A" to "H" correspond to the bit field values of 0 to 7: MTSRxA ->0, MTSRxB->1 ...
SCIS	[10:8]	rw	Slave Mode Clock Input Select SCIS selects one out of eight module kernel SCLK input lines (0 to 7 correspondingly) that is used as clock input line in slave mode. The signal suffixes "A" to "H" correspond to the bit field values of 0 to 7: SCLKIxA ->0, SCLKIxB->1 ...

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
SLSIS	[14:12]	rw	Slave Mode Slave Select Input Selection 000 _B Slave select input lines are deselected; QSPI is operating without slave select input functionality. 001 _B <u>SLSI</u> input line A is selected for operation 010 _B <u>SLSI</u> input line B is selected for operation 011 _B <u>SLSI</u> input line C is selected for operation 100 _B <u>SLSI</u> input line D is selected for operation 101 _B <u>SLSI</u> input line E is selected for operation 110 _B <u>SLSI</u> input line F is selected for operation 111 _B <u>SLSI</u> input line G is selected for operation The SLSIS must be programmed properly before the slave mode is set with GLOBALCON.MODE and the module is set to RUN mode.
0	[31:15], 11, 7, 3	r	Reserved Read as 0; should be written with 0.

Queued Synchronous Peripheral Interface (QSPI)

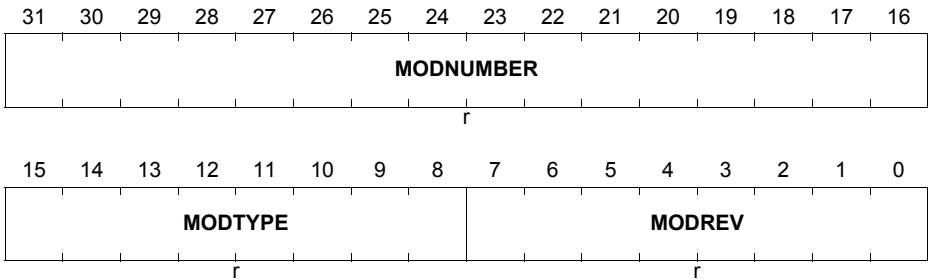
ID

The Module Identification Register ID contains read-only information about the module version.

(>> [Table 20-2](#) register overview)

ID

Module Identification Register (08_H) **Reset Value: 00C0 C0XX_H**



Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number MODREV defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MODTYPE	[15:8]	r	Module Type This bit field is C0 _H . It defines a 32-bit module.
MOD NUMBER	[31:16]	r	Module Number Value This bit field together with MODTYPE uniquely identifies a module.

Queued Synchronous Peripheral Interface (QSPI)

GLOBALCON

GLOBALCON contains configuration parameters which affect all channels.

(>> [Table 20-2](#) register overview)

GLOBALCON

Global Configuration Register

(10_H)

Reset Value: 000F 30FF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESETS				AR EN	MS	EN	0	STIP	SRF	STROBE					
w				rw	rw	rwh	r	rw	rw	rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEL 0	LB	EXPECT				SI	0	TQ							
rw	rw	rw				rw	r	rw							

Field	Bits	Type	Description
TQ	[7:0]	rw	Global Time Quantum Length Common n-divider scaling the baud rates of all channels in direction of higher or lower baud rates. Must not be changed during a running transaction. 0 _D division by 1 1 _D division by 2 ... _D ... 255 _D division by 256
SI	9	rw	Status Injection Selects if the status register content injection into the RxFIFO is enabled or disabled. The status injections, if enabled, is performed after each data block, depending on the BACON.DL and BYTE setting. 0 _B disabled 1 _B enabled
EXPECT	[13:10]	rw	Time-Out Value for the Expect Phase expressed in T_{QSPI} units 0 _D 64 (2 ⁶) 1 _D 128(2 ⁷) ... _D ... 15 _D 2097152 (2 ²¹ = 2 Mega)

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
LB	14	rw	Loop-Back Control Selects if the transmit output is internally connected to the receive input for test purposes. 0 _B Loop-Back inactive 1 _B Loop-Back active For detailed description, see the Loop-Back Mode section.
DELO	15	rw	Delayed Mode for SLSO0 Switches the delayed mode for external multiplexer enabling on and off 0 _B Delayed mode off 1 _B Delayed mode on
STROBE	[20:16]	rw	Strobe Delay for SLSO0 in Delayed Mode Defines the length of the SLSO0 delay in T_Q time units as defined for channel z (T_Q units) selected by the current BACON.CS , if GLOBALCON.DELO = 1. 00000 _B 1 00001 _B 2 ... _B ... 11111 _B 32
SRF	21	rw	Stop on RxFIFO Full If this bit is set, the data fetching out of the TxFIFO by the shift register stops when the RxFIFO is full, in order to prevent RxFIFO overflow. Master mode only. 0 _B Feature disabled 1 _B Feature enabled
STIP	22	rw	Slave Transmit Idle State Polarity This bit determines the logic level of the Slave Mode transmit signal MRST when the QSPI slave select input signals are inactive ($PISEL.SLSIS \neq 000_B$). 0 _B MRST = 0 when QSPI is deselected in Slave Mode. 1 _B MRST = 1 when QSPI is deselected in Slave Mode

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
EN	24	rwh	<p>Enable Bit</p> <p>Used to request transition between PAUSE and RUN mode per software. Cleared by hardware automatically at leaving the following states: disabled, suspend and sleep.</p> <p>In order to determine if the requested state has actually been reached, the STATUS.PHASE bit field should be polled. See also Operation Modes.</p> <p>0_B PAUSE requested 1_B RUN requested</p>
MS	[26:25]	rw	<p>Master Slave Mode</p> <p>Selects if the module operates in master or slave mode. This bit field must be configured before the first write to the TXFIFO.</p> <p>00_B Master Transmit and Receive 01_B Reserved 10_B Slave Transmit and Receive 11_B Slave Transmit and Receive</p>
AREN	27	rw	<p>Automatic Reset Enable</p> <p>Enables the reset of the GLOBALCON.EN on baud rate and spike error in slave mode.</p> <p>0_B disabled 1_B enabled</p>
RESETS	[31:28]	w	<p>Bits for resetting sub-modules per software</p> <p>Write to this bit field triggers a reset operation. The reset operation is not a single cycle operation, but takes several clock cycles, not more than the least common multiple of CCUCON0.SPBDIV and CCUCON0.BAUD2DIV (the configured division ratios, not the content of the bit fields).</p> <p>0000_B No reset triggered 0111_B State Machine, TXFIFO and RXFIFO reset, registers not reset 1111_B Module reset others, reserved</p> <p>For resetting the whole module, use KRST0 / KRST1 reset mechanism.</p>

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
0	8, 23	r	Reserved Read as 0; should be written with 0.

Note: If the EN bit is cleared first, and then the partial state machine reset activated, then the state machine goes into PAUSE state.

Queued Synchronous Peripheral Interface (QSPI)
GLOBALCON1

GLOBALCON1 contains bit fields for control of the extension of the slave select signals by using an external demultiplexer.

(>> [Table 20-2](#) register overview)

GLOBALCON1
Global Configuration Register 1
(14_H)
Reset Value: 0005 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0	0	RXFM	TXFM	PT2		PT1		RXFIFO INT	TXFIFO INT							
r	r	rw	rw	rw		rw		rw	rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
USR EN	0	0	PT2 EN	PT1 EN	RX EN	TX EN	ERRENS									
rw	r	r	rw	rw	rw	rw	rw									

Field	Bits	Type	Description
ERRENS	[8:0]	rw	Error Enable Bits Bits for enabling interrupt on all available error types 000000000 _B All errors disabled 000000001 _B Parity Error (PAREEN) 000000010 _B Unexpected Configuration Error 000000100 _B Baud Rate Error (slave mode) BEN 000001000 _B TXFIFO overflow (software error) 000010000 _B TXFIFO underflow (slave mode) TEN 000100000 _B RXFIFO overflow REN 001000000 _B RXFIFO underflow (software error) 010000000 _B EXPECT timeout 100000000 _B SLSI misplaced inactivation enable
TXEN	9	rw	Tx Interrupt Event Enable Enables the Tx interrupt. 0 _B Disabled 1 _B Enabled
RXEN	10	rw	Rx Interrupt Event Enable Enables the Rx interrupt. 0 _B Disabled 1 _B Enabled

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
PT1EN	11	rw	Interrupt on PT1 Event Enable Enables the PT interrupt on an PT1 event, as selected by the PT1 bit field. 0 _B Disabled 1 _B Enabled
PT2EN	12	rw	Interrupt on PT2 Event Enable Enables the PT interrupt on an PT2 event, as selected by the PT2 bit field. 0 _B Disabled 1 _B Enabled
USREN	15	rw	Interrupt on USR Event Enable Enables the USR interrupt on an USR event, as selected by the PT1 bit field. 0 _B Disabled 1 _B Enabled
TXFIFOINT	[17:16]	rw	Transmit FIFO Interrupt Threshold If the TXFIFO filling level is equal or less than this threshold, than each move of data or configuration from the TXFIFO triggers a transmit interrupt. Reset value of the level is 01 _B . 00 _B 1 01 _B 2 10 _B 3 11 _B 4
RXFIFOINT	[19:18]	rw	Receive FIFO Interrupt Threshold If the RXFIFO filling level is equal or greater than this threshold, than each move of data or status (if enabled) into the RXFIFO triggers a receive interrupt. Reset value of the level is 01 _B . 00 _B 0 01 _B 1 10 _B 2 11 _B 3

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
PT1	[22:20]	rw	<p>Phase Transition Event 1 Selects the first phase transition to trigger the PT interrupt.</p> <p>000_B BUSY (end of WAIT phase) 001_B SCLKPC (serial clock polarity change) 010_B SOF (Start of Frame) 011_B TBE (Transmit Buffer Emptied) 100_B RBF (Receive Buffer Filled) 101_B EOF (End of Frame) 110_B DNA (Data not Available = Start of Expect) 111_B CONT (End of EXPECT phase)</p>
PT2	[25:23]	rw	<p>Phase Transition Event 2 Selects the second phase transition to trigger the PT interrupt. In master mode, the following events are available:</p> <p>000_B BUSY (end of WAIT phase) 001_B SCLKPC (serial clock polarity change) 010_B SOF (Start of Frame) 011_B TBE (Transmit Buffer Emptied) 100_B RBF (Receive Buffer Filled) 101_B EOF (End of Frame) 110_B DNA (Data not Available = Start of Expect) 111_B CONT (End of EXPECT phase)</p> <p>In slave mode, the SLSI deactivated event is the only option available for PT2. For this purpose, always use the setting 101 (EOF).</p>
TXFM	[27:26]	rw	<p>TXFIFO Mode Selects between the Single Move Mode and the Batch Move Mode.</p> <p>00_B Combined Move Mode 01_B Single Move Mode 10_B Batch Move Mode 11_B reserved</p>
RXFM	[29:28]	rw	<p>RXFIFO Mode Selects between the Single Move Mode and the Batch Move Mode.</p> <p>00_B Combined Move Mode 01_B Single Move Mode 10_B Batch Move Mode 11_B reserved</p>

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
0	30, 31, 13,14	r	Reserved Read as 0; should be written with 0.

Queued Synchronous Peripheral Interface (QSPI)
BACON

Defines the basic configuration parameters for the current slave select. It can be read by software, or written through a write to the TXFIFO.

(>> [Table 20-2](#) register overview)

BACON
Basic Configuration Register
(18_H)
Reset Value: 0F87 1C71_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CS				DL				BY	MSB	UINT	PAR	TRAIL			
rh				rh				rh	rh	rh	rh	rh			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPRE			LEAD			LPRE			IDLE			IPRE		LAS	T
rh			rh			rh			rh			rh		rh	rh

Field	Bits	Type	Description
LAST	0	rh	Last Word in a Frame Defines if the following data word is last in the current frame or not 0 _B Not Last 1 _B Last
IPRE	[3:1]	rh	Prescaler for the Idle Delay Length in T_{BAUD2} units 000 _B 1 001 _B 4 010 _B 16 ... _B ... 111 _B 16384
IDLE	[6:4]	rh	Idle Delay Length Defines the length of both idle delays, IDLEA and IDLEB, in T_{BAUD2} units pre scaled with IPRE 0 _D 1 units 1 _D 2 unit ... _D ... 7 _D 8 units

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
LPRE	[9:7]	rh	Prescaler for the Leading Delay Length in T_{BAUD2} units 000 _B 1 001 _B 4 010 _B 16 ... _B ... 111 _B 16384
LEAD	[12:10]	rh	Leading Delay Length Defines the length of the leading delay, in T_{BAUD2} units pre scaled with LPRE 0 _D 1 units 1 _D 2 unit ... _D ... 7 _D 8units
TPRE	[15:13]	rh	Prescaler for the Trailing Delay Length in T_{BAUD2} units 000 _B 1 001 _B 4 010 _B 16 ... _B ... 111 _B 16384
TRAIL	[18:16]	rh	Trailing Delay Length Defines the length of the leading delay, in T_{BAUD2} units pre scaled with TPRE 0 _D 1 units 1 _D 2 unit ... _D ... 7 _D 8 units
PARTYP	19	rh	Parity Type Valid for both receive and transmit direction 0 _B Even parity 1 _B Odd parity

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
UINT	20	rh	<p>User Interrupt at the PT1 Event in the Subsequent Frames</p> <p>This bit is an enable signal for the PT1 event routed to the User Interrupt Service Request. The interrupt signals are generated until disabled with the next BACON.</p> <p>0_B Disabled 1_B Enabled</p>
MSB	21	rh	<p>Shift MSB or LSB First</p> <p>This bit sets the shift direction of the shift register. If the MSB option is set, and the data is a block longer than 32 bits, the block must be fed into the TXFIFO in reverse direction, from the end of the block until its beginning.</p> <p>0_B Shift LSB first 1_B Shift MSB first</p>
BYTE	22	rh	<p>Byte</p> <p>Defines if data length is expressed in bits or bytes</p> <p>0_B DL defines the data length in bits 1_B DL defines the data length in bytes</p>
DL	[27:23]	rh	<p>Data Length</p> <p>Defines the data length in bits or bytes of one data block, depending on the setting of the bit field BYTE</p> <p>0_D 2 bits if BYTE=0; XXL mode if BYTE=1 1_D 2 bits or bytes ..._D ... 31_D 32 bits or bytes</p> <p>For the maximum baud rate of 50 MBaud, the minimum data length possible is four.</p>
CS	[31:28]	rh	<p>Channel Select</p> <p>Selects the channel to which the subsequent data entry belongs (channel = the SLSO signal to be activated and the corresponding ECON configuration extension)</p> <p>This bit field selects one slave in a range of 0 to 15, by driving one SLSO signal out of 16 available.</p> <p>In case of an external demux mode, this bit field appears on the lines SLS01 to SLS04 as it is, additionally inverted or not, as defined in the SSOC register.</p>

Queued Synchronous Peripheral Interface (QSPI)

If an application does not use the SLSO signals of the QSPI module, but uses software channel selection, then the alternate function multiplexer in the ports must be configured so that the signal SLSO is not selected, but the corresponding bit Pn.OUT.

Alternatively, to emulate SLSO functionality, the software can disable (low) all the **SSOC**.OEN bits, and then toggle the **SSOC**.AOL bits.

Queued Synchronous Peripheral Interface (QSPI)

ECON

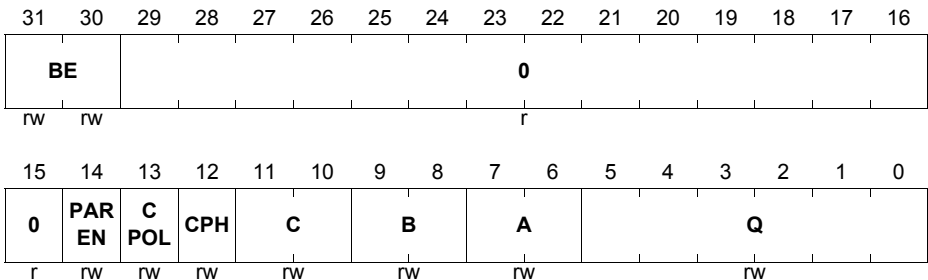
Configuration extensions for channels 0 to 15. Register x defines several timing characteristics for two channels: z and z+8.

(>> [Table 20-2](#) register overview)

(>> [Figure 20-12](#), [Figure 20-13](#) timing diagrams)

ECONz (z = 0 - 7)

Configuration Extension z (20_H + z*4_H) **Reset Value: 0000 1450_H**



Field	Bits	Type	Description
Q	[5:0]	rw	Time Quantum Defines the time quantum length used by A, B, and C to define the baud rate and duty cycle by. This prescaler cascades the prescaler GLOBALCON.TQ . 000000 _B 1 000001 _B 2 ... _B ... 111111 _B 64
A	[7:6]	rw	Bit Segment 1 Length expressed in time quantums of ECONz.Q . 00 _B 1 01 _B 2 10 _B 3 11 _B 4

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
B	[9:8]	rw	Bit Segment 2 Length expressed in time quanta of ECONz.Q. 00 _B 0 01 _B 1 10 _B 2 11 _B 3
C	[11:10]	rw	Bit Segment 3 Length expressed in time quanta of ECONz.Q. 00 _B 0 (if B=0, then C is minimum 1 per hardware) 01 _B 1 10 _B 2 11 _B 3
CPH	12	rw	Clock Phase Delay of one half SCLK clock cycle. 0 _B Disabled 1 _B Enabled
CPOL	13	rw	Clock Polarity Idle level of the shift clock signal at the SCLK pin 0 _B Idle level low 1 _B Idle level high
PAREN	14	rw	Enable Parity Check This bit field enables both the parity generation in transmit and parity check in receive direction. 0 _B Disabled 1 _B Enabled
BE	[31:30]	rw	Permutate bytes to / from Big Endian 00 _B Disabled 01 _B 16-bit big endian 10 _B 32-bit big endian 11 _B Disabled
0	[29:15]	r	reserved

Queued Synchronous Peripheral Interface (QSPI)

STATUS

Status register contains bits flagging the current status of the module and its sub-modules.

Note: It is not recommended to set the STATUS register flags per software for purposes other than testing. In such cases, use write instructions only, not read-modify-write instructions or bit instructions which compile to read-modify-write instructions.

Note: When using the RXFIFO status injection feature, only bits 22 to 31 reflect the status at the moment of injection, that is for the latest frame. Due to pipeline effects, the other bits contain not-latest information.

(>> [Table 20-2](#) register overview)

STATUS

Status Register (40_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PHASE			TPV	RPV	SLAVESEL			RXFIFO LEVEL			TXFIFO LEVEL				
rh			rh	rh	rh			rh			rh				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
USRF	0	0	PT2F	PT1F	RXF	TXF	ERRORFLAGS								
rwh	r	r	rwh	rwh	rwh	rwh	rwh								

Field	Bits	Type	Description
ERROR FLAGS	[8:0]	rwh	<p>Sticky Flags Signalling Errors</p> <p>Writing 1 sets the error Flag and triggers an error interrupt, if enabled. Writing 0 has no effect.</p> <p>00000000_B No Error 00000001_B Parity Error 00000010_B Unexpected Configuration Error 00000100_B Baud Rate Error (slave mode) 00001000_B TXFIFO overflow (software error) 00001000_B TXFIFO underflow (slave mode) 00010000_B RXFIFO overflow 00100000_B RXFIFO underflow (software error) 01000000_B EXPECT time out error 10000000_B SLSI misplaced inactivation (slave mode)</p>

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
TXF	9	rwh	Transmit Interrupt Request Flag Flags an occurrence of a request to feed the TXFIFO, which is generated when an element is fetched from the FIFO, and the FIFO filling level is equal or less than the set threshold level. Writing 1 sets the flag and triggers an interrupt if GLOBALCON1.TXEN = 1 . Writing 0 has no effect.
RXF	10	rwh	Receive Interrupt Request Flag Flags an occurrence of a request to empty the RXFIFO, which is generated when an element is written into the FIFO, and the FIFO filling level is equal or greater than the set threshold level. Writing 1 sets the flag and triggers an interrupt if GLOBALCON1.RXEN = 1 . Writing 0 has no effect.
PT1F	11	rwh	Phase Transition 1 Flag Flags an occurrence of a PT1 event, as selected with the GLOBALCON1.PT1 , and triggers an interrupt if GLOBALCON1.PT1EN = 1 . Writing 1 sets the flag and triggers an error interrupt. Writing 0 has no effect.
PT2F	12	rwh	Phase Transition 2 Flag In master mode, flags an occurrence of a PT2 event, as selected with the GLOBALCON1.PT2 , and triggers an interrupt if GLOBALCON1.PT2EN = 1 . In slave mode, set by the SLSI deactivated event. Writing 1 sets the flag and triggers an error interrupt. Writing 0 has no effect.
USRF	15	rwh	User Interrupt Request Flag Flags an occurrence of an USR event. Writing 1 sets the flag and triggers an interrupt if GLOBALCON1.USREN = 1 . Writing 0 has no effect.

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
TXFIFO LEVEL	[18:16]	rh	TXFIFO Filling Level Shows how many entries in the TXFIFO are waiting for transmission 000 _B 0 001 _B 1 010 _B 2 011 _B 3 100 _B 4 ... _B reserved
RXFIFO LEVEL	[21:19]	rh	RXFIFO Filling Level Shows how many entries in the RXFIFO are waiting for software to move them to RAM 000 _B 0 001 _B 1 010 _B 2 011 _B 3 100 _B 4 ... _B reserved
SLAVESEL	[25:22]	rh	Currently Active Slave Select Flag Displays the currently active slave select.
RPV	26	rh	Received Parity Value Displays the last received parity bit, if parity was enabled. Else if the parity is disabled, reads 0.
TPV	27	rh	Transmitted Parity Value Displays the last transmitted parity bit, if parity was enabled. Else 0.

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
PHASE	[31:28]	rh	Flags the ongoing phase Displays the current phase number. Relevant only in master mode. In slave mode this bit field indicates always 0. 0000 _B Wait 0001 _B Idle A 0010 _B Idle B 0011 _B Lead 0100 _B Data 0101 _B Trail 0110 _B Expect 0111 _B Lead Strobe 1000 _B Trail Strobe Not 0 means busy.
0	13, 14	r	Reserved Read as 0; should be written with 0.

Note: Slave TXFIFO underflow error is activated if the transmission has been started by the master at the time when the slave FIFO was empty, or at the same moment updated. In the second case inconsistent data will be transmitted.

Note: Reading the TXFIFO filling level bit field returns a value which can be used to calculate how many writes can be performed by a CPU without causing an overflow (in case no DMA is programmed to access the TXFIFO in parallel). For example, if the TXFIFO level was one, maximum three (TXFIFO depth of four minus one) write accesses are possible. Due to the volatility of the bit field, the filling level can go down in some nanoseconds after the read.

Note: Reading the RXFIFO filling level bit field returns a value which shows directly how many reads can be performed by a CPU without causing an underflow. Due to the volatility of the bit field, the filling level can go up in some nanoseconds after the read.

Note: Reading the PHASE bit field shows a previous phase of the frame, and a PT1F and PT2F flags indicate previous phase transitions. If the communication speed is not too high or the phases duration not very short compared to the software latency delays, which would normally be the case, these would be the latest completed phase and the latest phase transitions. Nevertheless, in case of high baud rates, the duration of the phases must be taken

Queued Synchronous Peripheral Interface (QSPI)

into consideration, but only if these bit fields are used in an application, and not only for debugging purposes.

Queued Synchronous Peripheral Interface (QSPI)

STATUS1

 (>> [Table 20-2](#) register overview)

STATUS1
Status Register 1

 (44_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SPD	SPD	BRD	BRD							0					
rw	rw	rw	rw							r					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0										
r		r		r		r								rh	

Field	Bits	Type	Description
BITCOUNT	[7:0]	r	Number of the bit shifted out Supports up to 256 bits. BITCOUNT = 0 indicates two states: no transmission and transmission of the first bit. The differentiation can be made by reading the phase information. BITCOUNT = 1 indicates transmission of the second bit and so on until 256. After transmission of the last bit the counter goes to zero.
BRDEN	28	rw	Baud Rate Deviation Enable Enables the signal path. 0 _B disabled 1 _B enabled
BRD	29	rw	Baud Rate Deviation Flag Shows if baud rate deviation has been detected. Write of 1 sets the bit and raises the event per software. Write of 0 has no effect. 0 _B no event 1 _B event detected
SPDEN	30	rw	Spike Detection Enable Enables the signal path. 0 _B disabled 1 _B enabled

 Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
SPD	31	rwh	Spike Detection Flag Shows if spike has been detected. Write of 1 sets the bit and raises the event per software. Write of 0 has no effect. 0 _B no event 1 _B event detected
0	[27:8]	r	Reserved Read as 0; should be written with 0.

If the **STATUS.ERRORFLAGS[2]** bit is cleared via **FLAGSCLEAR.ERRORCLEARS[2]**, the both flags SPD and BRD are automatically cleared.

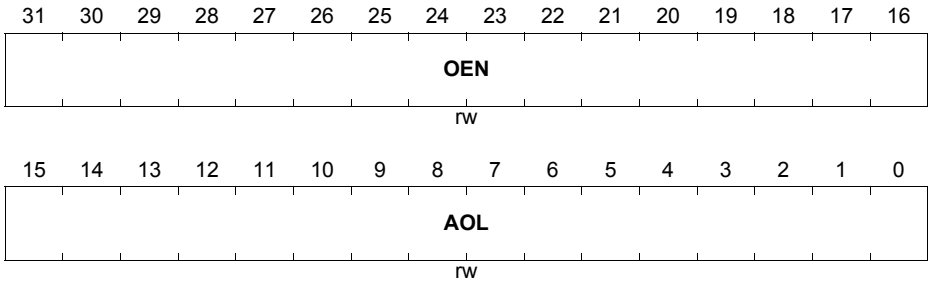
Queued Synchronous Peripheral Interface (QSPI)

SSOC

SSOC controls the level of each slave select and enables/disables each one individually.
 (>> [Table 20-2](#) register overview)

SSOC

Slave Select Output Control Register (48_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
OEN	[31:16]	rw	Enable Bits for the SLSO Outputs In disabled state the SLSO output drives the idle level as defined by the AOL bit field. "0" at certain position means that the corresponding SLSO is disabled. "1" means enabled.
AOL	[15:0]	rw	Active Output Level for the SLSO Outputs The idle level is the inverted one. "0" at certain position means active low level for the corresponding SLSO. "1" means active high.

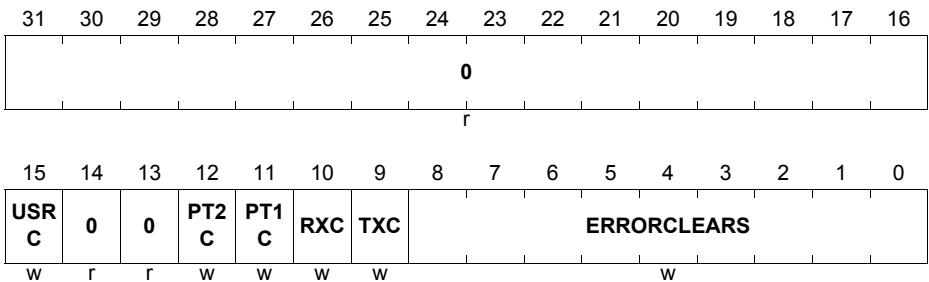
Queued Synchronous Peripheral Interface (QSPI)

FLAGSCLEAR

(>> [Table 20-2](#) register overview)

FLAGSCLEAR

Flags Clear Register (54_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
ERROR CLEARS	[8:0]	w	<p>Write Only Bits for Clearing the Error Flags Writing 1 clears the corresponding error flag in the ERORRFLAGS bit field. Reading returns 0.</p> <p>00000000_B No clear 00000001_B Parity Error clear 00000010_B Unexpected Configuration Error clear 00000100_B Baud Rate Error clear 00001000_B TXFIFO overflow clear 00010000_B TXFIFO underflow clear 00100000_B RXFIFO overflow clear 00100000_B RXFIFO underflow clear 01000000_B EXPECT time out clear 10000000_B SLSI misplaced inactivation clear</p>
TXC	9	w	<p>Transmit Event Flag Clear Write of 1 clears the STATUS.TXF bit. Write of 0 has no effect. Read delivers 0.</p> <p>0_B no action 1_B clear</p>

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
RXC	10	w	Receive Event Flag Clear Write of 1 clears the STATUS .RXF bit. Write of 0 has no effect. Read delivers 0. 0 _B no action 1 _B clear
PT1C	11	w	PT1 Event Flag Clear Write of 1 clears the STATUS .PT1F bit. Write of 0 has no effect. Read delivers 0. 0 _B no action 1 _B clear
PT2C	12	w	PT2 Event Flag Clear Write of 1 clears the STATUS .PT2F bit. Write of 0 has no effect. Read delivers 0. 0 _B no action 1 _B clear
USRC	15	w	User Event Flag Clear Write of 1 clears the STATUS .USRF bit. Write of 0 has no effect. Read delivers 0. 0 _B no action 1 _B clear
0	13, 14	rw	Reserved Read as 0; should be written with 0.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

Queued Synchronous Peripheral Interface (QSPI)

XXLCON

The XXLCON register provides counter for sending frames with very long blocks of data by extending the long data mode. It avoids a need for further BACON entries, like those needed in continuous mode.

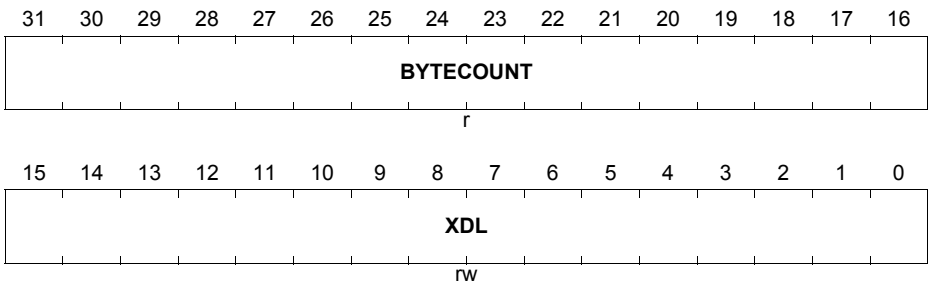
Data length in this register overrides the **BACON.DL** setting in XXL mode, when **BACON.BYTE=1** and **BACON.DL=0**. The data is sent to the slave as defined by **BACON.CS** bit field.

(>> **Table 20-2** register overview)

XXLCON

Extra Large Data Configuration Register(58_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
XDL	[15:0]	rw	<p>Extended Data Length Defines the length of the data block in bytes in range of 2 to 65536. Overrides BACON.DL when BACON.BYTE=1 and BACON.DL=0.</p> <p>0_D 2 bytes 1_D 2 bytes ..._D ... 65535_D 65536 bytes</p>

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
BYTECOUNT	[31:16]	r	<p>Extended Data Length</p> <p>In the XXL mode, shows the current state of the internal byte down counter (bytes remaining to be sent). In short and long modes, the value of this bit field is don't care.</p> <p>0_D zero bytes</p> <p>1_D two bytes</p> <p>..._D ...</p> <p>65535_D 65536 bytes</p>

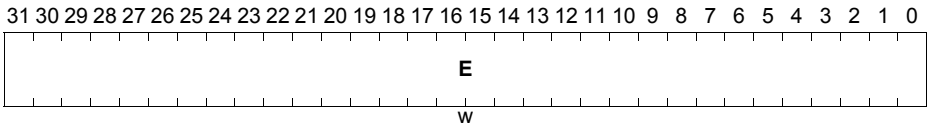
Queued Synchronous Peripheral Interface (QSPI)

MIXENTRY

(>> [Table 20-2](#) register overview)

MIXENTRY

MIX_ENTRY Register (5C_H) Reset Value: 0000 0000_H



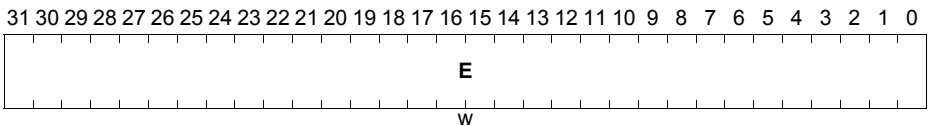
Field	Bits	Type	Description
E	[31:0]	w	Entry Point to the TxFIFO

BACONENTRY

(>> [Table 20-2](#) register overview)

BACONENTRY

BACON_ENTRY Register (60_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
E	[31:0]	w	Entry Point to the TxFIFO

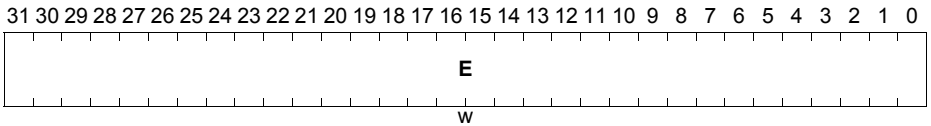
Queued Synchronous Peripheral Interface (QSPI)

DATAENTRY

(>> [Table 20-2](#) register overview)

DATAENTRY_x (x=0-7)

DATA_ENTRY Register x (64_H + x*4_H) Reset Value: 0000 0000_H



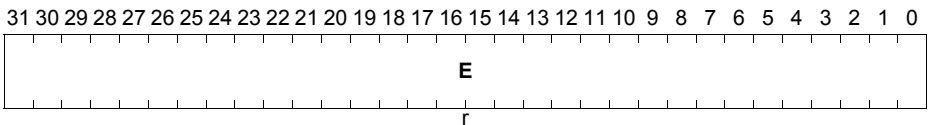
Field	Bits	Type	Description
E	[31:0]	w	Entry Point to the TxFIFO

RXEXIT

(>> [Table 20-2](#) register overview)

RXEXIT

RX_EXIT Register (90_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
E	[31:0]	r	Read Point from the RxFIFO

Note: The RXFIFO has a property that a read access from an empty RXFIFO generates an underflow interrupt, and delivers only "1" bits, which overrules the reset value. Therefore reading from a non initialized RXFIFO delivers all "1" and not all "0".

Queued Synchronous Peripheral Interface (QSPI)

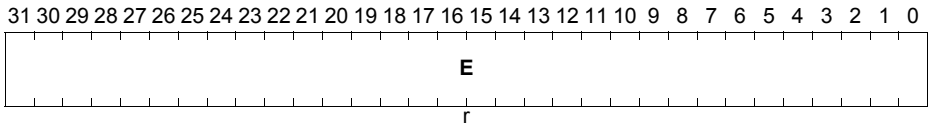
RXEXITD

The register **RXEXITD** provides a non-destructive address, showing the next available value in the RXFIFO.

(>> **Table 20-2** register overview)

RXEXITD

RX_EXIT Debug Register (94_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
E	[31:0]	r	Read Point from the RxFIFO

*Note: This register provides a non-volatile access to the RXFIFO. It delivers the same value as **RXEXIT**, but without affecting read pointer and the filling level of the RXFIFO.*

Queued Synchronous Peripheral Interface (QSPI)

20.10 Operation Modes

Generally, the QSPI module transmits a RAM queue of an arbitrary length, fed by a DMA module, and makes pauses between the queues. The QSPI module does not know anything about the queue length or the current position within the queue or the future schedule of the messages that may come.

If the QSPI module gets a request for switching off the clock or pausing the module, it disables all service request lines.

The QSPI module provides two options for switching off the clocks or pausing the module: hard suspend (as soon as possible regarding for example the length of interrupt request pulses), and soft suspend (transition to the PAUSE state with finishing the current frame). The requests can be triggered by hardware signals (disable, suspend, sleep) or by software write to **GLOBALCON.EN** bit.

Disable, Pause and Run

After being enabled by clearing the **CLC.DISR** bit, the QSPI enters the PAUSE state. In this state, the QSPI module can be initialized, the TXFIFO pre-filled, and then, by setting the **GLOBALCON.EN** bit, put into a RUN state.

The software requests PAUSE state of the running module by clearing the **GLOBALCON.EN** bit. The software can detect that the QSPI module has reached the PAUSE state by polling for the **GLOBALCON.EN=0** and **STATUS.PHASE=0** (indicating WAIT state, master mode only).

In PAUSE state, the user interface of the module is active. The registers and the FIFOs can be read and written. However, both the receive and transmit state machines are inactive, in master as well as in the slave mode. This allows for reconfiguration of the module without affecting the serial bus.

Queued Synchronous Peripheral Interface (QSPI)

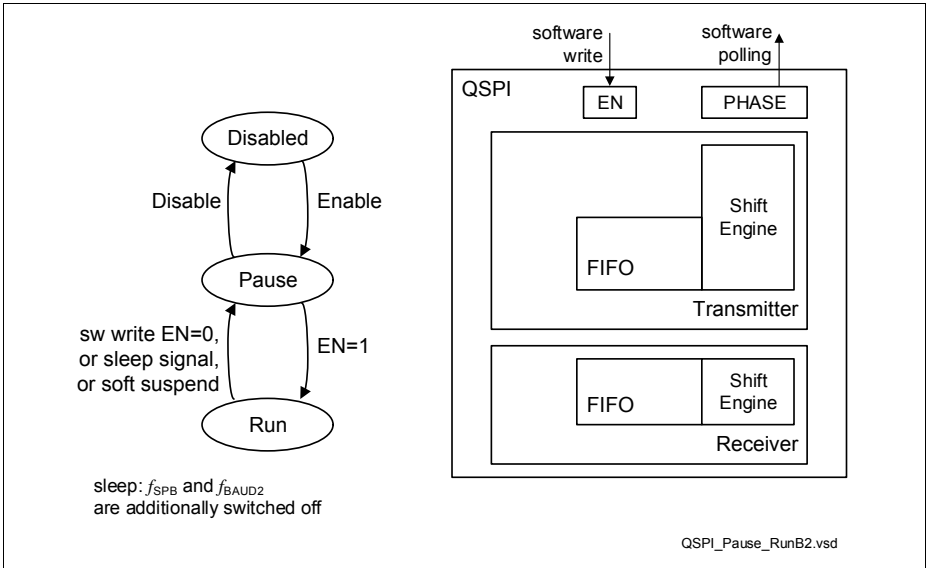


Figure 20-51 Pause and Run Overview

The concept of the PAUSE state is reused for handling the soft OCDS suspend and Sleep requests. In contrast to active Pause state and OCDS suspend, the sleep state has the special property that (due to the low power requirement) both FPI and QSPI clocks are switched off.

Queued Synchronous Peripheral Interface (QSPI)

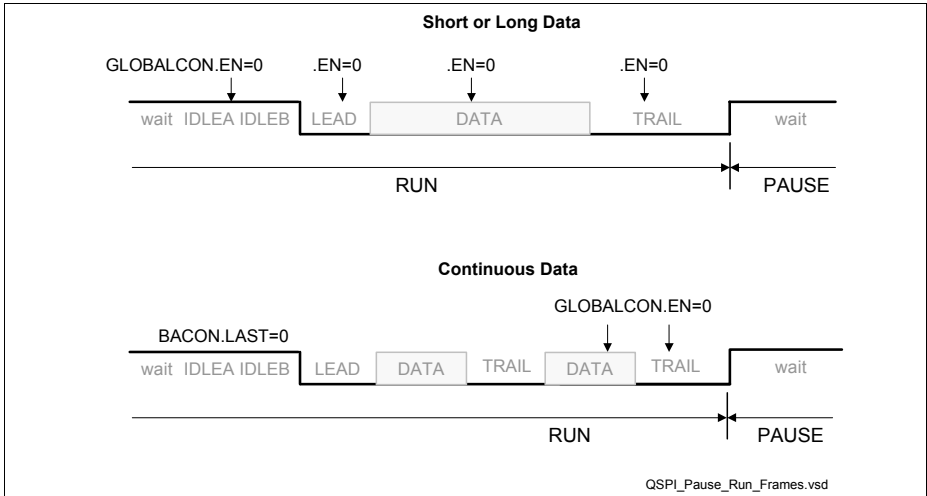


Figure 20-52 Entering the Pause State

If a running module receives a request to PAUSE, it waits for the end of the TRAIL phase. For all modes (Short, Long and Continuous), the PAUSE state is indicated by **GLOBALCON.EN=0** and **STATUS.PHASE=0**, master mode only.

Queued Synchronous Peripheral Interface (QSPI)

20.10.1 OCDS Suspend

The OCDS hard suspend request, for debugging purposes, simply freezes the QSPI module in the current state. No signal is changed including the service request lines, which remain frozen in the current state.

The QSPI responds to an OCDS soft suspend request by entering the PAUSE state immediately after the subsequent end of a trail phase, and then sending acknowledge (see [Figure 20-53](#)). The behavior is the same as for entering the **Sleep Mode**.

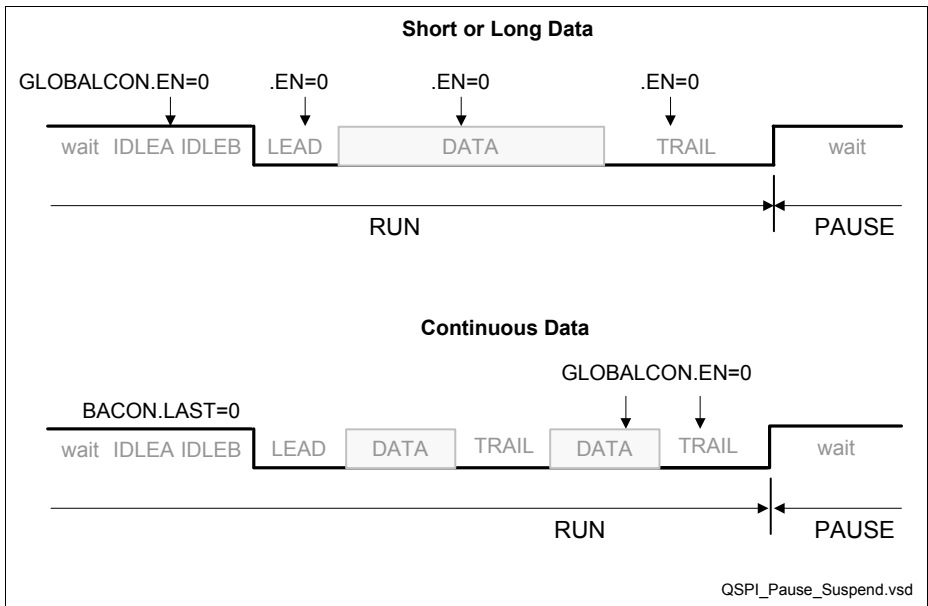


Figure 20-53 Entering the Sleep, Soft Suspend, and Disable State

Reading the FIFO entries by the Cerberus (detected by the FPI master tag) does not modify the FIFO content nor the read and write pointers.

Queued Synchronous Peripheral Interface (QSPI)

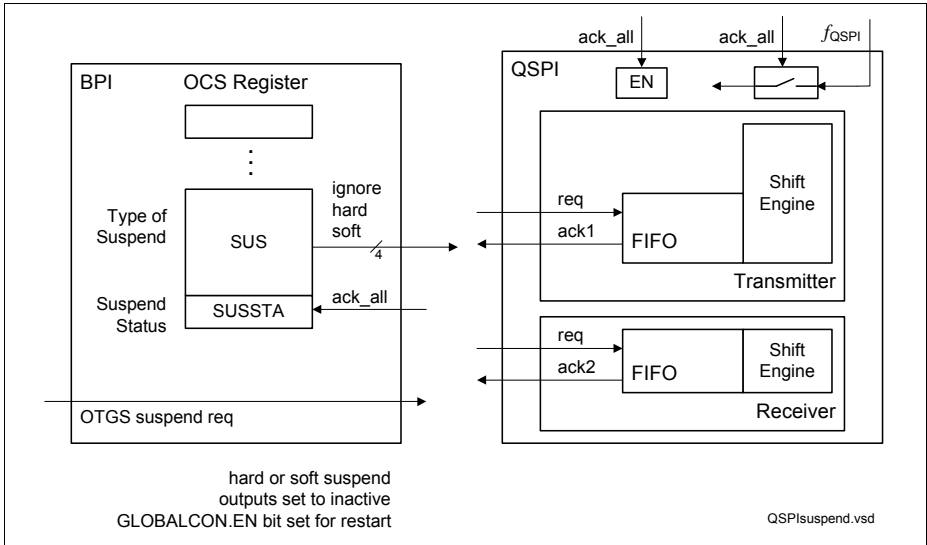


Figure 20-54 Suspend Overview

Note: In Hard Suspend Mode the QSPI kernel clock is switched off immediately. Reading and writing of registers is possible but will enable the kernel clock for a few cycles. Attention: register accesses with clocking in Hard Suspend Mode can have unintended side effects like signals becoming and staying active. This can affect also other modules, so a QSPI kernel reset might not be sufficient to bring the system into a defined state.

Queued Synchronous Peripheral Interface (QSPI)

20.10.2 Sleep Mode

Sleep signal is a system wide hardware signal requesting from all modules to go to low power saving state as soon as possible. For the QSPI module, going to sleep mode can be disabled or enabled by setting or clearing the CLC.EDIS bit. The module enters the sleep as soon as possible and remains in this state as long as the hardware sleep signal is active. Upon deactivation of the sleep signal, the QSPI wakes in PAUSE state (**GLOBALCON.EN** bit is cleared by the QSPI module) and waits for the software to enable the **GLOBALCON.EN** bit.

If CLC.EDIS bit is cleared, the sleep mode is enabled and the module enters the sleep state:

in master mode

- after ending the TRAIL delay if a transmission is in progress
- or immediately if the module is in WAIT state

in slave mode

- after the bit counter has counted the full predefined number of bits as defined in GLOBALCON.DL and BYTE bit fields.
- immediately if the module is in wait state, no transmission ongoing and shift register empty and TXFIFO empty
- or immediately if the SLSI is inactive (the one selected by PISEL)

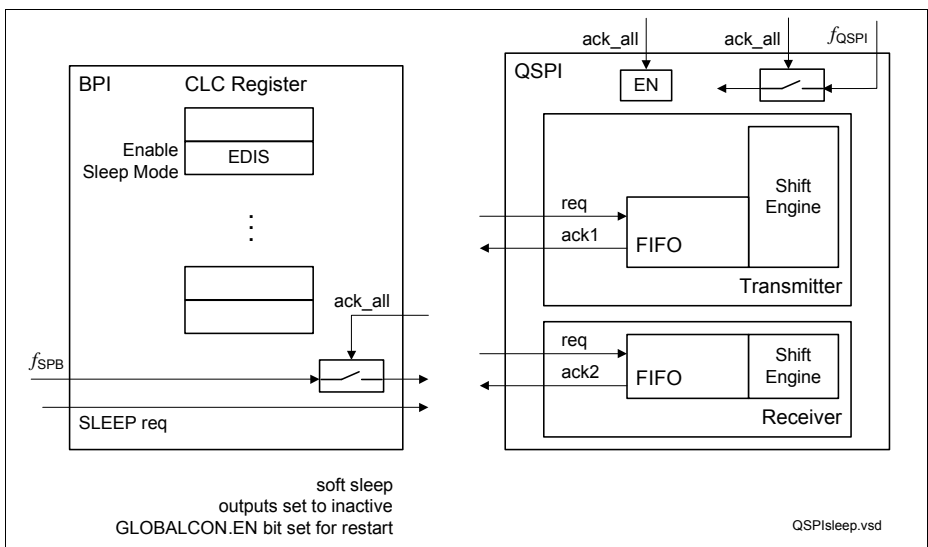


Figure 20-55 Sleep Overview

Queued Synchronous Peripheral Interface (QSPI)

It is responsibility of the user software to have the sleep enabled in a safe time intervals, where no race conditions or pipeline effects between the QSPI module and the DMA can occur in case of sleep request.

Note: CLC disable request and sleep request block immediately kernel register write accesses. This includes also the TXFIFO access. Therefore if the module expects more data (like in long or continuous mode), it will enter the expect phase and remain there and never do an acknowledge and enter the required state. Periodic timeout interrupts will be generated. Therefore it is not recommended to request a sleep or disable during an ongoing long or continuous transfer.

20.10.3 Disabling the QSPI

The software can switch off the QSPI module by setting the CLC.DISR bit. It is responsibility of the user software to issue a disable request in a safe time interval, where no race conditions or pipeline effects between the QSPI module and the DMA can occur. This is no issue if the switch-off procedure is one-way, that is no continuation of operation without reinitialization/reset of the module is expected. **GLOBALCON.EN** bit is cleared.

The QSPI module itself behaves in the following way.

in master mode like entering the sleep mode:

- after ending the TRAIL delay if a transmission is in progress
- or immediately if the module is in WAIT state

in slave mode

- after the bit counter has counted the full predefined number of bits as defined in GLOBALCON.DL and BYTE bit fields, if a transmission is ongoing or
- immediately if the module is in wait state, no transmission ongoing and shift register empty and TXFIFO empty

Queued Synchronous Peripheral Interface (QSPI)

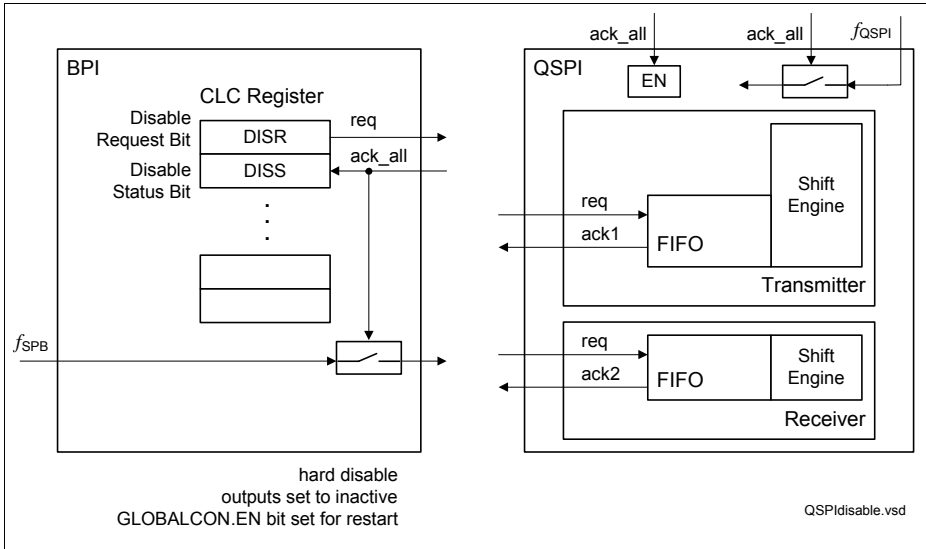


Figure 20-56 Disable Overview

20.11 Reset Behavior

There are two sources of reset:

- BPI (Bus Peripheral Interface)
- Reset bits in the kernel register - **GLOBALCON.RESETS**

BPI reset puts the whole QSPI module in reset state. All outputs get the reset value.

Software writing the **GLOBALCON.RESETS** bit field contains bits for resetting: TXFIFO, RXFIFO, shift state machine, register file. Setting all the bits in a single write resets the whole module.

Queued Synchronous Peripheral Interface (QSPI)

20.12 QSPI Module Implementation

This section describes the interfaces of the QSPI module instances with the clock control, port connections, interrupt control, and address decoding. There are 4 module instances, QSPI0 to QSPI3.

20.12.1 Module Identification Registers

The reset values of the QSPIx_ID module identification registers are 00C0 C0XX_H.

20.12.2 Interfaces of the QSPI Modules

Figure 20-57 shows the implementation details and interconnections of the QSPI module.

Each of the QSPI modules is supplied with a separate clock control, interrupt control, and address decoding logic. Two interrupt outputs can be used to generate DMA requests. The QSPIx I/O lines are connected to the ports as described in the pinning and ports chapters.

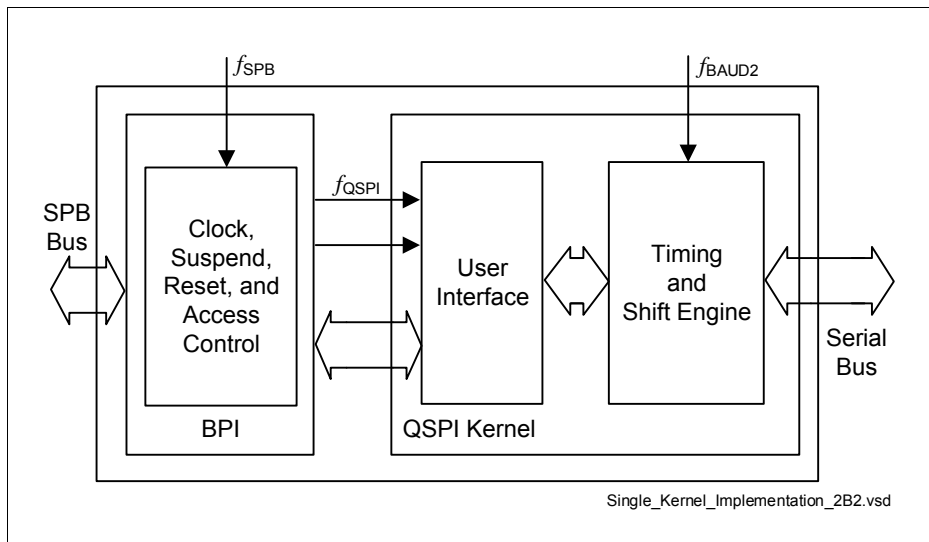


Figure 20-57 Module Implementation Overview

Queued Synchronous Peripheral Interface (QSPI)

20.12.3 On-Chip Connections

This section describes the on-chip connections of the QSPIx module instances.

SCU Connections

Wake-up from sleep is done by the SCU.ERU triggering a software interrupt on a programmable edge on SLSI pin.

DMA Requests

There are no dedicated DMA request lines. The interrupt or service request signals are general purpose request signals that can be routed to each service provider.

Port/Pin Connections

The connections of the QSPI modules to the pins / ports is described in the chapters regarding the pinning and the ports.

The MTSR and MRST signals are also mapped to LVDS output and input pads. For details see the Ports chapter and the Pinning chapter.

Note: If LVDSH receiver pad is used, the software must take into account the corresponding control and configuration bits, located in the HSCT module and in the corresponding port registers.

Input signals not connected to ports/pins are connected to the following voltage levels:

- The unconnected SCLKI signals are connected to low level "0"
- The unconnected SLSI signals are connected to (inactive) high level "1"
- The unconnected MRST input signals are connected to low level "0".

Queued Synchronous Peripheral Interface (QSPI)

20.12.4 QSPI Related External Registers

Figure 20-58 summarizes the module-related external registers which are required for QSPI programming. These are:

- port registers and
 - service request node registers,
- described in the corresponding chapters.

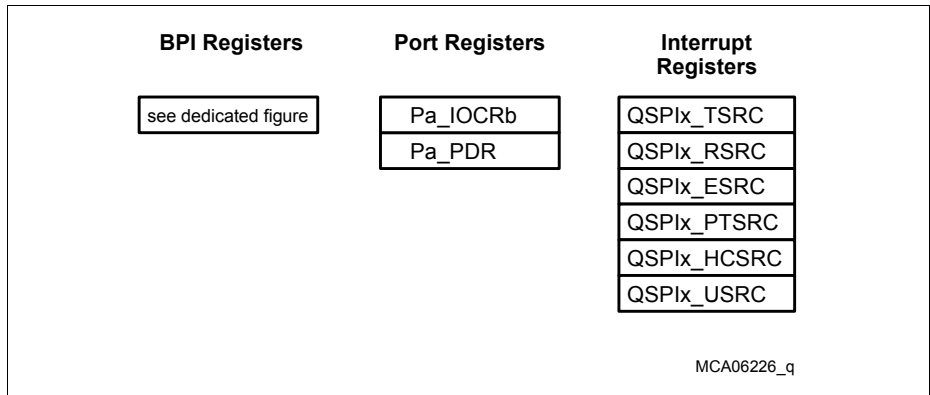


Figure 20-58 Implementation-specific Special Function Registers

Queued Synchronous Peripheral Interface (QSPI)

20.12.4.1 BPI_FPI Module Registers

System Registers

Figure 20-59 shows all registers associated with the BPI_FPI module, configured for one kernel. The Offset Address in the following BPI_FPI register table and in the BPI_FPI register descriptions are proposals. In a standard 32 bit peripheral the CLC should be mapped to offset address 00h, module ID to 08h. The proposal for the new BPI_FPI register is to map them to the end of the peripheral address range.

BPI_FPI Registers Overview

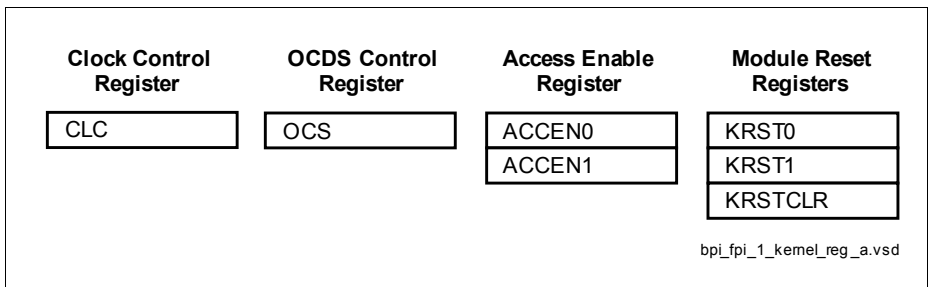


Figure 20-59 BPI_FPI Registers

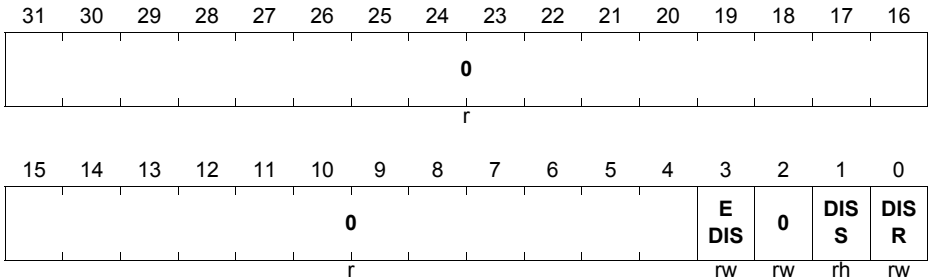
The writes of the bus masters to the QSPI module is controlled by Access Protection registers ACCENx.

The QSPI implements two ACCENx registers, ACCEN0 and ACCEN1.

The ACCENx registers are protected by Safety Endinit mechanism.

Clock Control Register (CLC)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI_FPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the f_{clk} module clock signal, sleep mode and disable mode for the module.

Queued Synchronous Peripheral Interface (QSPI)
CLC
Clock Control Register
(00_H)
Reset Value: 0000 0003_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
EDIS	3	rw	Sleep Mode Enable Control Used to enable the module's sleep mode. 0 _B Enabled 1 _B Disabled
0	[31:4]	r	Reserved Read as 0; should be written with 0.
0	2	rw	Reserved Should be written with 0.

OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32 bit wide only and requires Supervisor Mode

Queued Synchronous Peripheral Interface (QSPI)
OCS
OCDS Control and Status
(E8_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUS STA	SUS _P	SUS				0							
r		rh	w	rw				r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												0			
r												rw			

Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately. 2 _H Soft suspend others , reserved
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:0], [31:30]	r	Reserved Read as 0; must be written with 0. The bits [4:0] are rw (readable and writable).

Access Enable Register (ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

Queued Synchronous Peripheral Interface (QSPI)

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ... , EN31 -> TAG ID 011111_B.

ACCEN0

Access Enable Register 0

(FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
ENn (n = 0-31)	n	rW	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register (ACCEN1)

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000_B, EN1 -> TAG ID 100001_B, ... , EN31 -> TAG ID 111111_B.

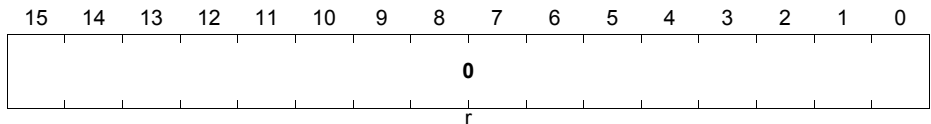
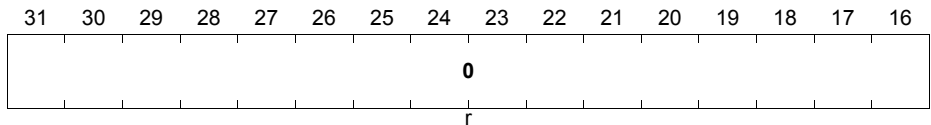
Queued Synchronous Peripheral Interface (QSPI)

ACCEN1

Access Enable Register 1

(F8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

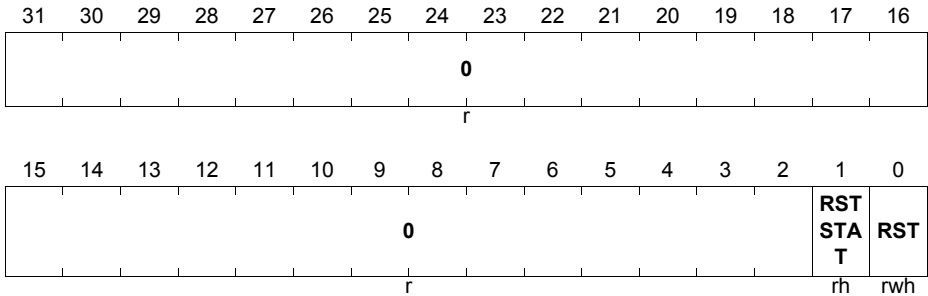
Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLR.CLR register bit.

Note: During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

Queued Synchronous Peripheral Interface (QSPI)

KRST0
Kernel Reset Register 0
(F4_H)
Reset Value: 0000 0000_H


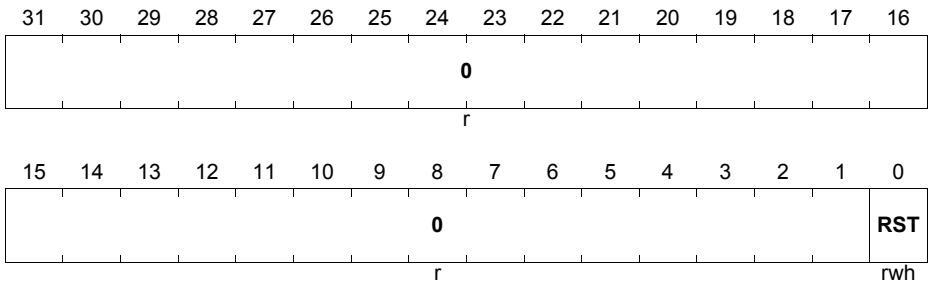
Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
RSTSTAT	1	rh	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits. 0 _B No kernel reset was executed 1 _B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST

Queued Synchronous Peripheral Interface (QSPI)

bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

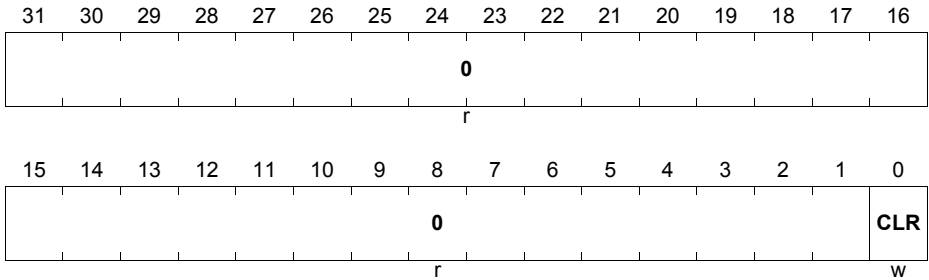
KRST1
Kernel Reset Register 1
(F0_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Status Clear Register (KRSTCLR)

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

Queued Synchronous Peripheral Interface (QSPI)

KRSTCLR
Kernel Reset Status Clear Register (EC_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Queued Synchronous Peripheral Interface (QSPI)

20.12.4.2 Interrupt Control Registers

The interrupts of the QSPI module are controlled by six service request control registers, located in the IR (Interrupt Router) module:

- QSPIxTX Transmit Interrupt
- QSPIxRX Receive Interrupt
- QSPIxERR Error Interrupt
- QSPIxPT Phase Transition Interrupt
- QSPIxHC High Speed Capture Interrupt
- QSPIxU User Interrupt

Queued Synchronous Peripheral Interface (QSPI)

20.13 High Speed Input Capture

The HSIC sub-module provides high precision measurement of relatively low speed input signals. It contains 15-bit timer can be configured to continuously measure:

- whole period (rising edge to rising edge),
- high time (rising edge to falling edge) or
- low time (falling edge to rising edge)
- whole period (falling edge to falling edge)

The resolution of the counter is f_{BAUD2}^{-1} . The result is provided in a capture register and an interrupt is raised after each capture.

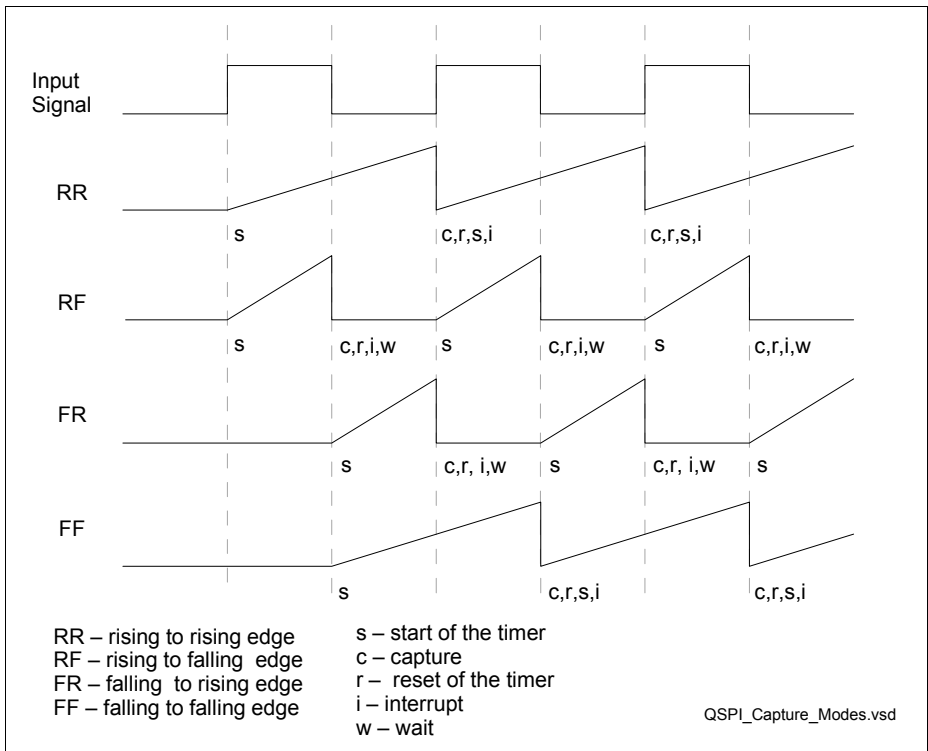


Figure 20-60 HSIC Modes of Operation

Each capture event causes the content of a 15-bit counter to be captured to the CAPCON.CAP bit field, see Figure 20-61.

Queued Synchronous Peripheral Interface (QSPI)

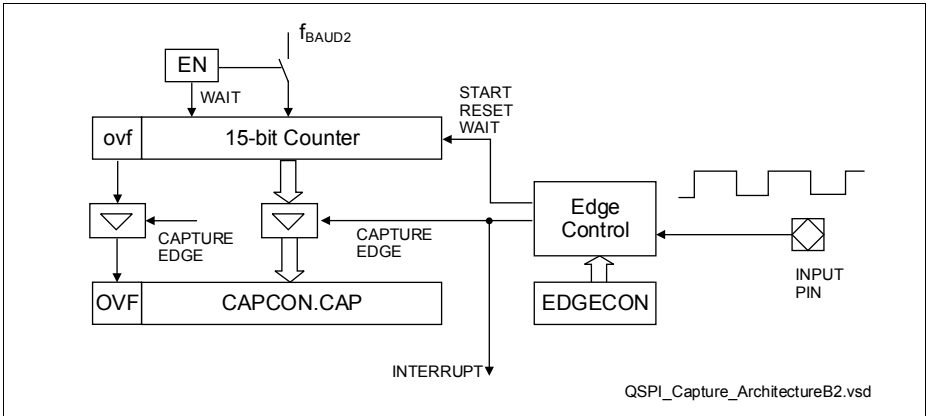


Figure 20-61 Capture Timer Architecture

The interrupt raised by the capture event is routed to the HC interrupt, see [Figure 20-62](#). The flag bit, set, clear and select bits are located in the **CAPCON** register.

The first interrupt after enabling the HSIC block by setting **CAPCON.EN** may deliver random **CAPCON.CAP** value, depending on the level of the input signal at the moment of enabling. Therefore, the software should discard the first value captured immediately after enabling the HSIC block.

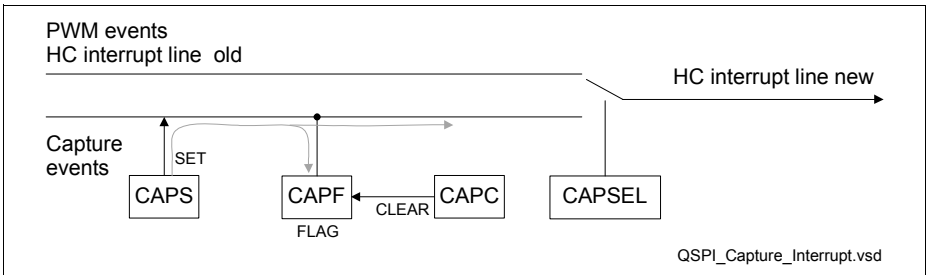


Figure 20-62 HSIC Interrupt Path

Queued Synchronous Peripheral Interface (QSPI)

Disable Request

Clearing the disable request bit **CLC.DISR** switches the clock of the whole QSPI module including the HSIC block on.

Setting the bit **QSPI2_CAPCON.EN** enables the HSIC sub-module. The counter first enters wait state, waiting for the first start edge as defined in the **QSPI2_CAPCON.EDGECON**. Afterwards, it operates according to the **Figure 20-60**.

If **CLC.DISR** is cleared when **QSPI2_CAPCON.EN = 1**, the counter is cleared to zero, waits for the next start edge (as defined in **QSPI2_CAPCON.EDGECON**) to start counting, and continues operating according to the **Figure 20-60**.

Debug Suspend

Suspend of the QSPI module causes suspend of the HSIC; separate suspend is not possible. Nevertheless, the targeted use cases are either HSIC timer active, or QSPI communication, but not both.

Active and enabled hard debug suspend signal stops the HSIC block immediately (QSPI kernel disabled). After the suspend signal is deactivated, the counter continues counting. Writing to a register in hard suspend state causes the HSIC block to receive a couple of clock cycles, which causes the counter to increase its value.

Active and enabled soft debug suspend signal stops the HSIC block immediately (QSPI kernel disabled). After the suspend signal is deactivated, the counter is cleared to zero, waits for the next start edge (as defined in **QSPI2_CAPCON.EDGECON**) to start counting, and continues operating according to the **Figure 20-60**.

Sleep Mode

Active and enabled sleep signal stops the HSIC block. After the sleep signal is deactivated, the counter is cleared to zero, waits for the next start edge (as defined in **QSPI2_CAPCON.EDGECON**) to start counting, and continues operating according to the **Figure 20-60**.

Reset

Each reset of the QSPI module causes reset of the HSIC sub-module; separate reset is not possible.

Queued Synchronous Peripheral Interface (QSPI)

20.13.1 HSIC Registers

20.13.1.1 Register Description

CAPCON

CAPCON is the only control register for the HSIC sub-module. It contains the configuration, control bits and the captured value of the counter.

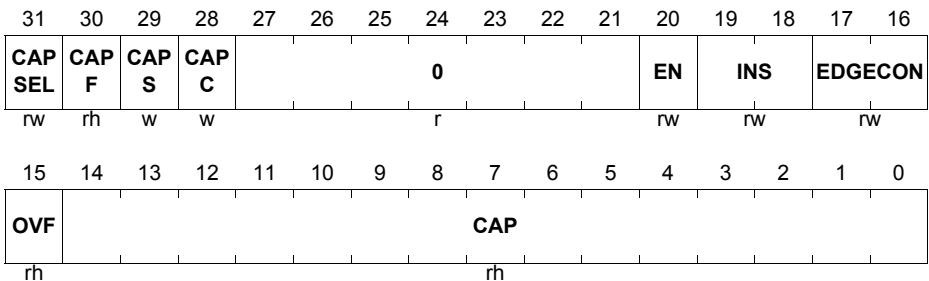
(>> [Table 20-2](#) register overview)

QSPI2_CAPCON

Capture Control Register (A0_H) Reset Value: 0000 0000_H

QSPI3_CAPCON

Capture Control Register (A0_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
CAP	[14:0]	rh	Captured Value Provides the latest capture value from the capture timer.
OVF	15	rh	Overflow Flag Signals if an overflow event of the capture timer occurred in the latest measurement. Shows a value of 1 if one or more overflows of the capture timer occurred during the latest measurement. Each capture event refreshes this bit with the new value.

Queued Synchronous Peripheral Interface (QSPI)

Field	Bits	Type	Description
EDGECON	[17:16]	rw	Edge Configuration Configure the capture mode of the counter, defining which edge starts the counting, which edge captures its contents. 00 _B RR mode (rising edge to rising edge) 01 _B RF mode (rising edge to falling edge) 10 _B FR mode (falling edge to rising edge) 11 _B FF mode (falling edge to falling edge)
INS	[19:18]	rw	Input Selection Selects one of four input lines for the capture signal. 00 _B INA selected 01 _B INB selected 10 _B INC selected 11 _B IND selected
EN	20	rw	Enable Bit of the Capture Timer Enables the capture timer clock. Enabled timer waits for the next start edge as defined in EDGECON, before it starts counting. 0 _B disabled 1 _B enabled
CAPC	28	w	Capture Flag Clear Writing 1 to this bit clears the flag CAPF. Writing 0 has no effect. Returns 0 on read.
CAPS	29	w	Capture Flag Set Writing 1 to this bit sets the flag CAPF. Writing 0 has no effect. Returns 0 on read.
CAPF	30	rh	Capture Flag Indicates if a capture event has occurred since the last clear event by the bit CAPC.
CAPSEL	31	rw	Capture Interrupt Select Bit Select if the capture events are routed to the HC interrupt line. 0 _B not routed (capture interrupt disabled) 1 _B routed (capture interrupt enabled)
0	[27:21]	r	reserved

20.13.2 HSIC Implementation

HSIC block is available in the QSPI2 and QSPI3.

Queued Synchronous Peripheral Interface (QSPI)**20.13.2.1 Connection to the pins**

For the connections to the pins, see the pinning chapter.

The unconnected input lines are tied to zero.

Queued Synchronous Peripheral Interface (QSPI)

21 High Speed Serial Link (HSSL)

The HSSL module provides point-to-point communication of both single data values and of large data blocks, called streams. The communicating devices can be complex microcontrollers, or a microcontroller and a device with only basic execution capabilities. There are four channels to transfer single values to/from target. They support direct writing of 8/16/32 bit data from the initiator into a target's register, as well as reading a value from a target, performed by a modules internal master on the target side. For transferring large data blocks there is a channel containing FIFOs. The HSSL module implements Transport Layer tasks and hands over the data to another module which provides Data Link Layer and Physical Layer services, data serialization and transmission. All transfers are protected by safety features like CRC and timeout.

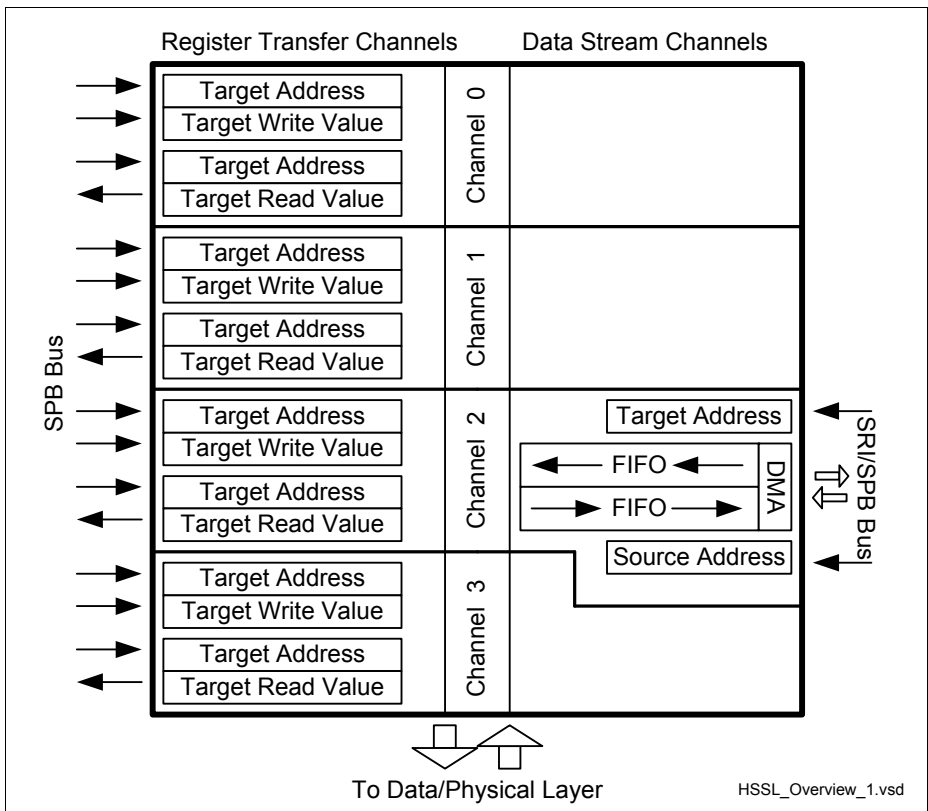


Figure 21-1 Block Diagram HSSL

High Speed Serial Link (HSSL)**Features**

- Connects microcontroller to microcontroller or to any digital device
- Writing a single 8 / 16 / 32 bit data value into the register of a target device
- Reading single data from an 8 / 16 / 32 bit register of a target device
- Support of 32-bit address range
- Transfers protected by CRC16
- Programmable time outs for detection of blocked answer transfers
- Automatic frame transfer ID generation for detection of dropped frames
- Support of DMA driven multiple register write / read transfers
- Efficient transmission and reception of large data blocks / streams
- Acknowledge for command and stream frames to reduce latency of error detection
- Two stage FIFOs for transmitting and receiving streaming data
- Automatic FIFO flush when entering the run mode, for error handling
- Write protection by an external Memory Protection Unit MPU
- Remote trigger of event / interrupt in the target device by the initiator
- Identification of the target by the JTAG ID number
- Feature set identification of the HSSL module possible by using the JTAG ID number
- Access protection from an external master

Description

The HSSL module generates all necessary protocol formats in order to enable two devices to communicate to each other. Two basic modes are supported.

In register transfer mode the transmitter (“sending device”) provides an address and a data value by writing them into dedicated registers of the HSSL module. The HSSL module puts it into an envelope (called frame) and forwards it via physical layer to the target device. The target takes the information out of the frame and writes the received data into the address which came in the same frame as the data. If the sender wants to read the content of a register or a memory location, it simply sends only the address. The target device reads the value from the specified location and replies it to the sender by transferring it through an answer frame, which has a format of it’s own.

The transfer of register / memory contents is protected by several security levels. A 16 bit CRC value (called CRC16) is attached to the end of each frame. Each transfer process triggers a watchdog, which makes sure that the communication happens within a deterministic schedule. The target device itself offers the capability to define access windows, which allow reading or writing only from / into certain address ranges.

Each frame must be acknowledged by the target by sending an appropriate response frame. If the response frame does not come inside a predefined time window, the initiator module triggers a timeout interrupt.

In data stream mode data blocks of user defined size can moved from the memory of the sending device into the memory of the target device. A DMA master, which can be configured with any source address and source length, autonomously fetches data, breaks it into smaller blocks of programmable size, and puts it into a FIFO. The lower level companion module of the HSSL module empties the FIFO from the “other side” to the physical lines, which connect two devices. In the target device receives a structure, which is inverse to the sender system, the data blocks, puts them into a FIFO, which is in turn emptied by a DMA master into the target memory.

All streaming data transfers are subject to the protection features which are available for the register transfer mode.

After module reset, most registers are only released for read for remote access. They have to be unlocked by the local SW.

In order to ensure consistent initialization of the communication channel there is a possibility to determine the feature set of the module by using the JTAG ID of the device, which uniquely identifies the available HW options of the module. ASIC versions of the HSSL interface may only employ stripped down instances.

High Speed Serial Link (HSSL)

21.1 Lower communication layers module (HSCT)

The lower level companion module/logic controls the lower communication layers positioned between two devices, the data and the physical level.

The chip to chip interface employs a digital interface for inter chip communication between a master IC and a slave IC. The interface is capable of running in a master or in a slave mode. During configuration phase the role of the Interface (master or slave) has to be defined. It is not intended to change the system role during an application.

The interface consists of a full-duplex RX and TX high-speed data interface based on double ended differential signals (in total 4 lines) and a master clock interface (SYS_CLK). The master IC owns the crystal and provides the clock to the slave. The interface reset is derived from the System reset and provided by chip internal reset signaling.

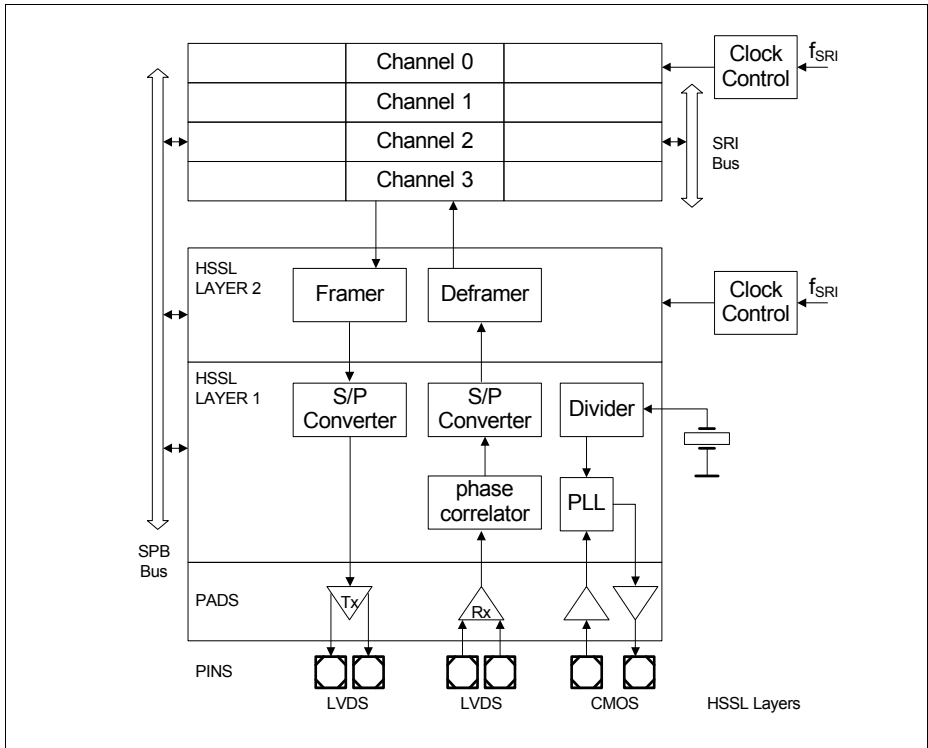


Figure 21-2 Physical Layer

High Speed Serial Link (HSSL)**Features:**

- Transmission of symbol information from the master IC to the slave IC
- Receive symbol information sent from the slave IC to the master IC
- System Clock provided by the master IC (crystal owner) to the slave IC
- Exchange of control information in both directions
- Clear To Send indication in both directions
- Unsolicited status in both directions
- Regular data transfer in both directions based on data channels
- Master IC data transfer speed 5 MBaud and 320 MBaud
- Three slave IC data transfer speeds available based on 20 MHz SYS_Clk:
 - 5 MBaud (low speed)
 - 20 MBaud (medium speed)
 - 320 MBaud (high speed)
- Two slave IC data transfer speeds available based on 10 MHz SYS_Clk (lower EMI):
 - 5 MBaud (low speed)
 - 320 MBaud (high speed)
- The interface is based on IEEE 1596.3 LVDS IOs
- To reduce the voltage swing a configuration option is available

21.2 HSSL Protocol Definition

This section describes the high level protocol definition of the HSSL interface.

The HSSL communication normally consists of two stage transactions, sending a command and an receiving a response. It involves two participants:

- an initiator, which starts a transaction by sending a command
- a target, which responds to the command

21.2.1 List of Abbreviations, Acronyms, and Term Definitions

ACK - Acknowledge Frame

NACK - Not Acknowledge Frame (Target Error Frame)

CRC - Cyclic Redundancy Check

HSSL - High Speed Serial Link

TO - Time Out

TT - Transaction Tag

TTERR - Transaction Tag Error

SPB - Serial Peripheral Bus

SRI - Shared Resource Interconnection

SoC - System on Chip

Initiator - Device that starts a HSSL transaction by sending a command frame

Target - Device that responds to a command frame sent by an initiator

21.2.2 Frame Types

The HSSL module generates a serial frame by:

- taking the payload delivered by a CPU or a DMA
- first wrapping it in a HSSL frame by adding header and CRC fields, and then
- wrapping the HSSL frame in a HSCT frame by adding Sync, Header and End bit fields (see [Figure 21-3](#))

High Speed Serial Link (HSSL)

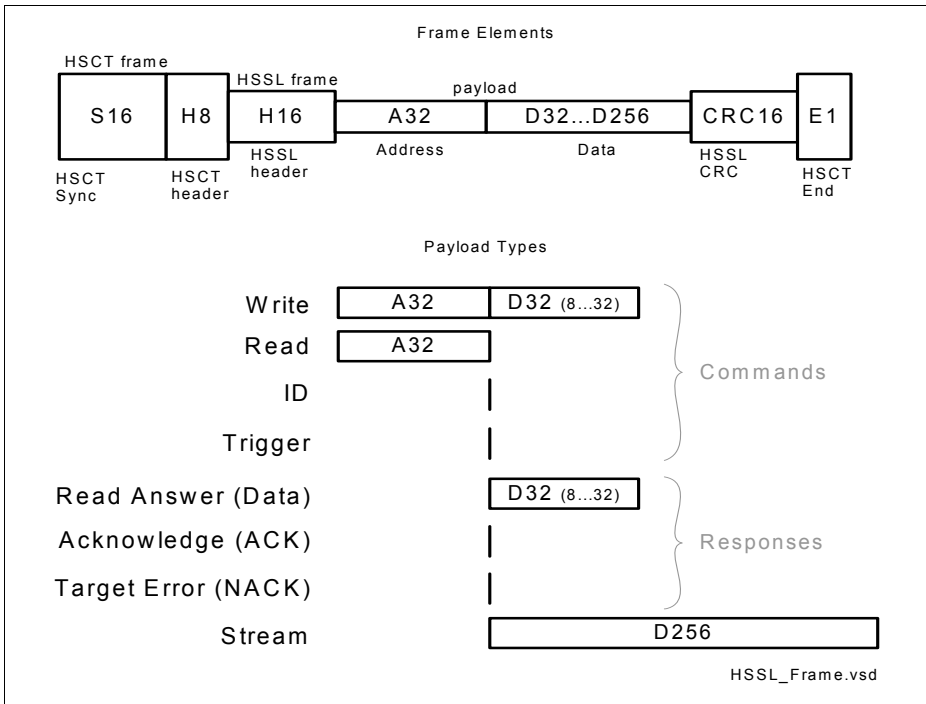


Figure 21-3 HSSL Frame Types

There are several types of frames:

Command Frames

Command frames are sent by the initiator to request an action by the target. There are four types of command frames:

- **Write Frame**
 - Used by the initiator to request the target controller to write the data to the address, both provided by the initiator.
- **Read Frame**
 - Used by the initiator to request the target controller to read and deliver a content from an address from its address space.
- **Read ID Frame**
 - Used by the initiator to request the target controller to read and deliver a content uniquely defining the target device. The answer is a standard Read Data Answer Frame.
- **Trigger Frame**

High Speed Serial Link (HSSL)

- Used by the initiator to trigger the trigger interrupt in the target module.

In this document, the command frames are sometimes referred to as WRT frames.

Response Frames

Response frames are sent by the target either to deliver the requested data or to report a successful or unsuccessful completion of a request. There are three types of response frames:

- **Read Data Answer Frame**
 - Used by the target to deliver the data, requested by the initiator with a read command frame. It is frequently referred to as a Data frame.
- **Acknowledge Frame**
 - Used by the target to confirm the completion of a write or trigger command. It is frequently referred to as an ACK frame.
- **Target Error Frame**
 - Used by the target to report the unsuccessful attempt to complete a write or read command. It is frequently referred to as a NACK frame.

Stream Frame

Used by the initiator controller for sending data streams. It behaves similar to a write command frame which contains long 256-bit data block, is acknowledged in the same way, but the acknowledge uses sliding window with depth of 2 (instead of 1 as for the simple command frames).

21.2.2.1 Frame and Payload Lengths

Each type of frame has a single data length associated with it. The length of the frame depends on the length of the payload. Different frame types have different payload lengths, but the payload length for one type is constant for all variations of the frame type. For example all data frames have the same length, although they can carry 8, 16 or 32 bit data. This is due to the fact that the data field in the frame is always 32 bit long, and if the data itself is shorter, it is copied several times to fill in the whole field. The same behavior is valid for the read answer frames.

The rest of the frame, consisting of sync and header fields, has always the same length (see [Figure 21-7](#)).

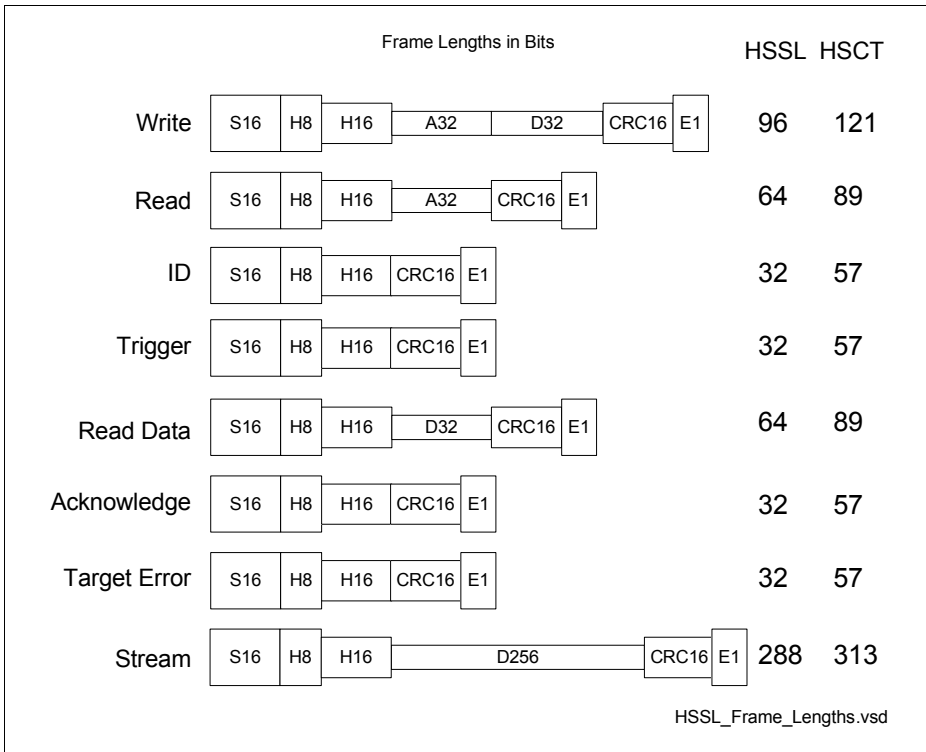


Figure 21-4 Frame Lengths

21.2.2.2 Data Types

There are four types of HSSL data:

- 8-bit
- 16-bit
- 32-bit
- 256-bit stream block

In the case of 8/16/32 bit data types, the transmitted number of bits is always 32, where the shorter data types are copied several times to fill up this length (see [Figure 21-5](#)).

High Speed Serial Link (HSSL)

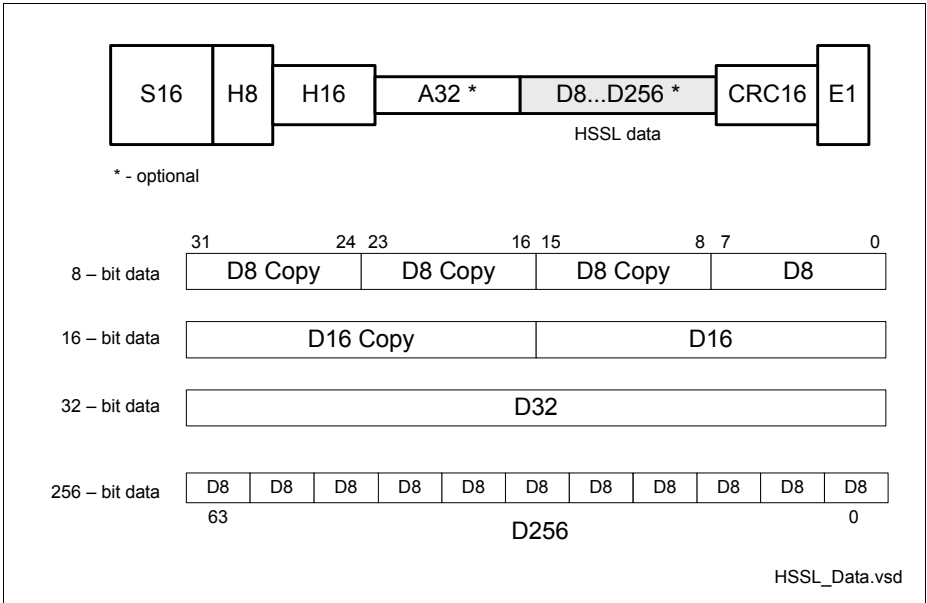


Figure 21-5 Data Types

21.2.2.3 Cyclic Redundancy Check Field - CRC

The HSSL protocol uses the CRC-16-CCITT polynomial: $x^{16} + x^{12} + x^5 + 1$ with the following standard properties:

- seed: 0xFFFF
- calculation direction: MSB first (header msb to lsb, then payload msb to lsb)
- CRC result direction: MSB first

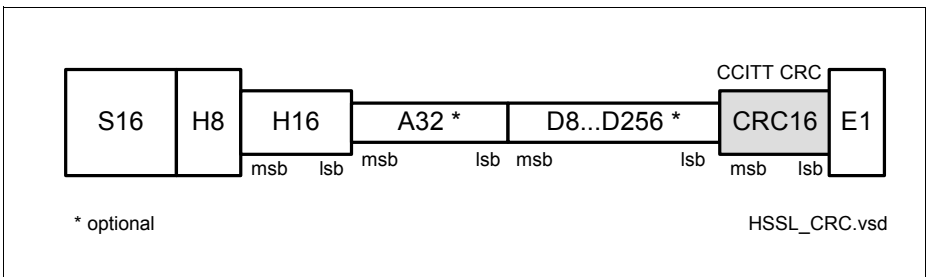


Figure 21-6 CRC Field

21.2.2.4 Header Structure

The HSSL header contains the protocol information that is required additionally to the raw payload data. The HSSL header describes the payload data regarding the length and the type, and carries additional information like channel number code and transaction tag. It also carries stand-alone information in case of frames without payload, like acknowledge and trigger frames.

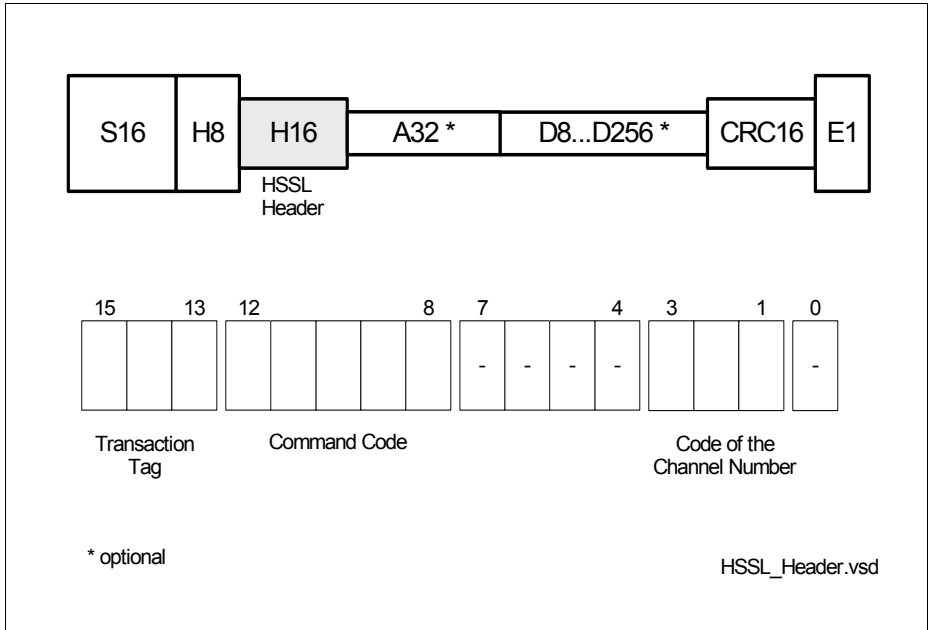


Figure 21-7 Header Types

Note: The unused bits in the header are padded with 0.

Table 21-1 Mapping of HSSL to HSCT channel codes

HSSL Channel Number	HSSL Channel Number, Binary Code	HSSL Channel Number, Special Code	HSCT Channel	HSCT Channel Code
0	000	100	A	0100
1	001	101	B	0101
2	010	110	C	0110
3	011	111	D	0111
4 (reserved)	100	000	E	1000
5 (reserved)	101	001	F	1001
6 (reserved)	110	010	G	1010
7 (reserved)	111	011	H	1011

Note: The HSSL module supports four channels, 0 to 3.

Table 21-2 List of Command Codes

Command Code	Command Description	HSSL Frame Size
00000	Read 8 bit	64
00001	Read 16 bit	64
00010	Read 32 bit	64
00011	RFU (Reserved for Future Use)	-
00100	Write 8 bit with ACK (Acknowledge)	96
00101	Write 16 bit with ACK	96
00110	Write 32 bit with ACK	96
00111	RFU	-
01000	ACK OK	32
01001	ACK Target Error	32
01010	Read Answer OK	64
01011	RFU	-
01100	Trigger with ACK	32
01101	RFU	-
01110	RFU	-

Table 21-2 List of Command Codes

Command Code	Command Description	HSSL Frame Size
01111	RFU	-
10000	RFU	-
10001	RFU	-
10010	Read JTAG ID 32-bit	32
10011	RFU	-
10100	RFU	-
10101	RFU	-
10110	RFU	-
10111	Stream Data with ACK (32 Byte Data)	288
11000	RFU	-
11001	RFU	-
11010	RFU	-
11011	RFU	-
11100	RFU	-
11101	RFU	-
11110	RFU	-
11111	RFU	-

21.2.3 Single and Block Transfers

Single frame transfers are always performed expecting an acknowledge.

21.2.4 Streaming Interface

Stream frame transfers are always performed with acknowledge.

There are only write streams possible. Read streams are not possible.

21.2.5 Sliding Window Protocol

For flow control the HSSL module implements two variants of the “Sliding Window” protocol regarding the sliding window depth:

- depth of one used for command frames
 - The depth of one means that a single timer tracks the arrival of the corresponding response frame. A new command can be sent only after the response to the previous command has been received, or a timeout occurred

High Speed Serial Link (HSSL)

- depth of two used for streaming frames
 - the depth of two means that there are two timers tracking the acknowledge times for the two latest streaming frames. In this case, a second stream frame can be sent although an acknowledge / timeout to the previous stream frame has not been received yet. After the second stream frame, no third stream frame can be sent before the first one has been acknowledged

An example of communication using sliding window with depth of one protocol is shown in **Figure 21-8**. Here, the initiator sends commands originating from one and the same channel, for example channel 0 (labeled with the capital letter “A”), and the corresponding acknowledge frames from the target are labeled with the small letter “a”.

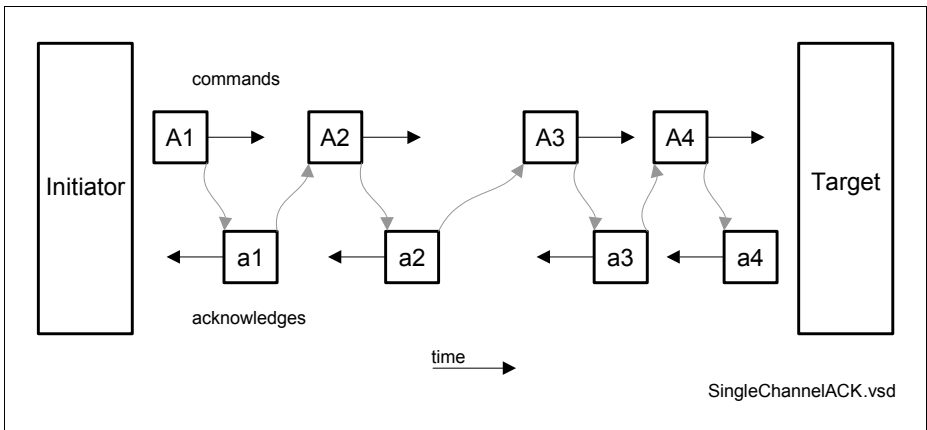


Figure 21-8 Flow Control, Single Channel, Sliding Window With Depth of One

The sliding window protocol requires capability of sending acknowledge frames in the target and timeout timers in the initiator. Every time the target receives a frame without detecting an error, it sends a response / acknowledge to the initiator. If the target detects a frame with a CRC error, it does not send an acknowledge. The missing acknowledge causes a timeout event in the initiator channel.

If the command frames are longer than the acknowledge frames plus the reaction time, like for example in case of streaming, back to back transfers are easily possible, see **Figure 21-9**.

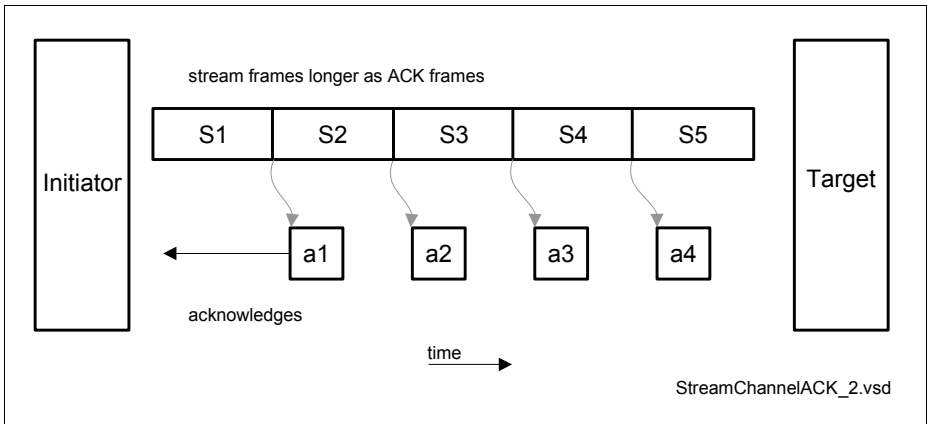


Figure 21-9 Flow Control, Streaming Frames

21.2.6 Error Management

The HSSL protocol defines four types of errors:

- Time Out
- Transaction Tag Error
- CRC Error
- Target Error

Time Out Error

A time out error is detected at the initiator side, if the expected ACK frame was not received within the expected time window. This can occur if a frame had been sent by the initiator, and the target detected an CRC error and did not answer with an acknowledge, or the connection between the initiator and the target is physically damaged in one or the other direction.

Transaction Tag Error

A transaction tag error occurs at the initiator side, if instead the expected ACK frame with the expected TAG number, an acknowledge with an unexpected transaction tag was received. This would indicate a missing frame or missing acknowledge. Transaction tag error generate frames that pass the CRC checking stage.

CRC Error

A CRC Error can occur:

- at the target side, in which case:

High Speed Serial Link (HSSL)

- the CRC error flag is set
- the received command frame with a CRC error is discarded
- no acknowledge frame is sent and
- a channel unspecific EXI error interrupt is generated, if enabled.
- at the initiator side, in which case:
 - the CRC error flag is set
 - the received response frame with a CRC error is discarded
 - a channel unspecific EXI error interrupt is generated, if enabled

Both scenarios lead to a time out at the initiator side.

In both cases the CRC error flag is set at the side receiving the erroneous frame (either initiator or target) and an interrupt is generated, if enabled.

For the purpose of CRC error injection, a bit field with an XOR mask used for toggling the bits of the calculated CRC is provided, see **CRC.XORMASK**. In order to switch on this feature, the E (Endinit) protected bit must be set by the application software. The error injection only influences the generation of the transmitted CRC.

Target Error

A Target Error can occur at the target side, when accessing the target memory a bus error or memory protection error occurs. In such a case, the target responds with an NACK frame indicating the error.

21.2.7 Shift Direction

In general, the HSSL interface makes a copy of a memory location or block from the address space of one microcontroller into the address space of the other microcontroller.

The HSSL module delivers 32-bit parallel data.

The transfer over HSCT serial link is done by using MSB first shifting at 32-bit level.

The overview of the path from the memory to the serial bus is shown in the **Figure 21-10**.

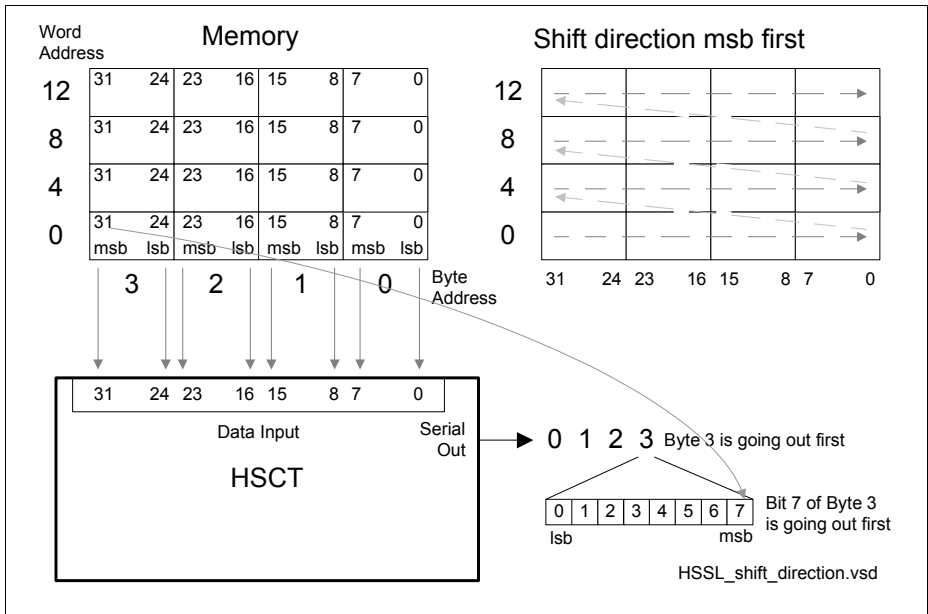


Figure 21-10 Shift Direction in Case of HSCT Layer

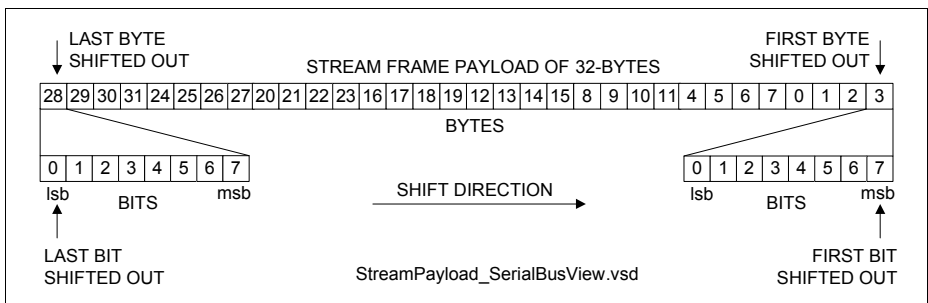


Figure 21-11 Stream Frame in Case of HSCT Layer

High Speed Serial Link (HSSL)

21.3 HSSL Implementation

This section describes the implementation of the HSSL protocol.

21.4 Overview

The HSSL module is connected to the SRI bus using a master interface and to the SPB bus using a BPI slave interface.

Additionally, it is connected to the SPB bus using a master interface.

The SRI master interface is capable of performing single and burst reads and writes on the SRI bus. Additionally, the HSSL kernel provides logic for performing streaming communication. The write and read accesses are always performed in User Mode.

The SPB BPI slave interface is used by an SPB master (DMA, CPU, HSSL SPB Master) for writing the HSSL registers, thus configuring the module and performing single read and write operations.

The SPB master interface is capable of performing single read operations, write operations, and streaming communication using BTR4 burst accesses on the SPB bus. The SPB master accesses are always performed in User Mode.

The HSSL module to use the SRI or the SPB master depending on the address of the access. Accesses to the SPB address space are automatically routed to the SPB master, accesses to the SRI address space are automatically routed to the SRI master. The address window $F000\ 0000_H - F7FF\ FFFF_H$ is used for the SPB bus, otherwise SRI.

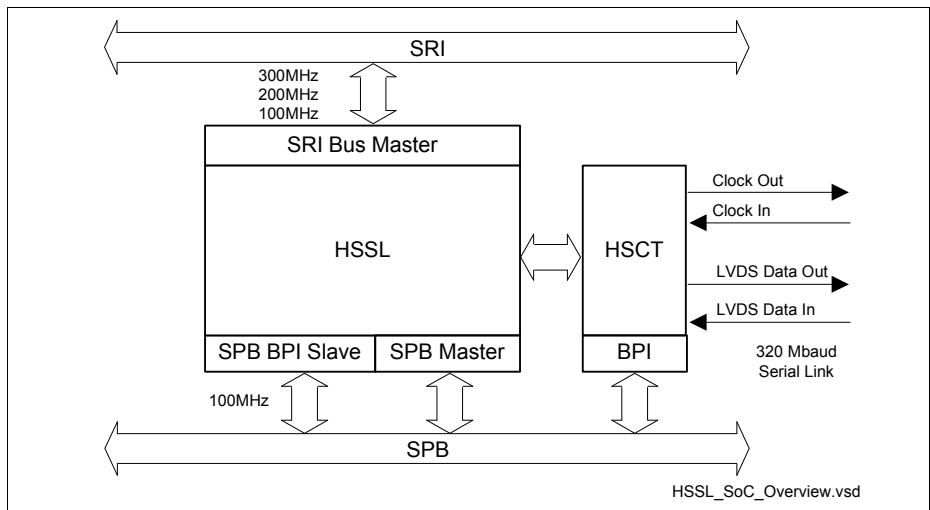


Figure 21-12 HSSL Subsystem Overview

High Speed Serial Link (HSSL)

21.4.1 HSSL Module Operation

The HSSL module consists of separate transmitter and receiver, each containing four channels.

The transmitter contains an arbiter block and a wrapper block, which are common for all channels. The arbiter provides fixed arbitration between the channels which are ready to send when the serial link is available, with channel 0 having the highest priority, and the channel 3 having the lowest priority. The wrapper block generates the CRC.

The receiver unwraps the HSSL frame, discards the frames with a CRC error, and distributes the error free commands and acknowledges to the channels.

A common prescaler generates the time basis for all channels, in particular for their timeout timers.

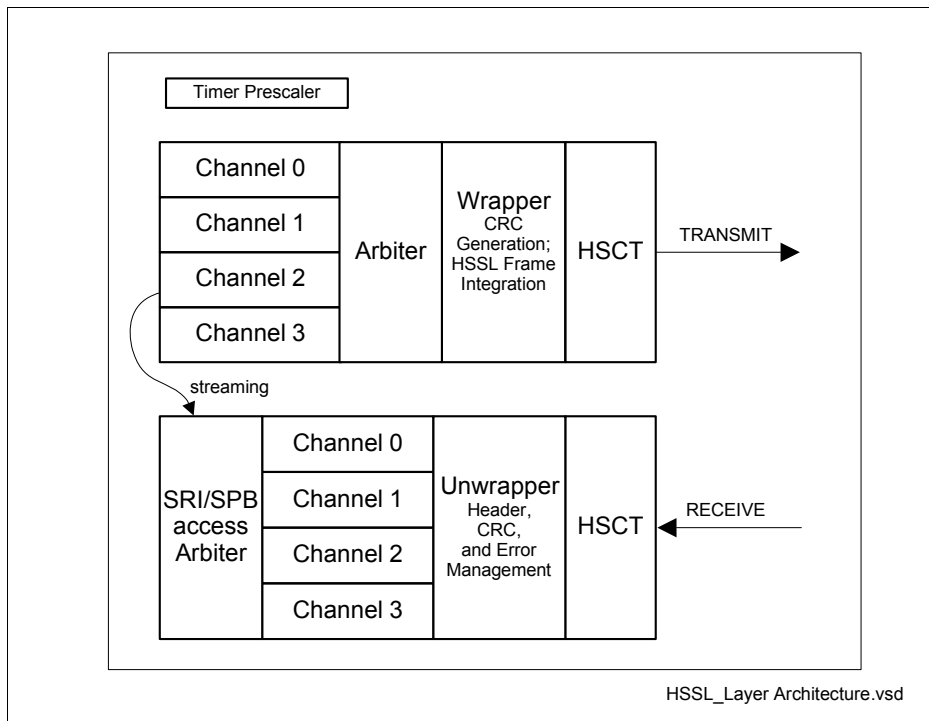


Figure 21-13 HSSL Layer Architecture

21.4.1.1 Frame Transmission Prioritisation

When more than one frames are pending, an arbitration is performed between them.

Each channel activates two flags: initiator command flag and response flag: channel 0 I0 and R0, channel 1 I1 and R1...

Between the response and command frame request types, the response request type has higher priority. Among the requests of a same type, lower channel number has higher priority, see **Figure 21-14**. A pending frame is indicated with a corresponding request flag in the register **QFLAGS**.

The SRI/SPB bus master sets the response requests ("R" flags).

The software write accesses via the SPB bus set the initiator requests ("I" flags).

The transmit arbiter clears the "I" and "R" flags of the arbitration winner.

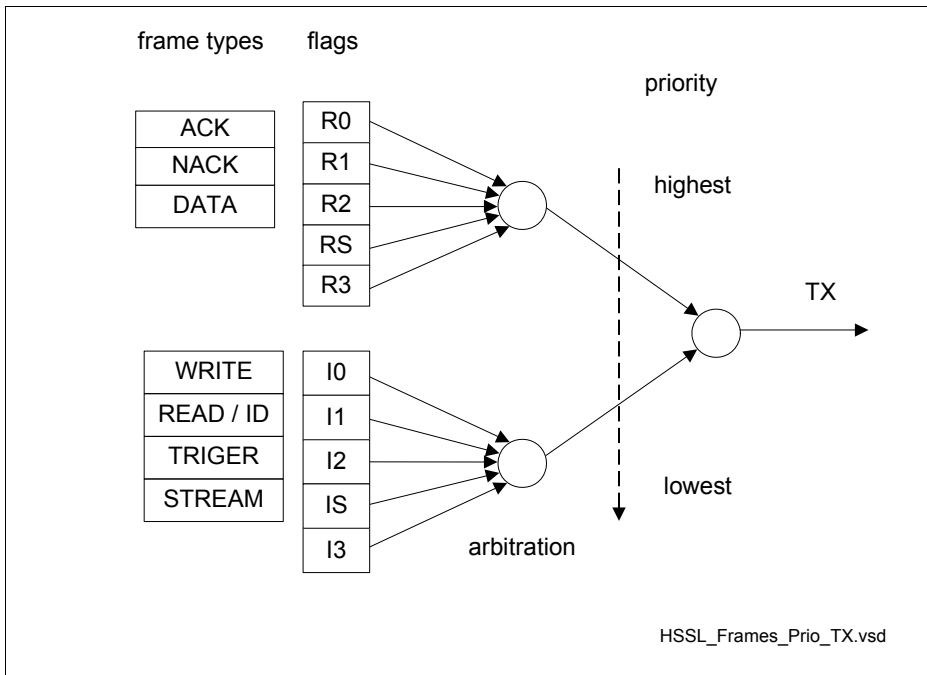


Figure 21-14 Frame Priorities in the HSSL module

21.4.1.2 Received Frame Management and Command Execution

After a frame passes the CRC check, it is checked for its type. A received command frame sets a request flag for the SRI/SPB master, a received response frame resets the expect flag and stops the appropriate channel timeout, see [Figure 21-15](#).

The target flags “T” are set by the frame distributor, and cleared by the arbiter of the SRI/SPB master.

The expect flags “E” are set by the Tx arbiter, and reset by the Rx distributor.

The ISB and ISF flags operate together to control the streaming process. The ISB flag (Initiator Stream Block flag) is set by the software to enable the streaming. The ISF flag (Initiator stream FIFO flag) is set/cleared by the TXFIFO according to its filling level.

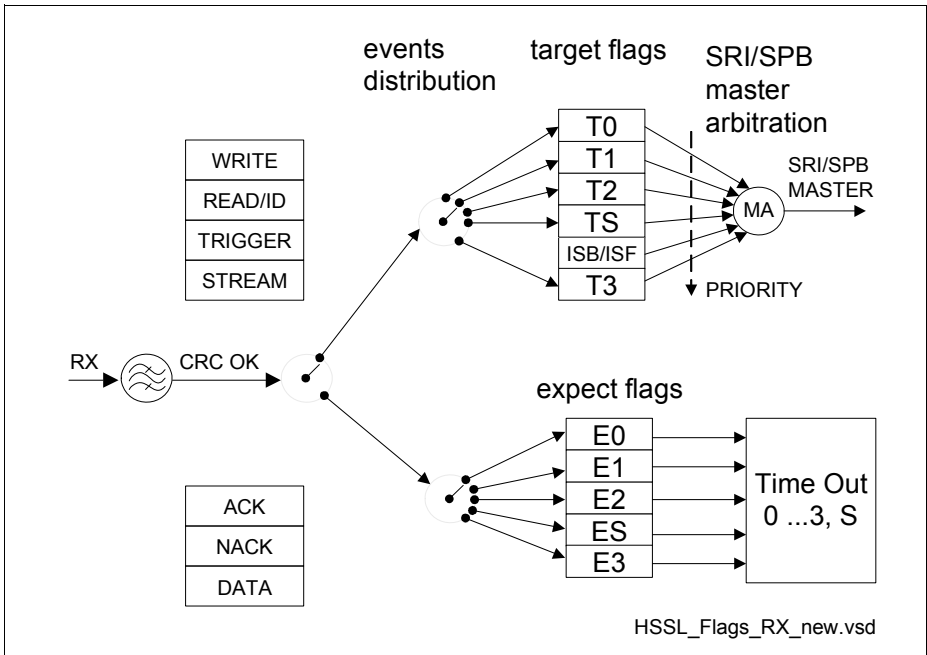


Figure 21-15 Received Frames Management

The CRC and Transaction Tag error flags are located in the **MFLAGS** register.

An overview of a full communication cycle over all sections of a channel (initiator, target or receive, transmit) is shown in [Figure 21-16](#).

See also the **QFLAGS** register description.

If the channel does not have a command pending and does not expect a response, it is ready.

High Speed Serial Link (HSSL)

Note: In case of a trigger command frame, the target command distributor sets the corresponding Rx flag, and not the corresponding Tx flag, due to the fact that the service request is triggered directly, and not by the SRI/SPB master.

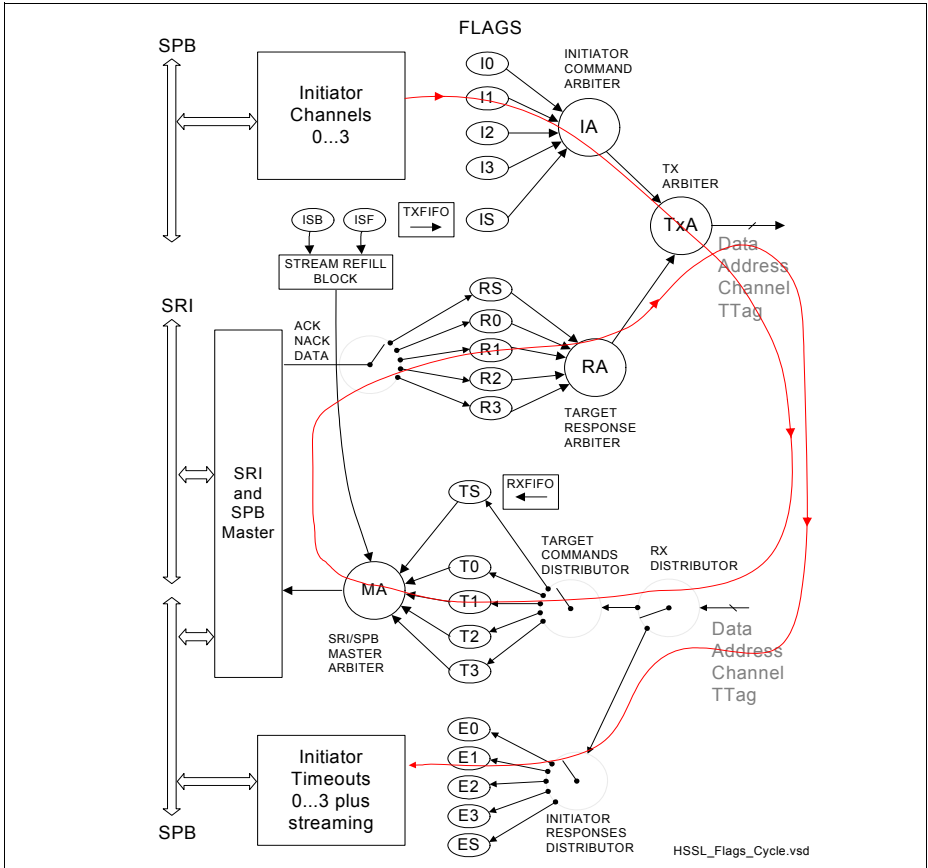


Figure 21-16 Request Flags Overview and a Communication Cycle Example

Table 21-3 Set and Clear of Request Flags

	I	R	T	E	ISB	ISF
SPB Software Write	set	-	-	-	set	-
Tx Arbiter	clear	clear	-	set	-	-
Rx Distributor	-	- ¹⁾	set	clear	-	-

High Speed Serial Link (HSSL)

Table 21-3 Set and Clear of Request Flags

	I	R	T	E	ISB	ISF
SRI / SPB Master	-	set	clear	-	-	-
Stream Refill Block	-	-	-	-	clear	clear
TXFIFO	-	-	-	-	-	set

- 1) In case of a trigger command, an R flag is set by the Rx Distributor instead of a T Flag, because a trigger command is not executed by the SRI/SPB master.

21.4.2 HSSL Channel Architecture

The HSSL module contains four channels. The functionality of each channel can be subdivided in two ways:

- transmitter and receiver functionality
 - The transmit functionality generates an appropriate header for each frame
 - The receiver generates an appropriate acknowledge header with the tag and channel number code. The read and writes are performed by the SRI/SPB master. Therefore, the module can send back to the transmitter a target error frame if an SRI/SPB bus error occurs and the transfer on the bus can not be executed.
 - The timeout monitoring on one hand and the acknowledge management on the other hand involve both the transmitter and receiver channels
- initiator and target functionality
 - The initiator functionality transmits write, read or trigger command frames in random sequence, after the previous command has been responded to.
 - Generation/comparison of a Transaction Tag per channel is an initiator functionality
 - The timeout management is a pure initiator functionality.
 - The target functionality is to receive commands, executes them, and generates and transmits responses.

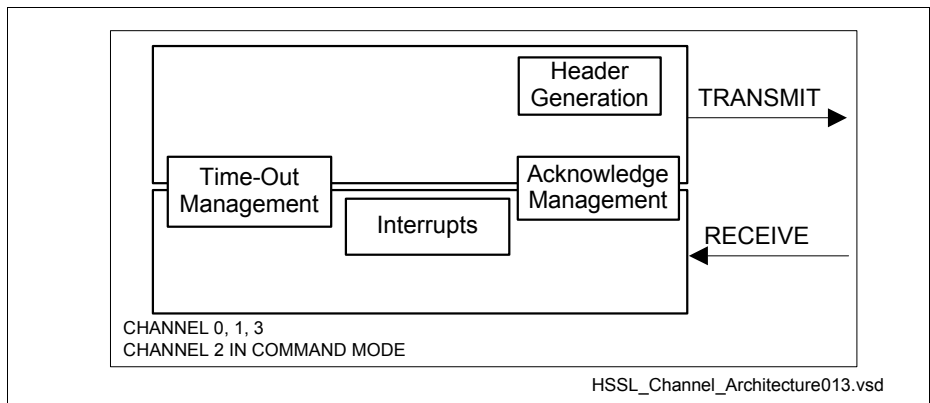


Figure 21-17 Architecture of Channels 0,1, 3 and Channel 2 in Command Mode

Channel 2 can be switched between command and streaming mode of operation. In single mode a single data register is used and in streaming mode a FIFO is used.

Channel 2 contains one initiator and one target address counter which are used to fill a memory range with data. Each counter contains two start address registers and one stream frame count register.

At reaching the predefined frame count, a wrap around is automatically executed.

High Speed Serial Link (HSSL)

The granularity of the start address is 256-bit.

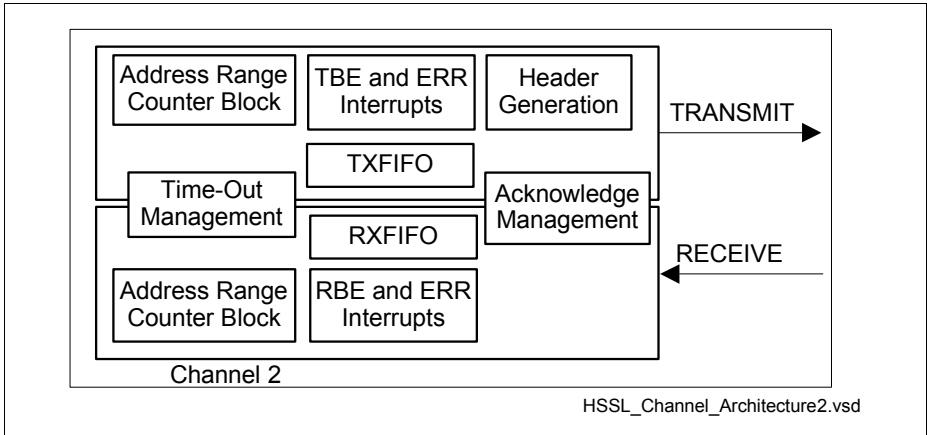


Figure 21-18 HSSL Architecture of Channel 2

A classification of the various channel functions is shown in [Figure 21-19](#).

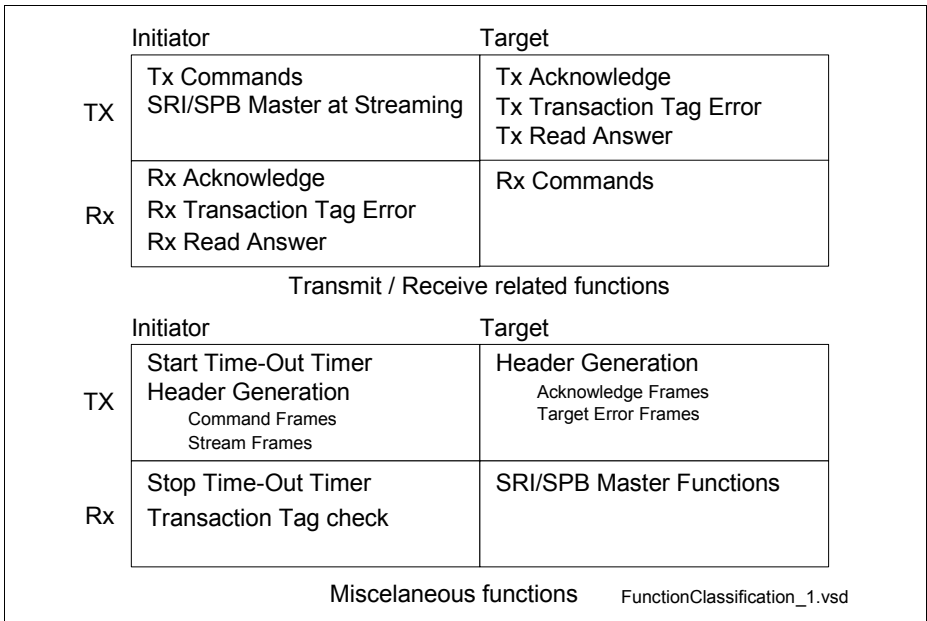


Figure 21-19 Overview of Channel Functions

High Speed Serial Link (HSSL)

21.4.3 Acknowledge Responses

A read or write command is executed by the SRI / SPB master. The SRI / SPB master reads or writes to the SRI / SPB bus and signals either a successful transaction or an error. The are the following cases:

- When the SRI master writes or reads an SRI memory, both cases are acknowledged so that the HSSL module receives a feedback in any case.
- When the SPB master writes or reads an SPB module or memory, both cases are acknowledged so that the HSSL module receives a feedback in any case.

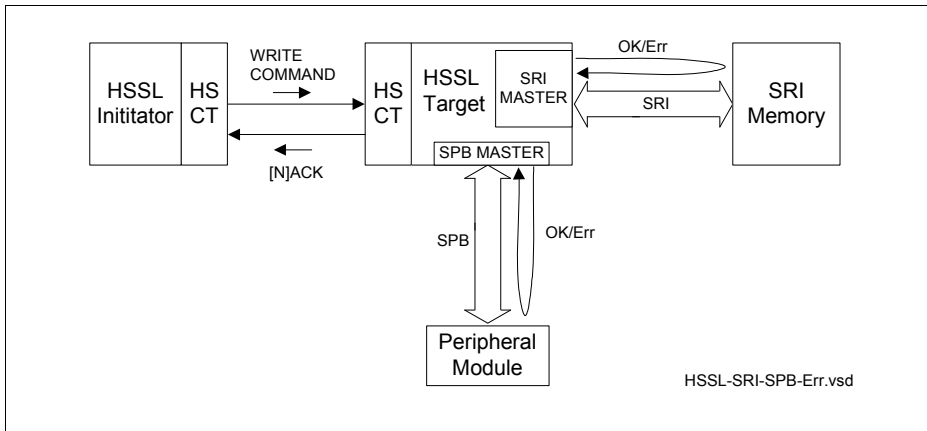


Figure 21-20 Bus Error Feedback Paths

21.4.4 Cross Dependencies Between the Frame Types

For a channel in a target role, there are cross-dependencies between command frame arrival and triggering of an ACK or NACK frame for a channel. Each command frame received with correct CRC of types Write, Stream, or Trigger Frame will result in either ACK or NACK (target error) response frame. Each Read Frame received with correct CRC will result either with Data Frame (Read Response Frame) or with NACK (Target Error Frame), see [Figure 21-21](#).

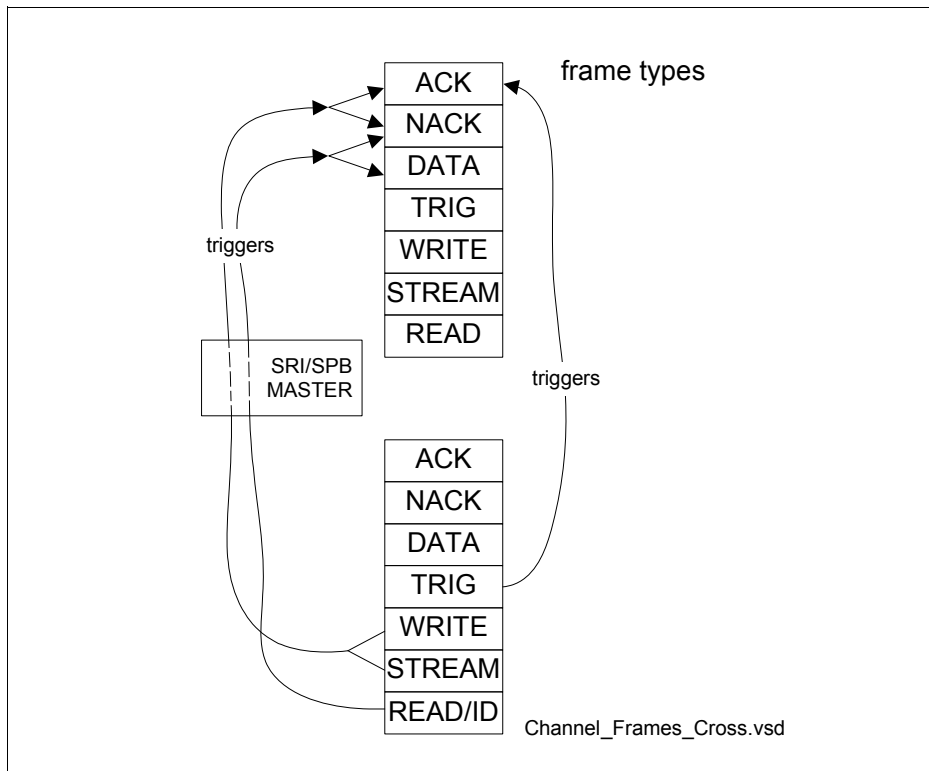


Figure 21-21 Channel 2 Cross Dependencies Between Stream and Command Frames and their Responses in a Target Role

Additionally, generating a response frame involves copying of the transaction tag of the command frame into the response frame. The response frame is triggered by an SRI/SPB master signalling either a successful SRI/SPB bus transaction or an SRI/SPB bus error. Only a Trigger Command Frame triggers immediate acknowledge, because it results in an interrupt flag being set, and not in a bus transaction.

High Speed Serial Link (HSSL)

On the initiator side, there is a forward looking cross dependency between a command frame and the expected response, see **Figure 21-22**. Setting a transmission request flag for a certain command is accompanied in parallel with:

- capturing the current transaction tag in the bit field **ICONx (x=0-3).CETT**.
- generating the next transaction tag by incrementing a three bit counter per channel, the **ICONx (x=0-3).ITTAG**.
- starting the timeout counter at the moment when the command wins the arbitration
- setting an expect tag indicating that the timeout is running and a response frame is expected. See **MFLAGS.Ex** bit fields.

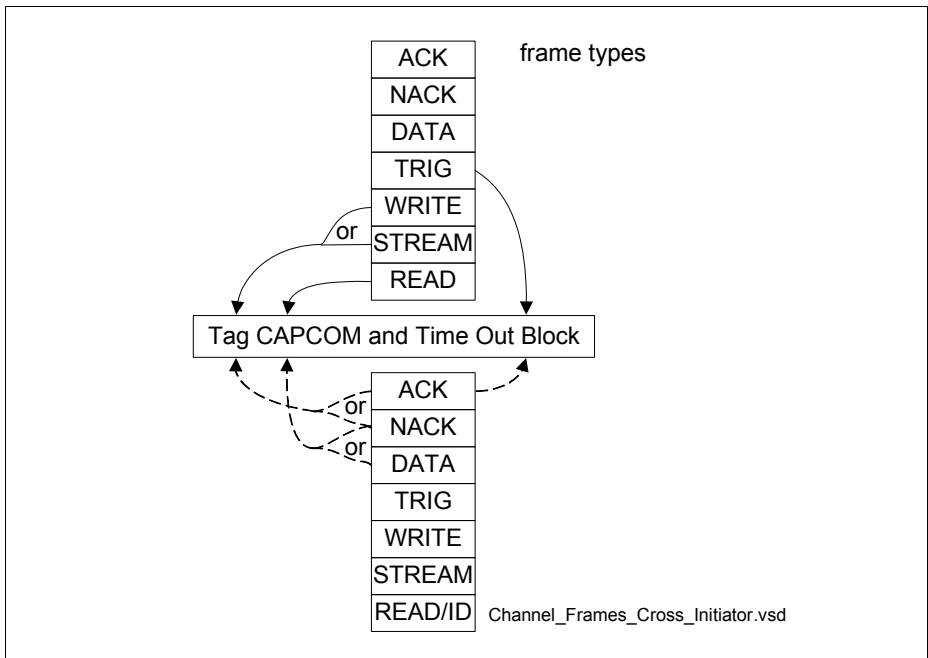


Figure 21-22 Channel 2 Cross Dependencies Between Stream and Command Frames and their Responses in an Initiator Role

21.4.5 Command Timeout

A timeout timer is started when a frame wins the arbitration and is delivered by the channel for transmission. The timeout timer is stopped if an appropriate response frame has arrived in time (without a CRC error). If an appropriate response has not arrived, and the timeout timer reaches zero, the timer is stopped, a timeout error is triggered and the expect flag is cleared.

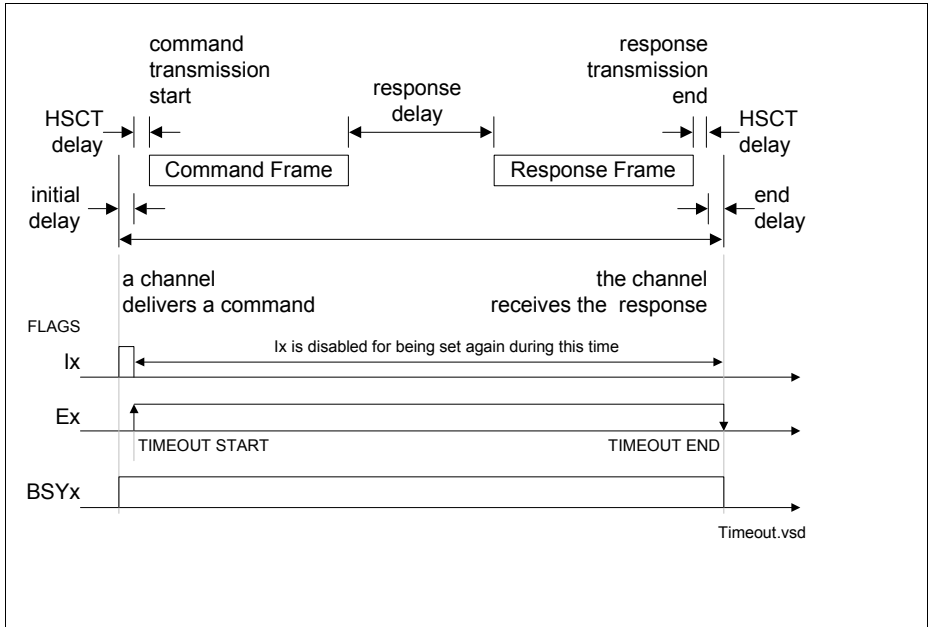


Figure 21-23 Command Timeout Measurement

21.4.5.1 Command Timeout Operation

The HSSL module contains one prescaler common to all channels, and one channel specific timer block per channel. The timer block generates the transaction tag for the channel by incrementing a three bit counter, captures the current transaction tag and compares it with the arrived response. In case of an error sequence CRC and timeout error, the timeout block generates a timeout signal. The timeout timer is reloaded and starts down-counting when the command transmission process starts.

The current timeout timer current value is indicated in the bit field **ICON.TOCV**, and the timeout reload value is configured in the bit field **ICON.TOREL**.

High Speed Serial Link (HSSL)

The currently expected transaction tag is indicated in the bit field **ICON.CETT**, the captured value of the latest erroneous transaction tag is indicated in the bit field **ICON.LETT**.

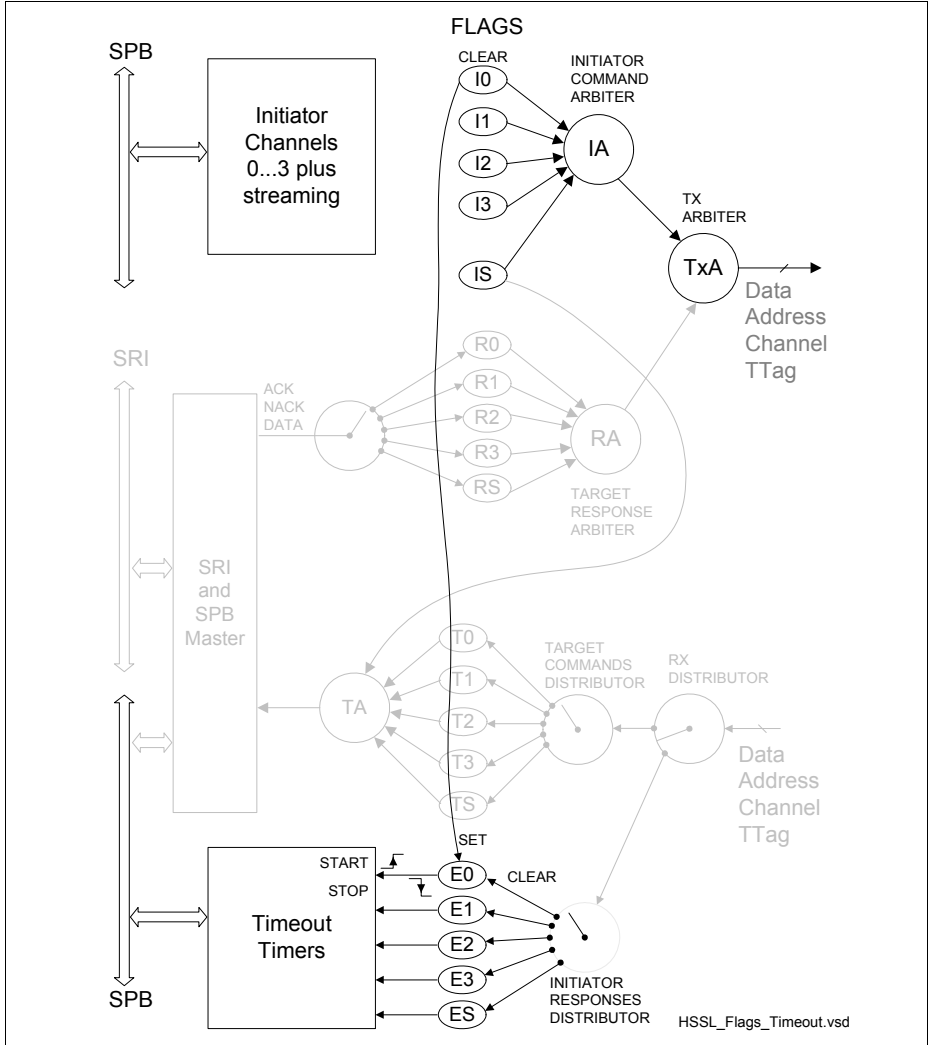


Figure 21-24 Request Flags Overview and a Communication Cycle Example

High Speed Serial Link (HSSL)

From the point of view of the flags, the timeout starts with the rising edge of the **QFLAGS**.Ex flag and ends with its falling edge, see **Figure 21-22**. The Ex flag is set by the falling edge of the lx flag, which is reset when its command request wins the transmit arbitration, see **Figure 21-24**.

21.4.6 Stream Timeout

A free timeout timer block is started when a frame wins an arbitration round and is delivered by the channel for transmission. The timeout timer block is reset to zero and stopped if an appropriate response frame has arrived in time (without a CRC error). If an appropriate response has not arrived, a timeout error is triggered and the timeout timer block is reset.

The stream initiator keeps track of the expected acknowledges using a virtual expect fifo having a virtual filling level EXFL. Each new stream frame transmission increment the EXFL, receiving a correct acknowledge decreases the EXFL. As long as the expect fifo is full (filling level 2) no new transmit request can be issued. In order a new stream frame request to be issued, which means the flag IS to be set by the module, three conditions must be met:

- $EXFL < 2$ - the expect level must be below two,
- $TXFL > 0$ - the TXFIFO must be not-empty and
- $ISB = 1$ - the bit ISB bit must have been set by the software.

High Speed Serial Link (HSSL)

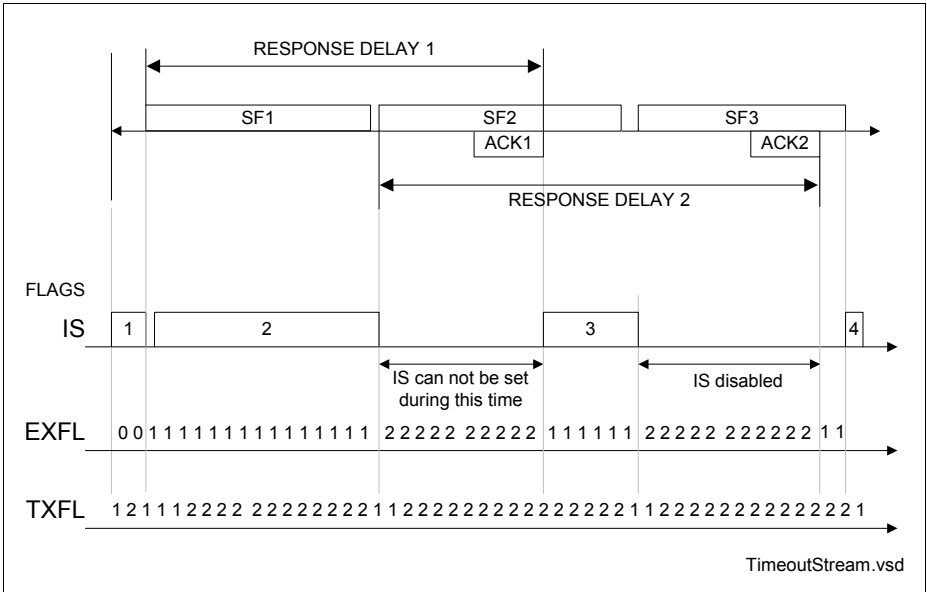


Figure 21-25 Stream Timeout Measurement

21.4.6.1 Stream Timeout Operation

The stream channel monitors the arrivals of the stream frame responses by using two timeout timer blocks. A single timer block is shown in **Figure 21-26**.

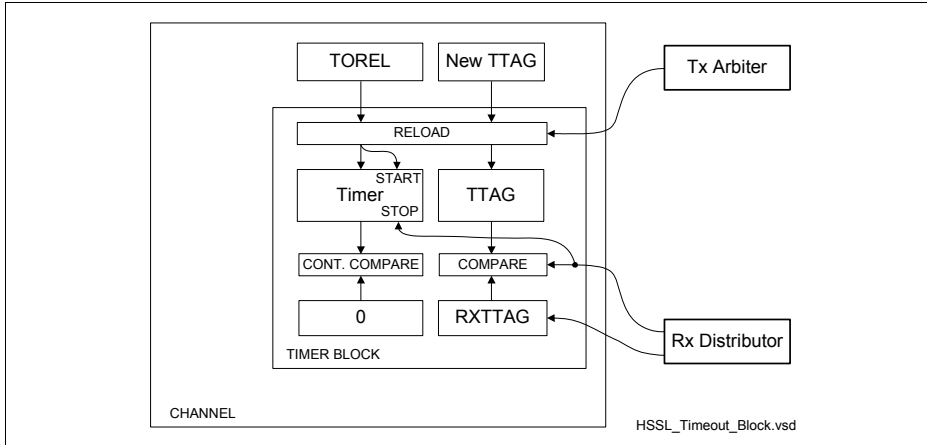


Figure 21-26 Single Timeout Block

The new initiator TTAG value which will be used for the next frame is visible in the bit field **ICONx.ITTAG**. The **ICONx** register also contains the TOREL and the TOCV bit fields.

High Speed Serial Link (HSSL)

The streaming functionality uses two timers sharing the same reload value, as defined by the **ICON2.TOREL**. The timeout timer blocks simulate the behavior of a FIFO (named Expect FIFO), by having additionally a “write” and “read” pointer:

- the “write” or transmit pointer points to the timer block where the next command frame will trigger the writing of the timer reload value and the transaction tag,
- the “read” or receive pointer points to the block which performs the transaction tag comparison with the currently arrived stream acknowledge TTAG.
- The EXFL or virtual filling level indicating how many timers are active 0, 1 or 2

The EXFL is visible in the register **SFSFLAGS**.

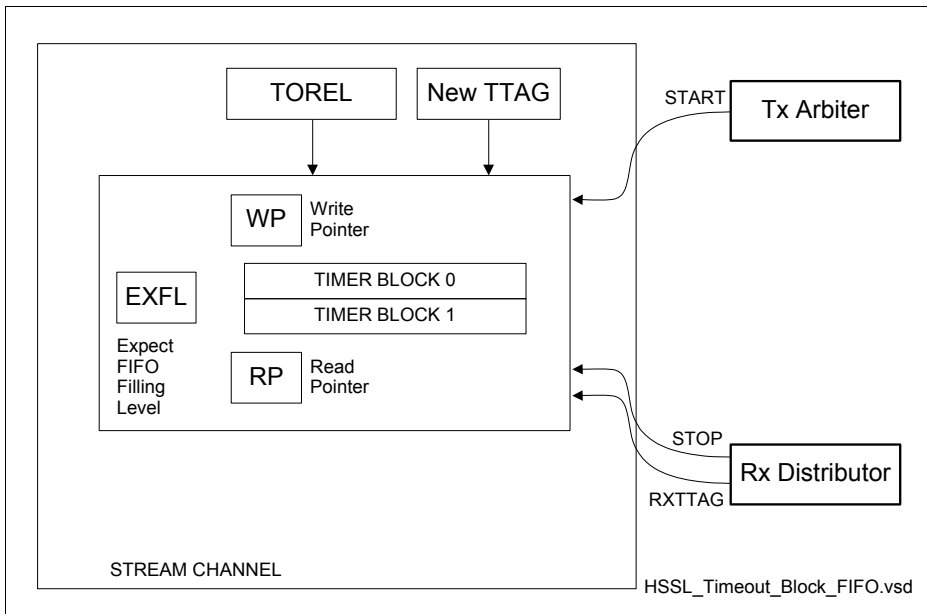


Figure 21-27 Stream Timeout Block

21.4.7 Data FIFOs of the Streaming Channel 2

Each stream frame delivers 256 bit data. This data is transferred by the SRI/SPB bus master in a memory by using BTR4 bursts. The start address and the end address is aligned at 256-bit block border, see **ISSAx (x=0-1)**, **ISFC**, **TSSAx (x=0-1)**, **TSFC**.

The channel expects one full streaming frame of 256 bit to be delivered before triggering either the SRI/SPB master or the HSCT module for fetching the data. The streaming is full duplex capable, which means one FIFO for each direction is available, and also Prioritisation for the requests to the SRI/SPB master.

Emptying the RXFIFO has higher priority than filling the TXFIFO.

A service request to the SRI/SPB master is generated when the complete payload of a streaming frame is available in a FIFO.

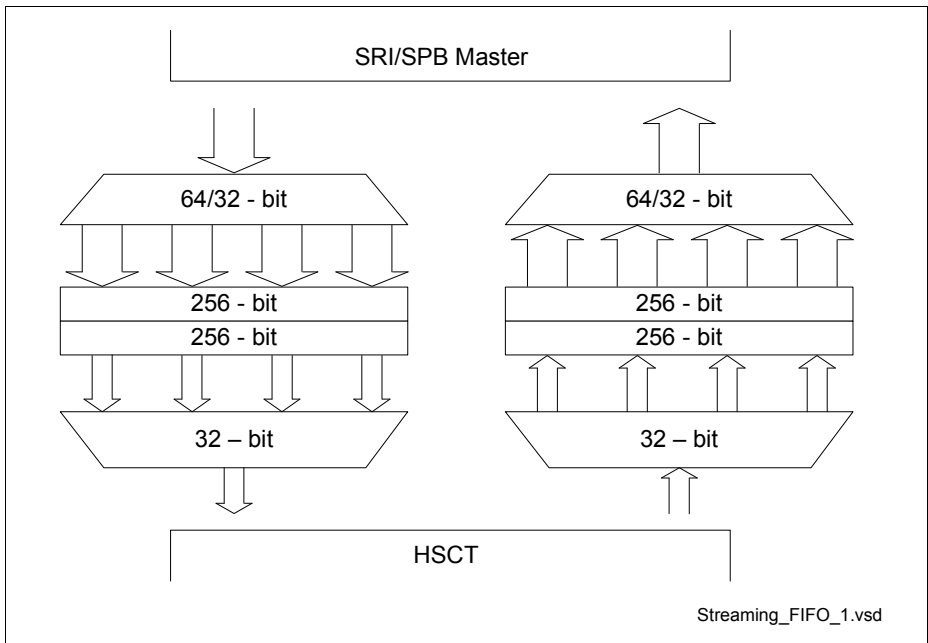


Figure 21-28 Streaming FIFO

21.5 Modes of Operation

The HSSL module has the following modes:

- Disabled mode (software controlled via DISR bit)
- Initialize mode (software via INI bit, sleep & suspend signals)
- Run mode (INI bit, sleep and suspend signals)
- OCDS soft suspend mode (OCDS suspend signal)
- Sleep mode (sleep signal)

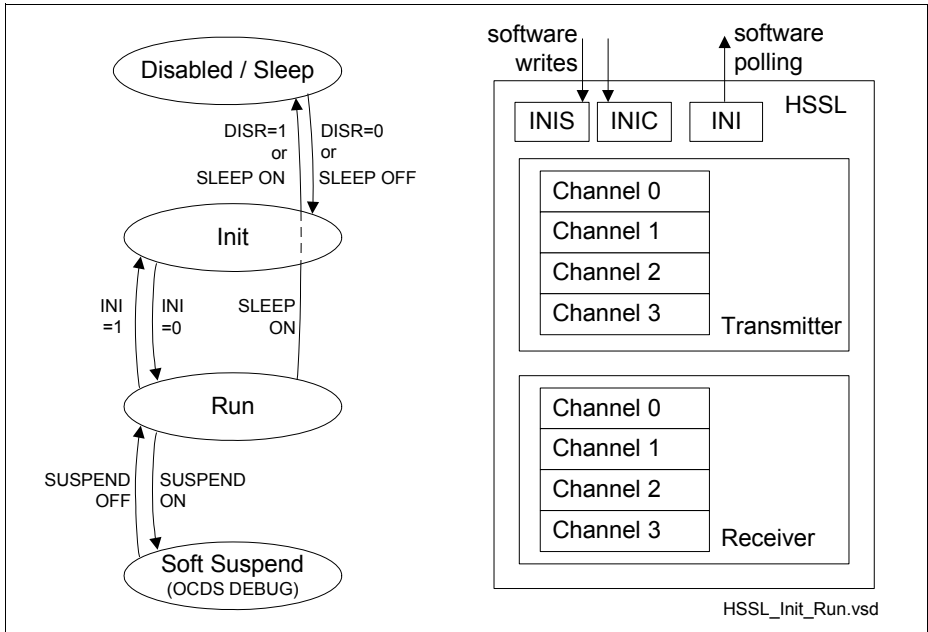


Figure 21-29 Modes of operation

After Init state has been requested by using the **MFLAGSSET.INIS** bit, the HSSL module immediately stops starting new SRI/SPB transactions and new command, response and stream frame transmissions by resetting all request bits (I, T and R) in the **QFLAGS** register and disabling these bits for being set again. The already started transactions/transmissions will be finished. Afterwards, it waits for all pending response frames to arrive or the corresponding timeout events to be raised. This condition is achieved when all expect bits in the **QFLAGS** register has been cleared by the hardware, and there are no ongoing transmission and SRI/SPB master activities. Subsequently the module enters the Init state and the **MFLAGS.INI** flag is set. The CTS signal is deactivated in order to inform the peer module about the Init state.

High Speed Serial Link (HSSL)

During the transition from run to sleep state, triggered by the sleep signal, the **MFLAGS.INI** bit is automatically set by the hardware (behaves as INI bit was set by software). When the sleep period is over and the sleep signal is deactivated, the module goes into the Init state and must be set to Run state by software, by writing **MFLAGSC.L.INIC=1**.

At the state transition of Init -> Run the timeout timers and all the flags are reset. The RUN state starts with all channels inactive.

At the state transition SoftSuspend -> Run the timers are not reset but continue to run and the flags states are preserved, because the module continues operating from the point at which it was suspended.

When leaving the Run state towards Sleep the CTS signal is deactivated and the transmission of the pending commands is stopped.

When a soft suspend is requested, the module stops the transmission of the pending command and stream frames, waits until all expect flags are cleared, then deactivates the CTS signal and at the end acknowledges the soft suspend request and sets the bit **OCS.SUSSTA**. Incoming target frames are served until the CTS signal stops the other side from sending new command and stream frames.

Hard Suspend request causes immediate switching off of the module clocks. After ending the Hard Suspend state both HSSL and HSCT modules must be reset by the application software and communication restarted from reset state.

Note: Reading and writing of registers is possible but will enable the kernel clock f_{CLC} for a few cycles. Attention: register accesses with clocking in Hard Suspend Mode can have unintended side effects like signals becoming and staying active. This can affect also other modules, so a HSSL kernel reset might not be sufficient to bring the system into a defined state.

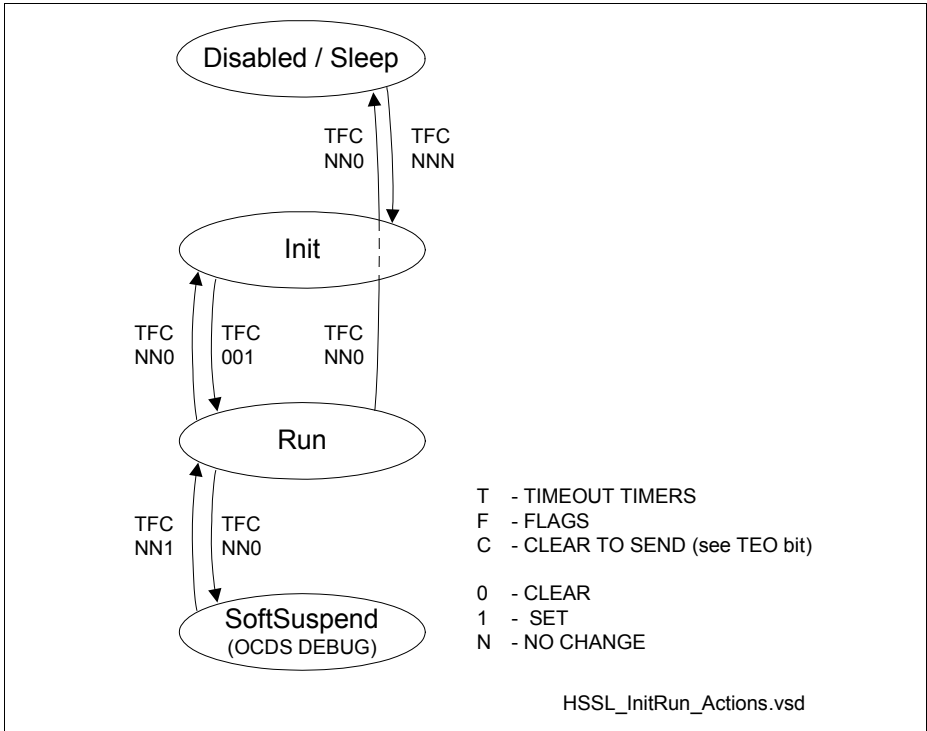


Figure 21-30 Actions on State Transitions

21.6 Interrupts

The HSSL module generates four types of interrupts per channel and one channel unspecific module interrupt.

Command Channel Interrupts

Each HSSL command channel generates four interrupts:

- Command OK interrupt COK
- Read Data interrupt RDI
- Error interrupt ERR
- Trigger interrupt, triggered by a trigger command frame TRG

An arrival of an error free frame response (ACK frame) triggers a COK interrupt, otherwise an ERR interrupt is triggered. An arrival of a read response frame triggers at the initiator side, additionally to the COK interrupt, an RDI interrupt.

An arrival of a trigger frame at the target side triggers a TRG interrupt there.

ERR interrupt is disjunct to COK and RDI interrupts, meaning that for one frame, either ERR or COK and optionally RDI can occur. After an ERR interrupt, normal transmission must be resumed by software, because an optional DMA would remain not retriggered and would wait for COK indefinitely.

Stream Channel Interrupts

The HSSL stream transmit sub-channel generates three interrupts:

- TBE - Transmit Block End interrupt, shared with the COK interrupt of the command mode of the channel 2
- error interrupt ERR, shared with the ERR interrupt of the command mode of the channel 2
- RBE - Receive Block End interrupt, shared with the TRG interrupt of the command mode of the channel 2.

Exception Interrupt EXI

If the receive stage of the HSSL module detects a CRC error or any inconsistency in the received data the global EXI Interrupt is activated, which is not channel specific.

Note: The HSCT module uses channels A to D for the HSSL channels 0 to 3. So, HSCT channel 0100_B corresponds to 000_B (binary code) or 100_B (special code) in the HSSL header.

The EXI interrupt is also raised on an edge of the TEI signal (HSCT CTS output).

In total, the HSSL module provides $4 \cdot 4 + 1 = 17$ interrupt lines.

21.7 Operating a Command Channel

All HSSL command channel provide identical functionality.

21.7.1 Initiating a Single Write Command

In order to initiate a write command, the software must

- configure the data width and the type of the request which will follow (read or write)
- provide the new data (most frequently, but not necessary)
- provide new address

Write to the address register requests the predefined type of the transfer

21.7.2 Initiating a Single Read Command

In order to initiate a read command, the software must

- configure the data width and the type of the request which will follow (read or write)
- skip providing of the data step, or provide dummy data
- provide new address

Write to the address register requests the predefined type of the transfer

21.7.3 Initiating a Single Trigger Command

In order to initiate a trigger command, the software can either

- request a trigger type of frame and provide dummy data and address
- directly request a trigger frame by writing **ICONx (x=0-3).TQ**

21.7.4 DMA Operated Command Queues

It is possible to use DMA to initiate lists of commands, by moving memory blocks containing the command lists to the HSSL module. In order to support such a way of operation, each channel provides three registers arranged in a particular way: first **IWD**, followed by **ICON** and **IRWA** at the end. For executing a list of write commands all three registers must be written for each command; for executing a list of read commands the first register **IWD** is optional (dummy write possible) but the last two **ICON** and **IRWA** are mandatory; for executing a trigger command only writing **ICON.TQ** is mandatory, but optionally even all three registers can be written: dummy write to **IWD**, write to **ICON.RWT** and write to **IRWA** to issue the request.

The answer data to a read command is available in the **IRD** register.

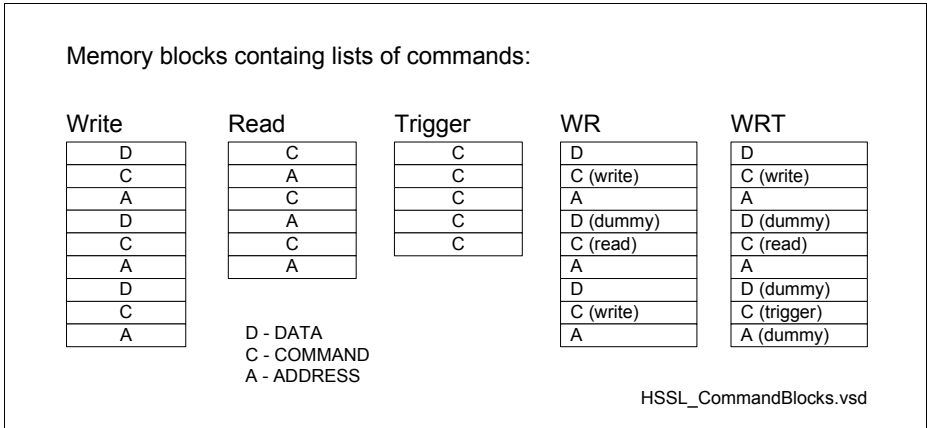


Figure 21-31 Command Queues

21.7.5 Receiver Error Handling

The HSSL receiver generates three error signals, two channel specific and one general:

- channel specific timeout and transaction tag errors and
- channel unspecific CRC error
- command frames, which pass the CRC check, containing an RFU command code are ignored
- Command frames, which pass the CRC check, containing invalid (greater than 3) channel number are ignored.
- If a command frame is received and the acknowledge of the previous command has not been sent yet, the new command frame is discarded.
- If a response frame comes, and the channel does not expect any more a response (the E flag is 0), than an incoming response is discarded without any error interrupts, additional to the timeout interrupt. This scenario can occur if a response is received after a timeout, and before the next command has been sent.
- If a channel in command mode receives a stream frame, the stream frame will be ignored.
- if a channel in streaming mode receives a command frame, the command frame will be ignored.

21.7.5.1 Timeout Error

Error triggered by the initiator.

21.7.5.2 Transaction Tag Error

Error triggered by the initiator.

21.7.6 Global Error

Global, channel not specific error is triggered if a CRC error, inconsistency between the HSCT and HSSL header, and SRI/SPB error occurs.

CRC error is triggered:

- by the target: when erroneous command frames has been received and
- by the initiator for erroneous response frames.

There is a dedicated interrupt named EXI which signals the CRC error. There is no indication if the CRC error was caused by a command or response frame. The CRC error caused by a response frame will be followed by a timeout interrupt. The received data is discarded; it is not available for the software in any register(s).

Additionally, the CRC error interrupt signals an inconsistency between the data length and channel number code delivered by the HSCT module and the information contained in the HSSL header. The differentiation between the two types of error can be done by reading the **MFLAGS**.CRCE, PIE1 and PIE2 flags. All three error types can occur and are detected independently of each other, so that more than one flag can be set at the same time.

An SRI error interrupt (EXI) is triggered if there was a transaction ID error, ECC error or SRI error acknowledge on the SRI bus. The flag **MFLAGS**.SRIE indicates this type of error. In case of an SRI read data ECC error, this event is additionally signalled to the SMU module.

An SPB error also triggers the global error interrupt (EXI) and sets the **MFLAGS**.SRIE flag.

21.8 Memory Block Transfer Modes of the Stream Channel

HSSL streaming channel is capable of transmitting one stream and receiving one stream in parallel. Streaming operates either in single block mode or continuous mode.

In single block mode, after triggering/enabling the streaming by using the **MFLAGS**.ISB/TSE bit, the SRI/SPB master of the HSSL module transmits/receives the preconfigured memory block, generates an interrupt signal and waits for the next trigger.

In continuous mode, after triggering/enabling the streaming by using the **MFLAGS**.ISB/TSE bit, the SRI/SPB master of the HSSL module transmits/receives two memory blocks of the same size in a round robin fashion indefinitely, or after being stopped by the software.

The streaming mode of the initiator/transmitter is configured by using the **CFG**.SMT bit. The streaming mode of the target/receiver is configured by using the **CFG**.SMR bit.

High Speed Serial Link (HSSL)

Triggering/enabling a block or continuous stream is done by using the Stream Block Request bit **MFLAGS.ISB/TSE** bit. In single block mode, the hardware resets this bit after performing a block transfer. In continuous mode, the transfer is ongoing continuously (**MFLAGS.ISB/TSE** remains set) and a stop of the transfer must be requested by writing one to **MFLAGSCL.ISBC/TSEC** bit.

On the initiator side, after completing the transfer of the current memory block that is currently read from the memory, the transmitter will be stopped and the **MFLAGS.ISB** will be cleared.

On the target side, after completing the transfer of the current memory block that is currently written to the memory, the receiver will be stopped and the **MFLAGS.TSE** will be cleared.

SRI/SPB error on the target triggers an error interrupt and the hardware stops. The TSE bit (and the RXFIFO) will be cleared by the hardware and the incoming frames are ignored.

Any error on the initiator side (like SRI/SPB error, target error, timeout, transaction tag error) triggers an error interrupt and the hardware stops streaming. The bit ISB (and the TXFIFO) will be cleared by hardware. All the incoming frames will be ignored, including the acknowledge frames of the previous frames.

Error can also occur at the beginning of a new memory block, while the last frames of the previous memory block are still in the TXFIFO. These frames are also lost, and the previous memory block remains incompletely received.

After an error while streaming in one direction the software must restart the streaming in that direction at both sides. The streaming will restart with a new block.

An error while streaming in most cases leaves the target waiting in the middle of a block. The target receiver can be stopped immediately (and afterwards restarted) by clearing the **MFLAGS.TSE** only when **TSFC.CURCOUNT** equals **TSFC.RELCOUNT**. Normally, **CURCOUNT** equals **RELCOUNT** before the start of a transfer of a block or between block transfers. Otherwise, while waiting in the middle of a block transfer, **RELCOUNT** can be made equal to **CURCOUNT** by writing the **CURCOUNT** value to **RELCOUNT**. This can be done either locally, by the target itself, or remotely, by the initiator using HSSL command frames.

The two initiator start addresses are configured in **ISSAx (x=0-1)**. The single block size is configured in the **ISFC** register. The current address being transferred is indicated in the read only (rh) **ISCA** register. The bit **MFLAGS.IMB** selects the corresponding **ISSAx (x=0-1)** register. This bit is not modified by the hardware in single block mode, and is toggled by the hardware in the continuous mode. At the start of the transfer, the selected start address **ISSAx (x=0-1)** is copied to the **ISCA** register.

The two target start addresses are configured in **TSSAx (x=0-1)**. The single block size is configured in the **TSFC** register. The current address being transferred is indicated in the read only (rh) **TSCA** register. The bit **MFLAGS.TMB** selects the corresponding

High Speed Serial Link (HSSL)

TSSAx (x=0-1) register. This bit is not modified by the hardware in single block mode, and is toggled by the hardware in the continuous mode. At the start of the transfer, the selected start address **TSSAx (x=0-1)** is copied to the **TSCA** register.

The HSSL stream transmitter generates one dedicated interrupt and several error events:

- transmit block end interrupt, generated after the acknowledge of the last frame has been received.
- error events that can trigger a channel error interrupt (target, timeout, transaction tag, unexpected error)
- a global SRI/SPB error interrupt (check if the ISB bit is cleared by hardware)

The HSSL stream receiver generates one dedicated interrupt and several error events:

- receive block end interrupt event, generated after the last stream frame in a block has been written to the memory and the SRI/SPB master has received a confirmation (an acknowledge, but not an error).
- error event (target error that triggers a NACK frame)
- a SRI/SPB global error interrupt (check if the TSE bit is cleared by hardware)

As long as the ISB bit is low, the TXFIFO and the expect FIFO are cleared and kept in the empty state.

As long as the TSE bit is low, the RXFIFO is cleared and kept in the empty state.

High Speed Serial Link (HSSL)

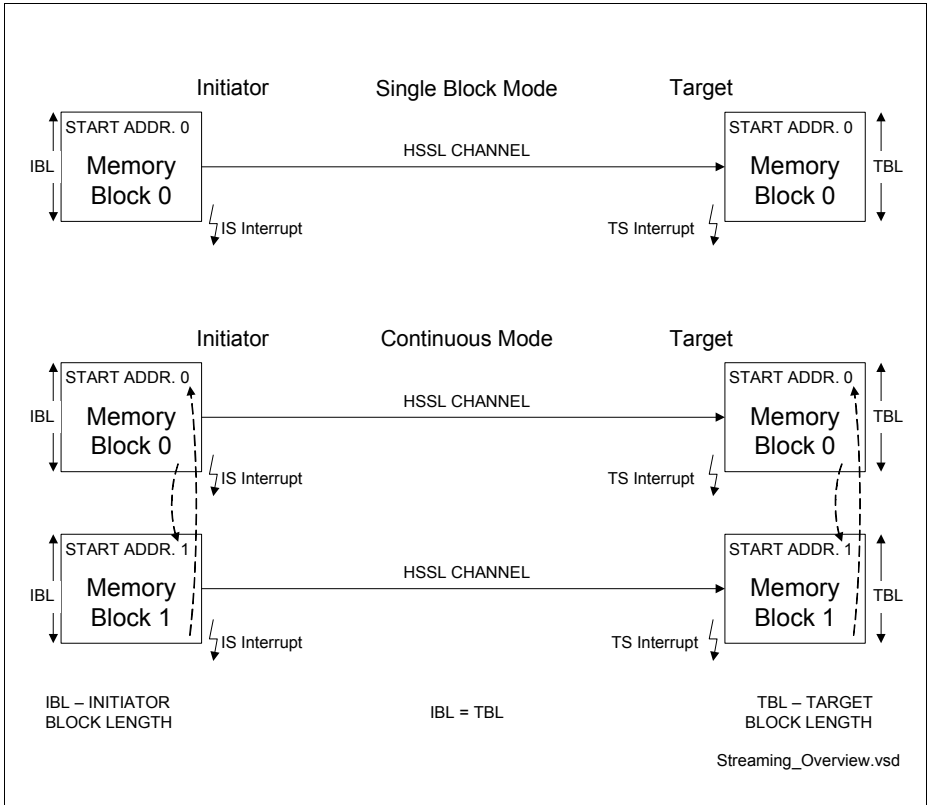


Figure 21-32 Stream Modes of Operation

21.9 HSSL Reset

Both HSCT and the HSSL modules must be reset together.

21.10 OCDS SRI / SPB Master Suspend

The SRI /SPB master access on the bus can be suspended by using the OCDS suspend request. When the master reaches idle state the suspend acknowledge signal is activated.

High Speed Serial Link (HSSL)

21.11 OCDS Trigger Sets

In order to support the debugging activities, the HSSL module provides a set of internal signals to the on-chip debug system OCDS. An overview of this feature is shown in **Figure 21-33**. Its configuration is done by using the bits **OC.S.TGS** and **TGB**.

An edge on any module internal signal belonging to the selected set going out on one of the two OTGB busses triggers an action of the OCDS system.

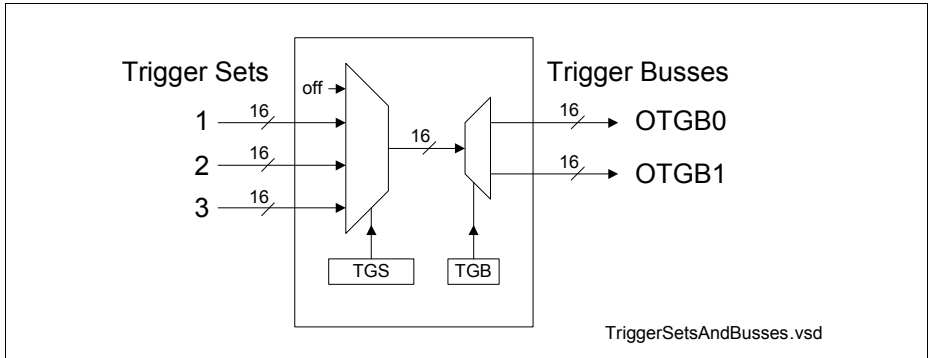


Figure 21-33 Overview of the Trigger Sets and Busses

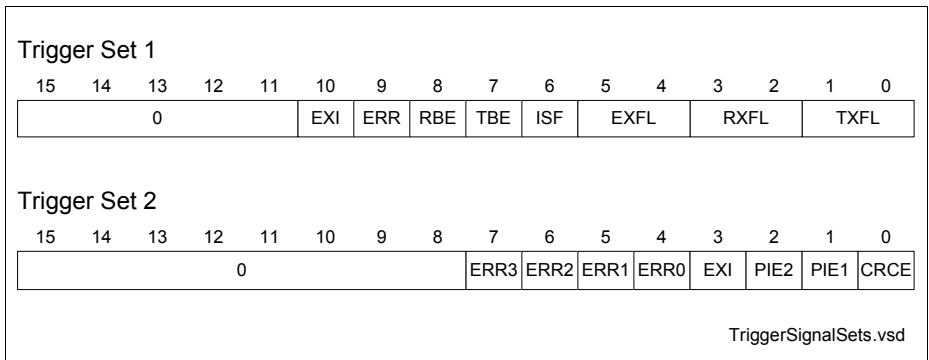


Figure 21-34 Overview of the Trigger Signal Sets

Table 21-4 HSSL Trigger Sets

Name	Description
TS16_STR	Streaming Channel Trigger Set (Trigger Set 1)
TS16_ERR	Error Trigger Set (Trigger Set 2)

High Speed Serial Link (HSSL)

The trigger sets 1 and 2 are used, the set 3 is not used (padded with 0).

Set 1, streaming channel debugging:

- 2 bits: TXFIFO filling level **SFSFLAGS.TXFL**
- 2 bits: RXFIFO filling level **SFSFLAGS.RXFL**
- 2 bits: Expect FIFO filling level **SFSFLAGS.EXFL**
- 1 bit: Initiator Stream Frame Request **SFSFLAGS.ISF**
- 4 bits: streaming channel interrupt signals TBE, RBE, ERR, EXI, see **Interrupts**.
- others: not used

Set 2, timeout errors caused by unspecific errors debugging:

- 1 bit: CRC error **MFLAGS.CRCE**
- 1 bit: Phy Inconsistency error, inconsistent channel number code, **MFLAGS.PIE1**
- 1 bit: Phy Inconsistency error, inconsistent data length, **MFLAGS.PIE2**
- 5 bits: error interrupt signals 1 x EXI and 4 x ERR, see **Interrupts**.
- others: not used

Note: The signal lists from above are mapped to the trigger sets in the following way: first item from the list to the trigger signal 0,... to 15.

21.12 Access Protection

The HSSL module provides memory access protection in form of four memory access windows. Each window can be located anywhere within the address space, having an arbitrary size with the granularity of 256 bytes.

Each window defines a memory range where an access is allowed. All four windows create a sort of an access filter that protects the memory from external writing or reading. A window can be a read only, write only or read and write window.

Each window is defined with:

- window start address, see register **AWSTART_x (x=0-3)**
- window end address, see register **AWEND_x (x=0-3)** and
- access rule: read only, write only or read and write, see register **AR**

If two access windows overlap, one read only and one write only, the common address range is read and write accessible. If no access window overlaps with r, w or rw window, the r, w or rw window wins. No access window means window disabled.

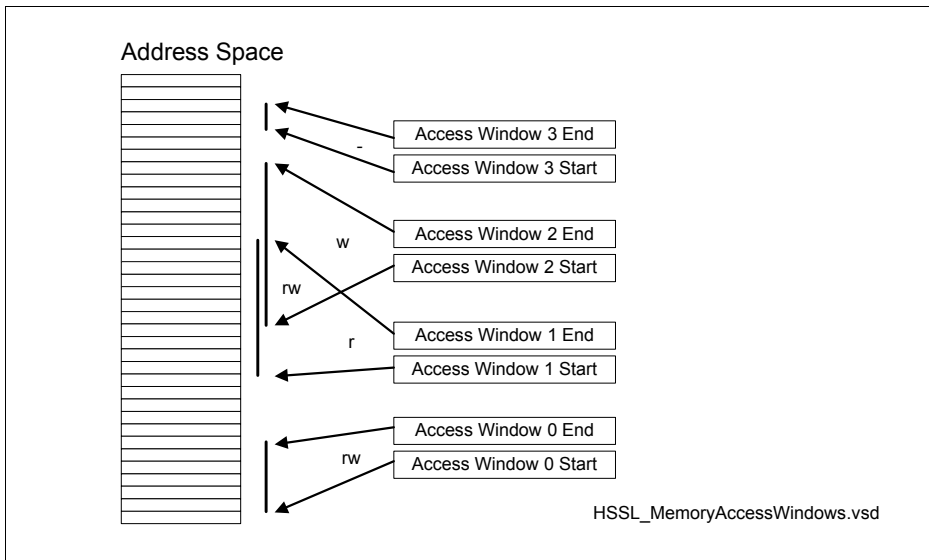


Figure 21-35 HSSL Access Windows

Normally, it is expected that the End Address is higher then or equal to the Start Address. Otherwise, the window is invalid and no commands can pass through the up-side-down range.

The **AWSTART_x (x=0-3)** registers define the first address of the first 256-byte block of an address window. The **AWEND_x (x=0-3)** registers define the first address of the last

High Speed Serial Link (HSSL)

256-byte block of an address window. The following diagram shows the details of the address window definition.

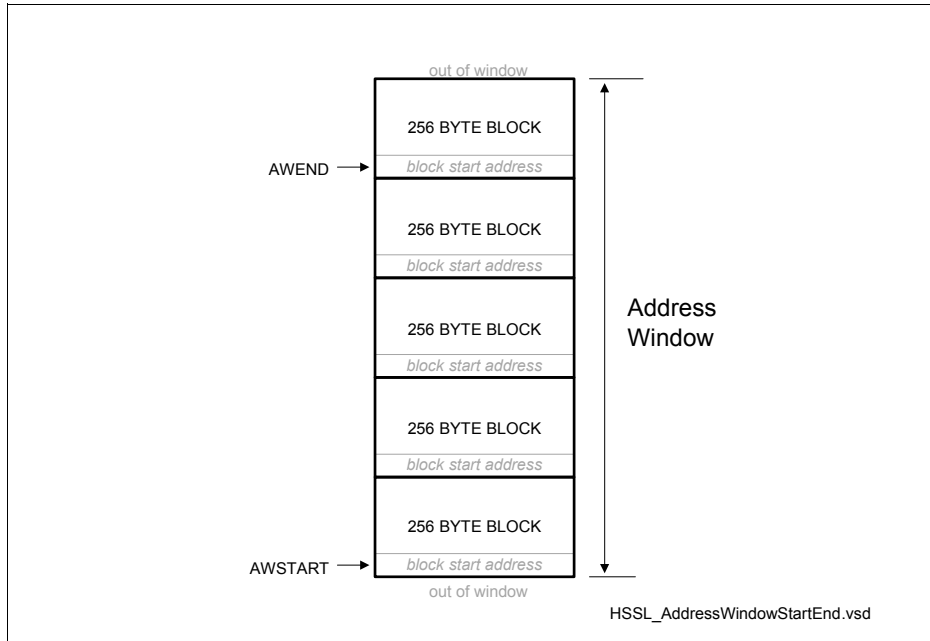


Figure 21-36 Access Windows Definition

These command frames are subject to filtering:

- Register Read
- Register Write and
- Stream Frame

These command always pass through:

- Read Answer
- ACK
- NACK
- Trigger
- ID Request

Command Frames Access Protection

If a HSSL target receives a read or write command frame with an address which passes through one of the windows, the command is executed and a response frame is sent back.

High Speed Serial Link (HSSL)

If a HSSL target receives a command frame with an address which does not pass through any windows, the command is not executed, a NACK response is sent back. The memory access violation flag **MFLAGS.MAV** is set, the access violating channel is captured in the bit field **AR.MAVCH**, and the EXI interrupt is triggered in the target.

Streaming Access Protection

In contrast to the read and write command frames, the stream frames do not contain an address information. There are two cases to distinguish, target side and initiator side.

At the target side, when the streaming starts, the initiator has already programmed the target addresses in the **TSSAx (x=0-1)** registers by using write frames, which pass through and carry the stream address in their data field. Therefore, the memory filtering uses the current access address, visible in the **TSCA** register, and if the current streaming address passes through the access protection filter, the read or write is executed, else a NACK frame is sent as a response.

If a HSSL target receives a stream frame and the current access address does not pass through any windows, the access is not executed, a NACK response is sent back, . The TSE bit will be cleared by hardware and the incoming frames will be ignored. The memory access violation flag **MFLAGS.MAV** is set, the access violating channel is captured in the bit field **AR.MAVCH**, and the EXI interrupt is triggered in the target.

At the initiator side, the **ISSAx (x=0-1)** registers, which are externally accessible, define the memory range to be transmitted. Therefore, the access filtering at the initiator side covers the streaming read accesses by using the current access address, visible in the **ISCA** register.

At the initiator side, access violation stops the streaming, and the ISB bit is cleared by hardware. The memory access violation flag **MFLAGS.MAV** is set, the access violating channel is captured in the bit field **AR.MAVCH**, and the EXI interrupt is triggered.

Public and Private Registers

The HSSL module contains two types of registers:

- public registers, which can be both read and written by the HSSL module itself
- private registers, which can be only read by the HSSL module itself

This behavior of the HSSL module registers is implemented by using the User Mode (U) and Supervisor Mode (SV) of writing registers. The private registers are writable in SV mode, the public registers are writable in U and SV mode.

The memory access protection registers **AWSTARTx (x=0-3)**, **AWENDx (x=0-3)** and **AR** including the **TIDADD** register are all writable only in Supervisor Mode, and are consequently private. The HSSL itself makes only User Mode accesses, so that it can not write these registers.

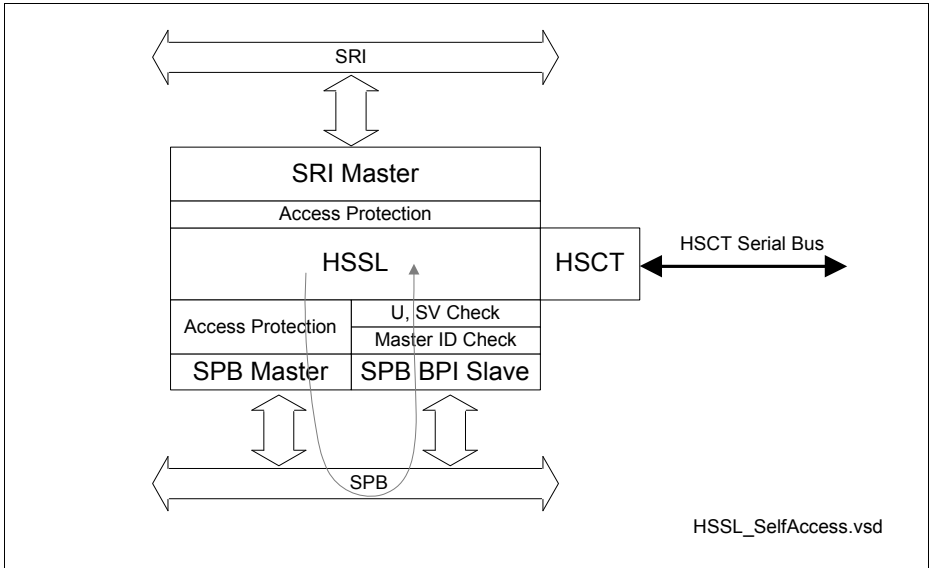


Figure 21-37 HSSL access to its own registers

21.13 Kernel Registers

This section describes the registers of the HSSL module.

Note: The HSSL module occupies 1KByte of address space, although only the first 256 bytes are used. Accessing a not used address location generates a bus error.

There are no registers in the HSSL with the destructive read property, where a read access would generate some action in the hardware.

Table 21-5 Registers Address Space

Module	Base Address	End Address	Note
HSSL	F0080000 _H	F00803FF _H	–

Table 21-6 Registers Overview

Register Short Name	Register Long Name	Offset Address	Write ¹⁾ Access	Reset Value
ID	Module Identification Register	08 _H	BE	00CB C0xx
CRC	Cyclic Redundancy Check	0C _H	SV, E, P	0000 0000
CFG	Configuration Register	10 _H	U, SV,P	0000 0000
QFLAGS	Request Flags Register	14 _H	U, SV,P	0000 0000
MFLAGS	Miscellaneous Flags Register	18 _H	U, SV,P	8000 0000
MFLAGSSET	Miscellaneous Flags Set Reg.	1C _H	U, SV,P	0000 0000
MFLAGSCL	Miscellaneous Flags Clear Reg.	20 _H	U, SV,P	0000 0000
MFLAGEN	Miscellaneous Flags Enable R.	24 _H	U, SV,P	0000 0000
SFSFLAGS	Stream FIFOs Status Flags Reg.	28 _H	U, SV,P	0000 0000
IWDx (x=0-3)	Initiator Write Data.	30 _H +10 _H *x	U, SV,P	0000 0000
ICONx (x=0-3)	Initiator Configuration Register	34 _H +10 _H *x	U, SV,P	0000 0000
IRWx (x=0-3)	Initiator Read Write Address	38 _H +10 _H *x	U, SV,P	0000 0000
IRDx (x=0-3)	Initiator Read Data	3C _H +10 _H *x	U, SV,P	0000 0000
TCDx (x=0-3)	Target Current Data	70 _H +8*x	U, SV,P	0000 0000
TCAx (x=0-3)	Target Current Address	74 _H +8*x	U, SV,P	0000 0000
TSTAT	Target Status Register	90 _H	U, SV,P	0000 0000
TIDADD	Target ID Address Register	94 _H	SV,P	0000 0000
ISSx (x=0-1)	Initiator Stream Start Address	A0 _H +4*x	U, SV,P	0000 0000
ISCA	Initiator Stream Current Address	A8 _H	U, SV,P	0000 0000

High Speed Serial Link (HSSL)
Table 21-6 Registers Overview

Register Short Name	Register Long Name	Offset Address	Write ¹⁾ Access	Reset Value
ISFC	Initiator Stream Frame Count	AC _H	U, SV,P	0000 0000
TSSAx (x=0-1)	Target Stream Start Address	B0 _H +4*x	U, SV,P	0000 0000
TSCA	Target Stream Current Address	B8 _H	U, SV,P	0000 0000
TSFC	Target Stream Frame Count	BC _H	U, SV,P	0000 0000
AWSTARTx (x=0-3)	Access Window Start Register	C0 _H +8*x	SV, P ²⁾	0000 0000
AWENDx (x=0-3)	Access Window End Register	C4 _H +8*x	SV, P ²⁾	0000 0000
AR	Access Rules Register	E0 _H	SV, P ²⁾	0000 0000
BPI Registers				
CLC	Clock Control Register	00 _H	SV, E, P	0000 0003
OCS	OCDS Control and Status Reg.	E8 _H	SV, P	0000 0000
KRSTCLR	Reset Status Clear Register	EC _H	SV, E, P	0000 0000
KRST1	Reset Control Register 1	F0 _H	SV, E, P	0000 0000
KRST0	Reset Control Register 0	F4 _H	SV, E, P	0000 0000
ACCEN1	Access Enable Register 1	F8 _H	SV, SE	0000 0000
ACCEN0	Access Enable Register 0	FC _H	SV, SE	FFFF FFFF

1) All registers have read access mode U, SV, except the access protection registers AWSTART, AWEND and AR, which have read access mode SV.

Read access from reserved addresses delivers bus error (BE).

All registers belong to reset class 3 except OCS, which belongs to reset class 1.

2) Read access mode: SV only.

This register delivers 0 to a read access in User Mode, but BE is not triggered.

List of Access Protection Abbreviations

U - User Mode

SV - Supervisor Mode

BE - Bus Error

nBE - no Bus Error

P - Access Protection, as defined by the ACCEN Register

E - ENDINIT

SE - Safety ENDINIT

21.13.1 Global Registers

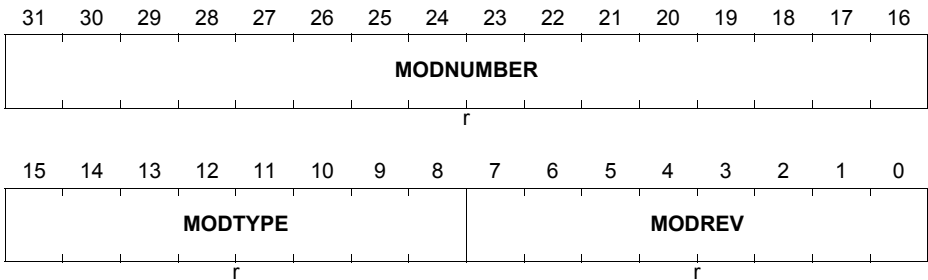
ID

The Module Identification Register ID contains read-only information about the module version.

(>> [Table 21-6](#) register overview)

ID

Module Identification Register (08_H) **Reset Value: 00CB C0XX_H**



Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number MODREV defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MODTYPE	[15:8]	r	Module Type This bit field is C0 _H . It defines a 32-bit module.
MOD NUMBER	[31:16]	r	Module Number Value This bit field together with MODTYPE uniquely identifies a module.

High Speed Serial Link (HSSL)

CRC

(>> [Table 21-6](#) register overview)

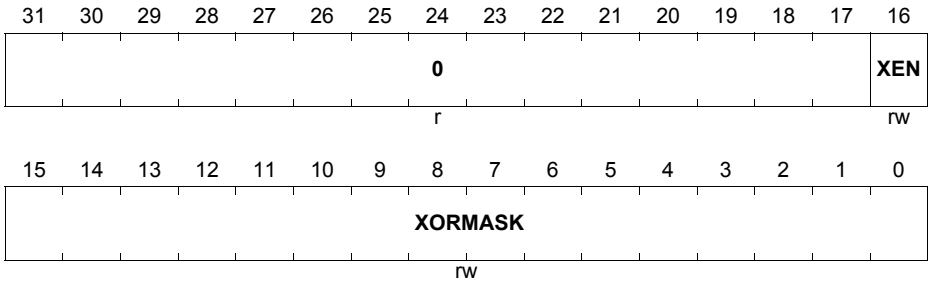
This register only influences the generation of the CRC in the transmit direction.

CRC

CRC Control Register

(0C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
XORMASK	[15:0]	rw	Value to be XORed with the Calculated CRC Used for error injection / test purposes.
XEN	16	rw	Enable the Error Injection via XORMASK 0 _B disabled 1 _B enabled
0	[31:17]	r	reserved

High Speed Serial Link (HSSL)

CFG

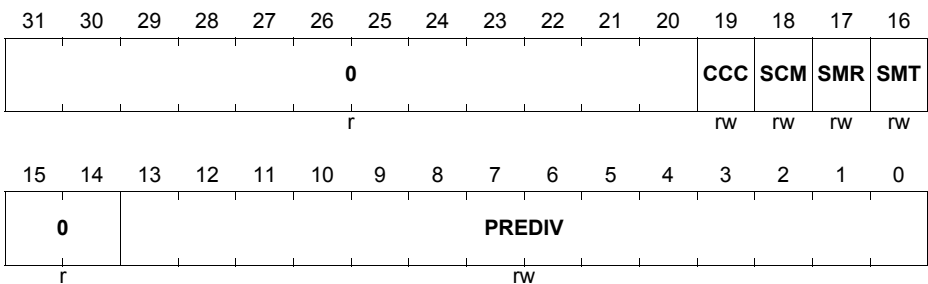
The CFG register is used to configure the HSSL module. It shall only be written in the Initialize Mode.

(>> **Table 21-6** register overview)

Note: The duration of a timeout interval may vary for an amount of one predivider clock period. Therefore the predivider should be set as low as possible, and the timeout length as many predivider periods as possible.

CFG

Configuration Register (10_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
PREDIV	[13:0]	rw	Global Predivider Defines the down-scaled module clock to be used by all channel timeout timers. 0 _D 1 1 _D 2 2 _D 3 others , corresponding down-scale factor
SMT	16	rw	Streaming Mode Transmitter 0 _B Continuous 1 _B Single
SMR	17	rw	Streaming Mode Receiver 0 _B Continuous 1 _B Single

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
SCM	18	rw	Streaming Channel Mode Defines if the channel 2 is used in a streaming or command mode. 0 _B Command 1 _B Streaming
CCC	19	rw	Channel Code Control Defines the coding of the channel number in the HSSL header. 0 _B Binary 1 _B Special
0	[31:20], [15:14]	r	reserved

21.13.2 Channel.Flags Registers

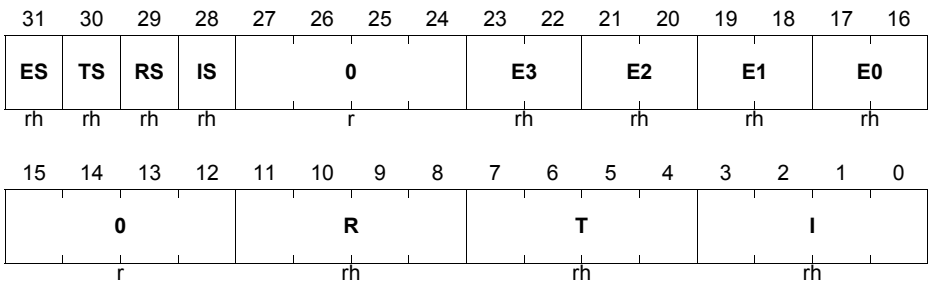
QFLAGS

This register contains flags indicating if an appropriate action request is pending or not. The action request can be pending in different stages of a communication cycle, see [Figure 21-16](#).

(>> [Table 21-6](#) register overview)

QFLAGS

Request Flags Register (14_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
I	[3:0]	rh	<p>Request Flags for Initiated Commands</p> <p>These flags are set by the corresponding channel when a WRTS command is initiated. The WRT commands are initiated via the SPB bus. The S(tream) commands are initiated by the module internally, by the TXFIFO, except for the first stream frame start, which is done via the SPB bus. See MFLAGS.ISB and ISF flags.</p> <p>0000_B No request pending 0001_B Channel 0 request pending 0010_B Channel 1 request pending 0100_B Channel 2 request pending 1000_B Channel 3 request pending others, corresponding combination of requests</p>

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
T	[7:4]	rh	<p>Request Flags for Commands Arrived at Target These flags are set by the hardware according to the header information of a frame arrived at the target without a CRC error. They are used by the arbiter of the SRI/SPB master and cleared when the appropriate command is executed.</p> <p>0000_B No request pending 0001_B Channel 0 request pending 0010_B Channel 1 request pending 0100_B Channel 2 request pending 1000_B Channel 3 request pending others, corresponding combination of requests</p>
R	[11:8]	rh	<p>Request Flags for Response Frames at the Target After a command has been executed by the SRI/SPB master, an appropriate flag is being set which indicates that an ACK/NACK/DATA is pending.</p> <p>0000_B No request pending 0001_B Channel 0 request pending 0010_B Channel 1 request pending 0100_B Channel 2 request pending 1000_B Channel 3 request pending others, corresponding combination of requests</p>
E0	[17:16]	rh	<p>Expect Flags for Activated Timeout Timer 0 An appropriate two bit flag is set by the hardware when a timeout timer for a channel is started. The hardware clears the appropriate flag when any response frame arrives at the initiator. In case of an unexpected response the flag UNEXPECTED is additionally set.</p> <p>00_B No request pending 01_B write request pending 10_B read request pending 11_B trigger request pending</p>
E1	[19:18]	rh	Expect Flags for Activated Timeout Timer 1
E2	[21:20]	rh	Expect Flags for Activated Timeout Timer 2
E3	[23:22]	rh	Expect Flags for Activated Timeout Timer 3

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
IS	28	rh	I Flag for Stream Frames See the “I” flag description above. 0 _B No request pending 1 _B Request pending
RS	29	rh	R Flag for Stream Frames See the “R” flag description above. 0 _B No request pending 1 _B Request pending
TS	30	rh	T Flag for Stream Frames See the “T” flag description above. 0 _B No request pending 1 _B Request pending
ES	31	rh	E Flag for Stream Frames See the “E” flag description above. 0 _B No request pending 1 _B Request pending
0	[27:24], [15:12]	r	reserved

MFLAGS

 (>> [Table 21-6](#) register overview)

 (>> [Memory Block Transfer Modes of the Stream Channel](#))

MFLAGS
Miscellaneous Flags Register
(18_H)
Reset Value: 8000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INI	TEO	TEI	TSE	0	0	CRC E	PIE2	PIE1	SRIE	MAV	ISB	IMB	TMB	0	
rh	rh	rh	rh	r	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNEXPECTED				TIMEOUT				TTE				NACK			
rh				rh				rh				rh			

Field	Bits	Type	Description
NACK	[3:0]	rh	Not Acknowledge Error - Target Error Indicates that a target error frame has been received.
TTE	[7:4]	rh	Transaction Tag Error Indicates for each channel if a CRC correct acknowledge frame with an unexpected transaction tag number has been received.
TIMEOUT	[11:8]	rh	Timeout Error Indicates for each channel if an timeout event has occurred.
UNEXPECTED	[15:12]	rh	Unexpected Type of Frame Error Indicates for each channel if an unexpected or inappropriate response is received. For example a NACK for a Trigger frame or DATA for WRITE frame.

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
TMB	18	rh	<p>Target Memory Block Selects the currently active memory block used by the target as a target for the streaming data, with its start address and frame counter. Switching the active block in the middle of a block transfer is not allowed.</p> <p>0_B Memory block 0 1_B Memory block 1</p>
IMB	19	rh	<p>Initiator Memory Block Selects the currently active memory block used by the initiator as a source for the streaming data, with its start address and frame counter. Switching the active block in the middle of a block transfer is not allowed.</p> <p>0_B Memory block 0 1_B Memory block 1</p>
ISB	20	rh	<p>Initiator Stream Block Request Indicates if stream block request is pending. Set by the software to start a stream block transfer by using the MFLAGSSET.ISBS bit; clear by the software possible (if needed) by using the MFLAGSCL.ISBC bit; cleared by hardware at the end of the current block transfer in single mode, but not in continuous mode.</p> <p>0_B No request or streaming ongoing 1_B Streaming ongoing</p>
MAV	21	rh	<p>Memory Access Violation Indicates a memory access violation.</p> <p>0_B No violation 1_B Violation</p>
SRIE	22	rh	<p>SRI/SPB Bus Access Error Indicates an error on the SRI bus - transaction ID, ECC error or error acknowledge. Indicates an error on the SPB bus, error acknowledge or timeout.</p> <p>0_B No error 1_B Error</p>

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
PIE1	23	rh	PHY Inconsistency Error 1 Indicates if HSCT to HSSL channel number code comparator has detected an inconsistency error. 0 _B No error 1 _B Error
PIE2	24	rh	PHY Inconsistency Error 2 Indicates if HSCT to HSSL data length comparator has detected an inconsistency error. 0 _B No error 1 _B Error
CRCE	25	rh	CRC Error Indicates if CRC checker has detected a CRC error. 0 _B No error 1 _B Error
TSE	28	rh	Target Stream Enable Used by the hardware to handle the single and continuous streaming. In single mode, cleared by hardware after the current block transfer ends. The module ignores afterwards the incoming steam frames. 0 _B Disabled 1 _B Enabled
TEI	29	rh	Transmit Enable Input Indicates the state of the TEI input signal of the HSSL module, which is driven by the CTS output signal of the HSCT module. Any edge on this signal triggers an EXI interrupt. This low level signal stops the transmission of both command and response frames.
TEO	30	rh	Transmit Enable Output Indicates the state of the TEO output signal of the HSSL module, which drives thee CTS input signal of the HSCT module. This bit is cleared by hardware at entering the INIT and Soft Suspend state.
INI	31	rh	Initialize Mode Indicates if the module is in the Initialize or Run mode. 0 _B Run mode 1 _B Initialize mode

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
0	[27:26], [17:16]	r	reserved

MFLAGSSET

 (>> [Table 21-6](#) register overview)

 (>> [Memory Block Transfer Modes of the Stream Channel](#))

Note: Read access to this register returns all zero.

MFLAGSSET
Miscellaneous Flags Set Register

 (1C_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INIS	TEOS	0	TSES	0	CRCES	PIES2	PIES1	SRIES	MAVS	ISBS	IMBS	TMBS	0		
w	w	r	w	r	w	w	w	w	w	w	w	w	w	r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNEXPECTEDS				TIMEOUTS				TTES				NACKS			
w				w				w				w			

Field	Bits	Type	Description
NACKS	[3:0]	w	NACK Flags Set Writing 1 sets the corresponding bit in the MFLAGS register and triggers the ERR interrupt for the corresponding channel, if enabled in the MFLAGSEN register. Writing 0 has no effect. 0 _B No action 1 _B Set
TTES	[7:4]	w	Transaction Tag Error Flags Set Writing 1 sets the corresponding bit in the MFLAGS register and triggers the ERR interrupt for the corresponding channel, if enabled in the MFLAGSEN register. Writing 0 has no effect. 0 _B No action 1 _B Set

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
TIMEOUTS	[11:8]	w	Timeout Error Flags Set Writing 1 sets the corresponding bit in the MFLAGS register and triggers the ERR interrupt for the corresponding channel, if enabled in the MFLAGSEN register. Writing 0 has no effect. 0 _B No action 1 _B Set
UNEXPECTEDS	[15:12]	w	Unexpected Error Flags Set Writing 1 sets the corresponding bit in the MFLAGS register and triggers the ERR interrupt for the corresponding channel, if enabled in the MFLAGSEN register. Writing 0 has no effect. 0 _B No action 1 _B Set
TMBS	18	w	Target Memory Block Flag Set Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Set
IMBS	19	w	Initiator Memory Block Flag Set Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Set
ISBS	20	w	Initiator Stream Block Request Set Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Set
MAVS	21	w	MAV Flag Set Writing 1 sets the corresponding bit in the MFLAGS register, and generates an EXI interrupt if enabled in the corresponding bit of the MFLAGSEN register. Writing 0 has no effect. 0 _B No action 1 _B Set

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
SRIES	22	w	SRI/SPB Bus Access Error Flag Set Writing 1 sets the corresponding bit in the MFLAGS register, and generates an EXI interrupt if enabled in the corresponding bit of the MFLAGEN register. Writing 0 has no effect 0 _B No action 1 _B Set
PIE1S	23	w	PIE1 Error Flag Set Writing 1 sets the corresponding bit in the MFLAGS register, and generates an EXI interrupt if enabled in the corresponding bit of the MFLAGEN register. Writing 0 has no effect 0 _B No action 1 _B Set
PIE2S	24	w	PIE2 Error Flag Set Writing 1 sets the corresponding bit in the MFLAGS register, and generates an EXI interrupt if enabled in the corresponding bit of the MFLAGEN register. Writing 0 has no effect. 0 _B No action 1 _B Set
CRCES	25	w	CRC Error Flag Set Writing 1 sets the corresponding bit in the MFLAGS register, and generates an EXI interrupt if enabled in the corresponding bit of the MFLAGEN register. Writing 0 has no effect. 0 _B No action 1 _B Set
TSES	28	w	Target Stream Enable Flag Set Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Set
TEOS	30	w	Transmit Enable Flag Set Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Set

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
INIS	31	w	Initialize Mode Flag Set Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Set
0	29, [27:26], [17:16]	r	reserved

High Speed Serial Link (HSSL)

MFLAGSCL

 (>> [Table 21-6](#) register overview)

 (>> [Memory Block Transfer Modes of the Stream Channel](#))

Note: Read access to this register returns all zero.

MFLAGSCL
Miscellaneous Flags Clear Register (20_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INIC	TEO C	0	TSE C	0	CRC EC	PIE2 C	PIE1 C	SRIE C	MAV C	ISBC	IMB C	TMB C	0		
w	w	r	w	r	w	w	w	w	w	w	w	w	w	r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNEXPECTEDC				TIMEOUTC				TTEC				NACKC			
w				w				w				w			

Field	Bits	Type	Description
NACKC	[3:0]	w	NACK Flags Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear
TTEC	[7:4]	w	Transaction Tag Error Flags Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear
UNEXPECTEDC	[15:12]	w	Unexpected Error Flags Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear
TIMEOUTC	[11:8]	w	Timeout Error Flags Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
TMBC	18	w	Target Memory Block Flag Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear
IMBC	19	w	Initiator Memory Block Flag Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear
ISBC	20	w	Initiator Stream Block Request Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear
MAVC	21	w	MAV Flag Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect 0 _B No action 1 _B Clear
SRIEC	22	w	SRI/SPB Bus Access Error Flag Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect 0 _B No action 1 _B Clear
PIE1C	23	w	PIE1 Error Flag Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect 0 _B No action 1 _B Clear
PIE2C	24	w	PIE2 Error Flag Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
CRCEC	25	w	CRC Error Flag Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear
TSEC	28	w	Target Stream Enable Flag Clear Writing 1 sets the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear
TEOC	30	w	Transmit Enable Flag Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear
INIC	31	w	Initialize Mode Flag Clear Writing 1 clears the corresponding bit in the MFLAGS register. Writing 0 has no effect. 0 _B No action 1 _B Clear
0	29, [27:26], [17:16]	r	reserved

High Speed Serial Link (HSSL)
MFLAGSEN

 (>> [Table 21-6](#) register overview)

MFLAGSEN
Flags Enable Register

 (24_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	TEI EN		0			CRC EEN	PIE2 EN	PIE1 EN	SRIE EN	MAV EN			0		
r	rw		r			rw	rw	rw	rw	rw			r		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNEXPECTEDEN				TIMEOUTEN				TTEEN				NACKEN			
rw				rw				rw				rw			

Field	Bits	Type	Description
NACKEN	[3:0]	rw	Not Acknowledge Error Enable Bits Used to enable the error interrupt associated to the corresponding bit in the FLAG register. 0 _B Disabled 1 _B Enabled
TTEEN	[7:4]	rw	Transaction Tag Error Enable Bits Used to enable the error interrupt associated to the corresponding bit in the FLAG register. 0 _B Disabled 1 _B Enabled
TIMEOUTEN	[11:8]	rw	Timeout Error Enable Bits Used to enable the error interrupt associated to the corresponding bit in the FLAG register. 0 _B Disabled 1 _B Enabled
UNEXPECTEDEN	[15:12]	rw	Unexpected Error Enable Bits Used to enable the error interrupt associated to the corresponding bit in the FLAG register. 0 _B Disabled 1 _B Enabled

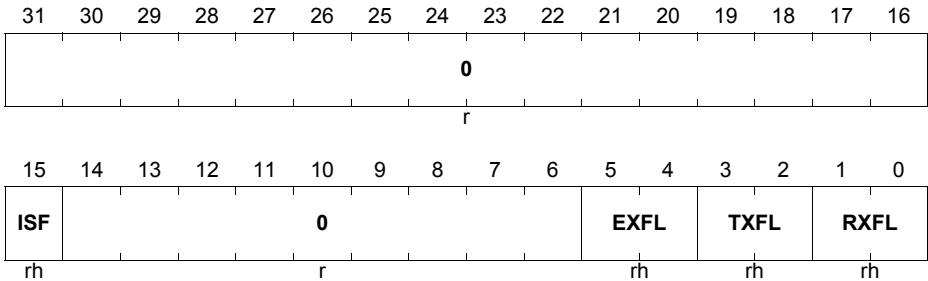
High Speed Serial Link (HSSL)

Field	Bits	Type	Description
MAVEN	21	rw	MAV Enable Bit Used to enable the interrupt associated to the corresponding bit in the FLAG register. 0 _B Disabled 1 _B Enabled
SRIEEN	22	rw	SRI/SPB Bus Access Error Enable Bit Used to enable the error interrupt associated to the corresponding bit in the FLAG register. 0 _B Disabled 1 _B Enabled
PIE1EN	23	rw	PIE1 Error Enable Bit Used to enable the error interrupt associated to the corresponding bit in the FLAG register. 0 _B Disabled 1 _B Enabled
PIE2EN	24	rw	PIE2 Error Enable Bit Used to enable the error interrupt associated to the corresponding bit in the FLAG register. 0 _B Disabled 1 _B Enabled
CRCEEN	25	rw	CRC Error Enable Bit Used to enable the error interrupt associated to the corresponding bit in the FLAG register. 0 _B Disabled 1 _B Enabled
TEIEN	29	rw	TEI Enable Bit Used to enable the interrupt associated to the corresponding bit in the FLAG register. 0 _B Disabled 1 _B Enabled
0	[31:30], [28:26], [20:16]	r	reserved

SFSFLAGS

 (>> [Table 21-6](#) register overview)

 (>> [Memory Block Transfer Modes of the Stream Channel](#))

SFSFLAGS
Stream FIFOs Status Flags Register (28_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RXFL	[1:0]	rh	Stream RxFIFO Filling Level Indicates the filling level of the FIFO with granularity of 32 bytes (one stream frame payload size). 00 _B 0 01 _B 1 10 _B 2 11 _B reserved (not possible)
TXFL	[3:2]	rh	Stream TxFIFO Filling Level Indicates the filling level of the FIFO with granularity of 32 bytes (one stream frame payload size). 00 _B 0 01 _B 1 10 _B 2 11 _B reserved (not possible)
EXFL	[5:4]	rh	Stream Expect FIFO Filling Level Indicates the filling level of the FIFO with granularity of 32 bytes (one stream frame payload size). 00 _B 0 01 _B 1 10 _B 2 11 _B reserved (not possible)

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
ISF	15	rh	Initiator Stream Frame Request Indicates if stream TXFIFO request is pending. Set and cleared by the TXFIFO. 0 _B TXFIFO fill request not pending 1 _B TXFIFO fill request pending
0	[31:16], [14:6]	r	reserved

21.13.3 Channel.Initiator Registers

This section describes the channel specific registers of the HSSL module, associated to both the initiator and the target functions. In order to support command queues, the addresses of the **IWD**, **ICON**, **IRWA**, and **IRD** registers are ordered in a sequence of IWD0, ICON0, IRWA0, IRD0, IWD1, ICON1, IRWA1, IRD1, IWD2, ICON2, IRWA2, IRD2, IWD3, ICON3, IRWA3, IRD3, see **DMA Operated Command Queues**. For the timeout handling description, see **Command Timeout Operation**.

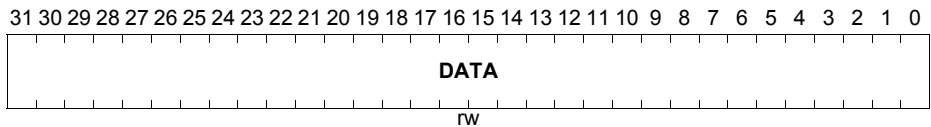
IWD

The IWD register contains the data part of a write command.

(>> **Table 21-6** register overview)

IWDx (x=0-3)

Initiator Write Data Register x ($30_H + x*10_H$) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
DATA	[31:0]	rw	Data Part of the Payload of a Write Frame For 8-bit and 16-bit write command frames, the whole frame payload width of 32-bit is automatically filled with copies of the lower 8-bits or 16-bits of the register.

High Speed Serial Link (HSSL)
ICON

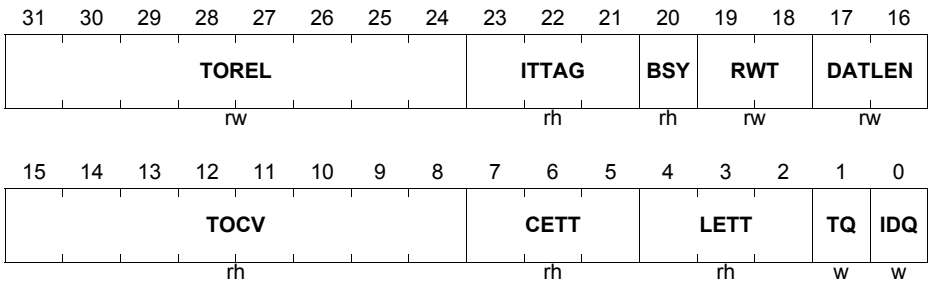
The ICON register contains the control and the configuration bits of a channel.

Bitfields TOREL and ITTAG are used for command frames, and also reused for streaming frames in channel 2. All other bit fields are used only for command frames.

(>> [Table 21-6](#) register overview)

ICONx (x=0-3)

Initiator Control Data Register x ($34_H + x*10_H$) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
IDQ	0	w	Read ID Request This bit provides the only way to request a read ID frame. Reads always 0. Write of 1 commences a request. In case of parallel write of 1 to TQ and IDQ, the IDQ request has higher priority. 0_B No read ID request 1_B Read ID frame request
TQ	1	w	Trigger Request This bit provides an alternative way to request a trigger frame, without having to write to the channel address register. Reads always 0. Write of 1 commences a request. In case of parallel write of 1 to TQ and IDQ, the IDQ request has higher priority. 0_B No trigger request 1_B Trigger frame request
LETT	[4:2]	rh	Last Error Transaction Tag
CETT	[7:5]	rh	Currently Expected Transaction Tag
TOCV	[15:8]	rh	Time Out Current Value

High Speed Serial Link (HSSL)

Field	Bits	Type	Description
DATLEN	[17:16]	rw	Data Length Defines the length of the data in bits of the write and read command. 00 _B 8-bit 01 _B 16-bit 10 _B 32-bit 11 _B reserved (implemented as 32-bit)
BSY	20	rh	Channel Busy
RWT	[19:18]	rw	Read Write Trigger Command Type Defines if the write to the IRWA register will trigger read, write or trigger request. 00 _B No action 01 _B Read Frame 10 _B Write Frame 11 _B Trigger Frame
ITTAG	[23:21]	rh	Initiator Transaction Tag This bit displays the current value of the three bit counter generating the new transaction tag value.
TOREL	[31:24]	rw	Time Out Reload Value Defines the duration of the timeout in units of prescaled clock periods. This parameter is valid both in command and stream mode of the channel 2.

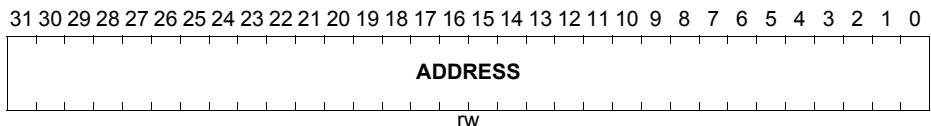
IRWA

The IRWA register contains the address part of a write command. Writing the IRWA register triggers a transmit request for the appropriate channel.

(>> [Table 21-6](#) register overview)

IRWAx (x=0-3)

Initiator Read Write Address Register(38_H + x*10_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
ADDRESS	[31:0]	rw	Address Part of the Payload of a Write Frame Writing to this registers triggers transmission of a Write Frame. The address must be aligned according to the data width: byte addresses for 8-bit data, word addresses for 16-bit data, double word addresses for 32-bit data.

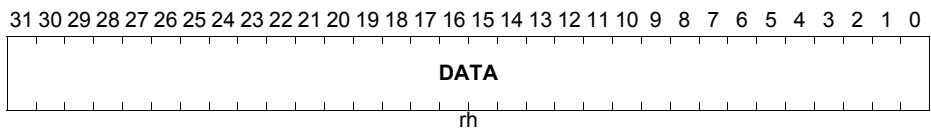
IRD

The IRD register contains the data read from the target, which has been fed back as a response to a read command.

(>> [Table 21-6](#) register overview)

IRD_x (x=0-3)

Initiator Read Data Register ($3C_H + x*10_H$) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
DATA	[31:0]	rh	Data Delivered by a Read Response Frame

21.13.4 Channel.Target Registers

This section describes the channel specific registers of the HSSL module, associated to both the initiator and the target functions.

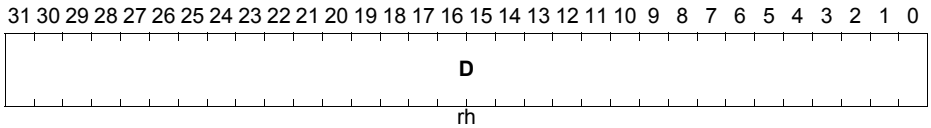
TCD

The TCD register contains the data part of a write command which is currently being processed by the channel on the target side, or the latest data which has been read by the SRI/SPB master in case of a read command.

(>> [Table 21-6](#) register overview)

TC Dx (x=0-3)

Target Current Data Register x ($70_H + x*8_H$) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
D	[31:0]	rh	Data Part of the Payload of a Write Command Frame or Read Data of a Read Command Frame

TCA

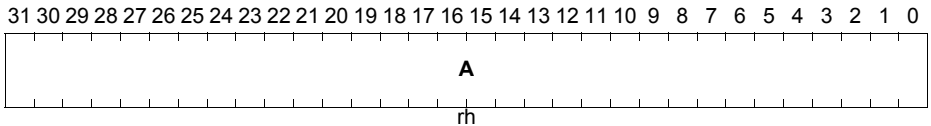
The TCA register contains the address part of a read or write command which is currently being processed by the channel on the target side.

In case of Read ID frame, the address is copied from the register [TIDADD](#).

(>> [Table 21-6](#) register overview)

TC Ax (x=0-3)

Target Current Address Register x ($74_H + x*8_H$) **Reset Value: 0000 0000_H**



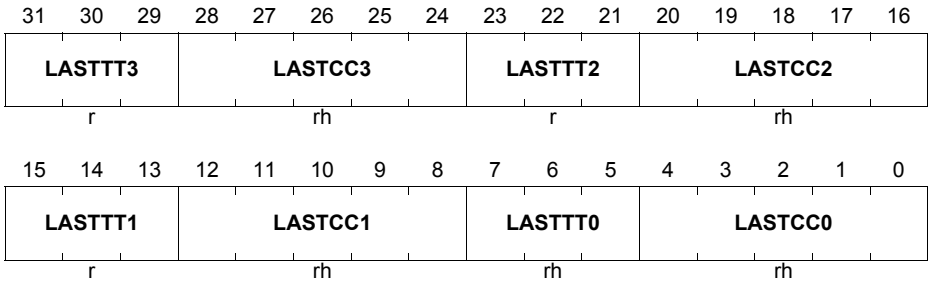
High Speed Serial Link (HSSL)

Field	Bits	Type	Description
A	[31:0]	rh	Address Part of the Payload of a Write Command Frame or a Read Command Frame or ID Frame

TSTAT

The TSTAT register contains the status bits of the common target functionality.

(>> [Table 21-6](#) register overview)

TSTAT
Target Status Register
(90_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
LASTCCx (x=0-3)	[8*x+4: 8*x]	rh	Last Command Code Indicates the latest command code from the header for the corresponding channel.
LASTTTx (x=0-3)	[8*x+7: 8*x+5]	rh	Last Transaction Tag Indicates the transaction tag of the latest command or stream frame for the corresponding channel.

High Speed Serial Link (HSSL)

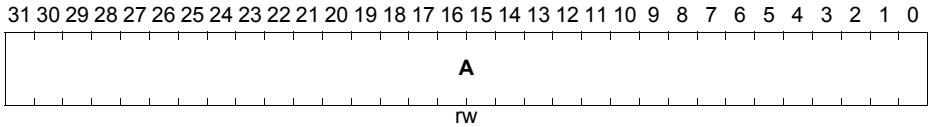
TIDADD

The TIDADD register contains the address from which the target ID is fetched.

(>> [Table 21-6](#) register overview)

TIDADD

Target ID Address Register (94_H) **Reset Value: 0000 0000_H**



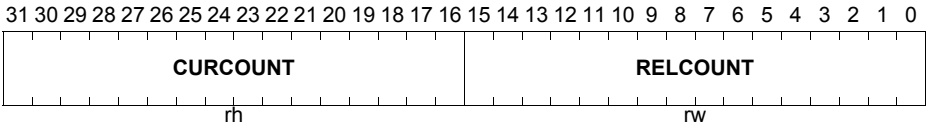
Field	Bits	Type	Description
A	[31:0]	rw	Address Pointer Address pointer containing the address of the memory location containing the unique ID data.

High Speed Serial Link (HSSL)

ISFC

Initiator Stream Frame Count Register(AC_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RELCOUNT	[15:0]	rw	<p>Reload Count Number Contains the number of frames to transfer per memory block. Bit field length depends on application requirements.</p> <p>0_D 1 1_D 1 2_D 2 3_D 3 others, corresponding number of frames</p>
CURCOUNT	[31:16]	rh	<p>Current Count Number Displays the current count number, which is generated by down-counting from the RELCOUNT value. Bit field length depends on application requirements.</p>

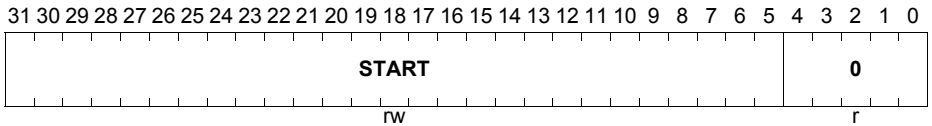
21.13.6 Target Stream Registers

(>> [Table 21-6](#) register overview)

(>> [Memory Block Transfer Modes of the Stream Channel](#))

TSSAx (x=0-1)

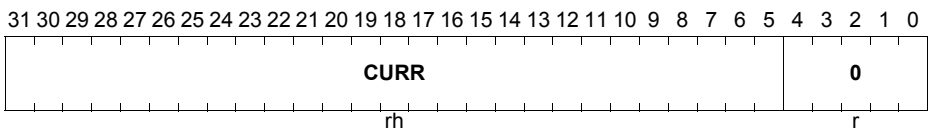
Target Stream Start Address Register x(B0_H + x*4_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
ADDR	[31:5]	rw	Start Address for the Memory Range Aligned on 256-bit (or 32 byte) limit (stream frame payload size).
0	[4:0]	r	reserved

TSCA

Target Stream Current Address Register (B8_H) **Reset Value: 0000 0000_H**



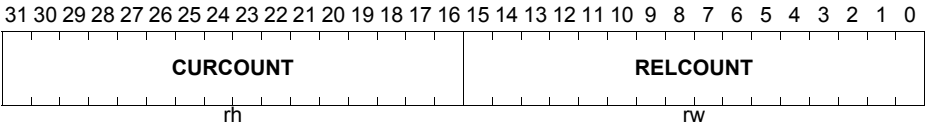
Field	Bits	Type	Description
CURR	[31:5]	rh	Address of the Memory Location for the Current Transfer Aligned on 256-bit (or 32 byte) limit (stream frame payload size).
0	[4:0]	r	reserved

High Speed Serial Link (HSSL)

TSFC

Target Stream Frame Count Register (BC_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RELCOUNT	[15:0]	rw	<p>Reload Count Number Contains the number of frames to transfer per memory block. Bit field length depends on application requirements.</p> <p>0_D 1 1_D 1 2_D 2 3_D 3 others, corresponding number of frames</p>
CURCOUNT	[31:16]	rh	<p>Current Count Number Displays the current count number, which is generated by down-counting from the RELCOUNT value. Bit field length depends on application requirements.</p>

21.13.7 Access Protection Registers

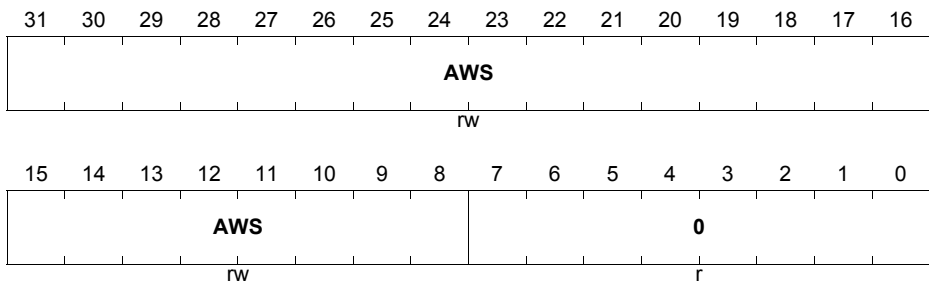
This sub section describes the registers defining the access windows where the HSSL accesses are allowed. These registers are writable only in supervisor mode, which makes them not writable by the HSSL itself.

(>> [Table 21-6](#) register overview)

AWSTARTx

AWSTARTx (x=0-3)

Access Window Start Register x ($C0_H + x*8_H$) **Reset Value: 0000 0000_H**

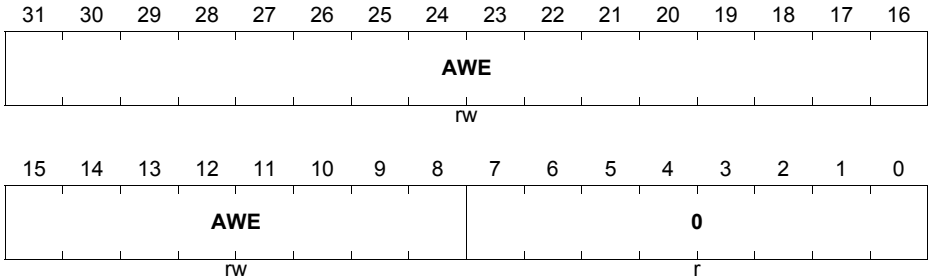


Field	Bits	Type	Description
AWS	[31:8]	rw	Access Window Start Address This bit field defines the upper 24 bits of the start address of the corresponding access window. This results in a granularity of 256 bytes for the start address.
0	[7:0]	r	reserved

AWENDx

AWENDx (x=0-3)

Access Window End Register x ($C4_H + x*8_H$) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
AWE	[31:8]	rw	Access Window End Address This bit field defines the upper 24 bits of the end address of the corresponding access window. This results in a granularity of 256 bytes for the end address.
0	[7:0]	r	reserved

High Speed Serial Link (HSSL)

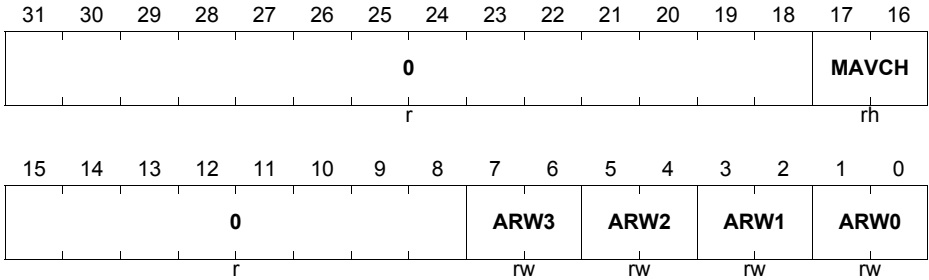
AR

AR

Access Rules Register

(E0_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
ARWx (x=0-3)	[1+2*x: 0+2*x]	rw	Access Rule for Window x 00 _B No access (window disabled) 01 _B Read access allowed 10 _B Write access allowed 11 _B Read and write access allowed
MAVCH	[17:16]	rh	Memory Access Violation Channel This bit field shows the number of the latest channel that attempted a not allowed access. 00 _B Channel 0 01 _B Channel 1 10 _B Channel 2 11 _B Channel 3
0	[31:18], [15:8]	r	reserved

21.14 Module Implementation

The following section describes the topics regarding the integration of the module on chip.

21.14.1 BPI_SPB Module Registers

21.14.1.1 System Registers

Figure 21-38 shows all registers associated with the BPI_FPI module, configured for one kernel. The Offset Address in the following BPI_FPI register table and in the BPI_FPI register descriptions are proposals. In a standard 32 bit peripheral the CLC is mapped to offset address 00h, module ID to 08h. The proposal for the new BPI_SPB register is to map them to the end address of the peripheral address range together with the peripherals SRNs.

BPI_FPI Registers Overview

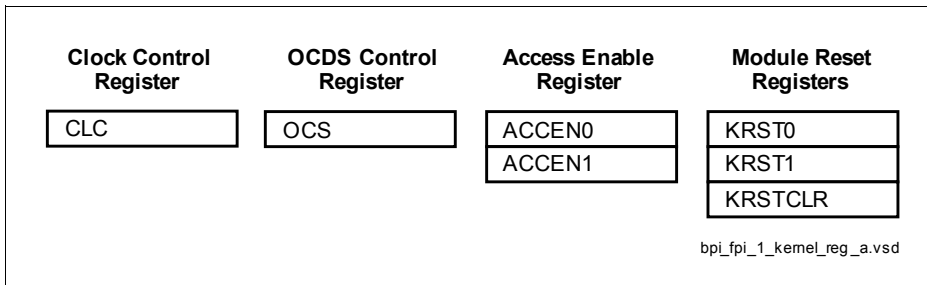


Figure 21-38 BPI_FPI Registers

The writes of the bus masters to the HSSL module is controlled by Acces Protection registers ACCENx.

The HSSL implements two ACCENx registers, ACCEN0 and ACCEN1.

The ACCENx registers are protected by Safety Endinit mechanism.

Clock Control Register (CLC)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI_FPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the f_{\leftarrow} module clock signal, sleep mode and disable mode for the module.

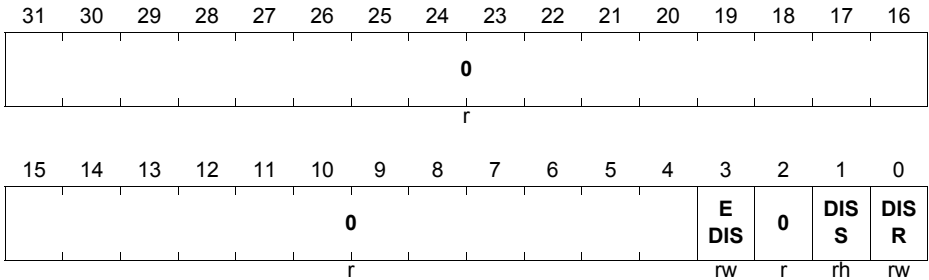
High Speed Serial Link (HSSL)

CLC

Clock Control Register

(00_H)

Reset Value: 0000 0003_H



Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
EDIS	3	rw	Sleep Mode Enable Control Used to control module's sleep mode.
0	[31:4], 2	r	Reserved Read as 0; should be written with 0.

OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32 bit wide only and requires Supervisor Mode

High Speed Serial Link (HSSL)
OCS
OCDS Control and Status
(E8_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUS STA	SUS _P	SUS				0							
r		rh	w	rw				r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TG _P	TGB	TGS	
r												w	rw	rw	

Field	Bits	Type	Description
TGS	[1:0]	rw	Trigger Set for OTGB0/1 0 _H No Trigger Set output 1 _H TS16_STR, Streaming Channel Trigger Set 2 _H TS16_ERR, Errors Trigger Set 3 _H reserved
TGB	2	rw	OTGB0/1 Bus Select 0 _B Trigger Set is output on OTGB0 1 _B Trigger Set is output on OTGB1
TG_P	3	w	TGS, TGB Write Protection TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately. 2 _H Soft suspend others , reserved,
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:4], [31:30]	r	Reserved Read as 0; must be written with 0.

High Speed Serial Link (HSSL)

Note: The bit *SUSSTA* is set only at entering the suspend state from the run state, according to the [Figure 21-29](#). It will not be set if the soft suspend request comes in any other state than run state, like init or disabled/sleep state.

Access Enable Register (ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B,..., EN31 -> TAG ID 011111B.

ACCEN0
Access Enable Register 0
(FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
ENn (n = 0-31)	n	rW	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register (ACCEN1)

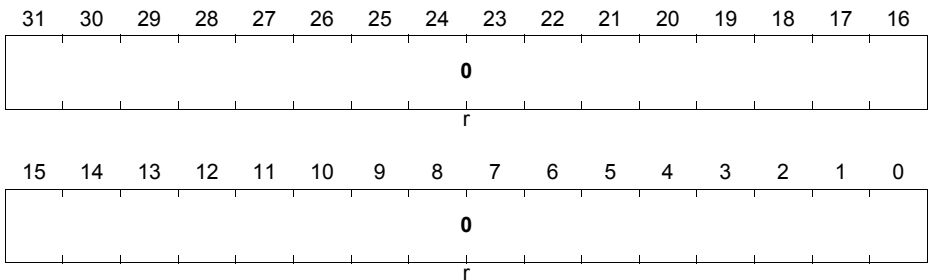
The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

High Speed Serial Link (HSSL)

TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000B, EN1 -> TAG ID 100001B, ..., EN31 -> TAG ID 111111B.

ACCEN1
Access Enable Register 1
(F8_H)
Reset Value: 0000 0000_H


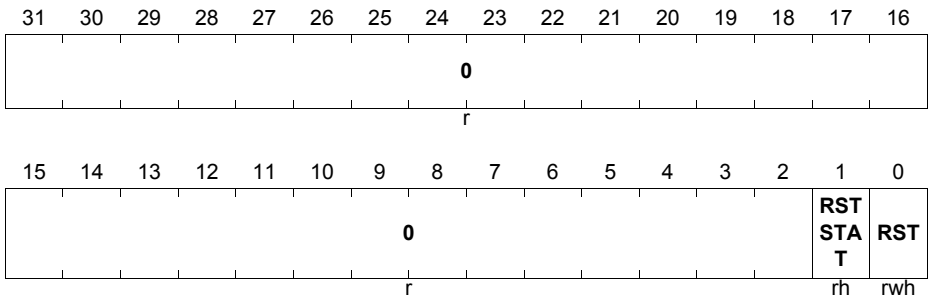
Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLR.CLR register bit.

During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

KRST0
Kernel Reset Register 0
(F4_H)
Reset Value: 0000 0000_H


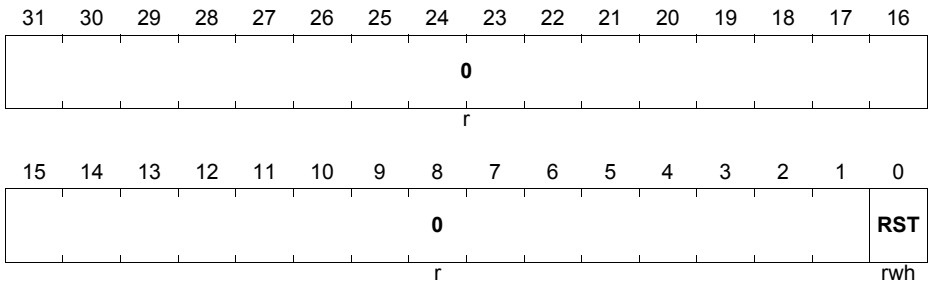
Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
RSTSTAT	1	rh	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits. 0 _B No kernel reset was executed 1 _B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST

High Speed Serial Link (HSSL)

bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

KRST1
Kernel Reset Register 1
(F0_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Status Clear Register (KRSTCLR)

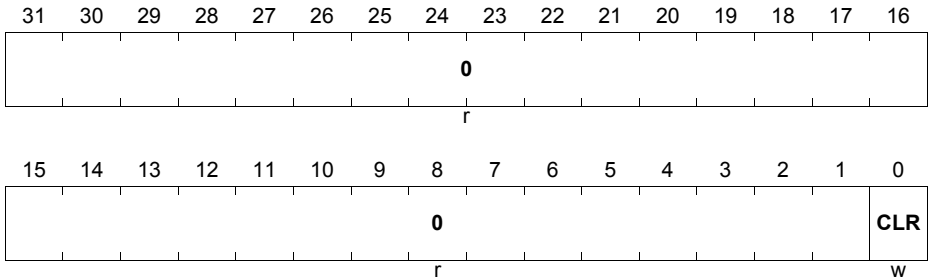
The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

High Speed Serial Link (HSSL)

KRSTCLR

Kernel Reset Status Clear Register (EC_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

21.15 High Speed Communication Tunnel (HSCT)

This chapter describes the High Speed Communication Tunnel interface for the Microcontroller high speed data communication including the physical layer and the control (link) layer.

21.15.1 Overview

The HSCT chip to chip communication interface employs an interface for inter chip communication between a master interface and a slave interface. The interface is based on LVDS IO technology and developed to function as a master or a slave interface. During initialization phase the role of the Interface (master or slave) must to be defined. It is not intended to change the system role during an application.

The interface is a full-duplex interface for TX and RX data transfer. The data transfer speed can be configured to function in low or high speed and allows to configure in each transfer direction an independent transfer speed in order to reduce power and EMI, based on transfer speed requirements. A sleep mode for the LVDS IO's is implemented, which reduces power on the IO, if the interface is not in use.

The master interface is defined to own the crystal and provides the clock (**SysClik**) to the slave interface. The slave interfaces allows a master interface generated SysClik routing through the SysClik IO or the XTAL1 IO. The second one allows to use SysClik as reference clock for the System PLL. The HSCT Master/Slave interface in the system allows one crystal only. The interface reset is derived from the System reset and provided by chip internal reset signaling.

In **Figure 21-39** a system level view of the interface and interface connection is shown.

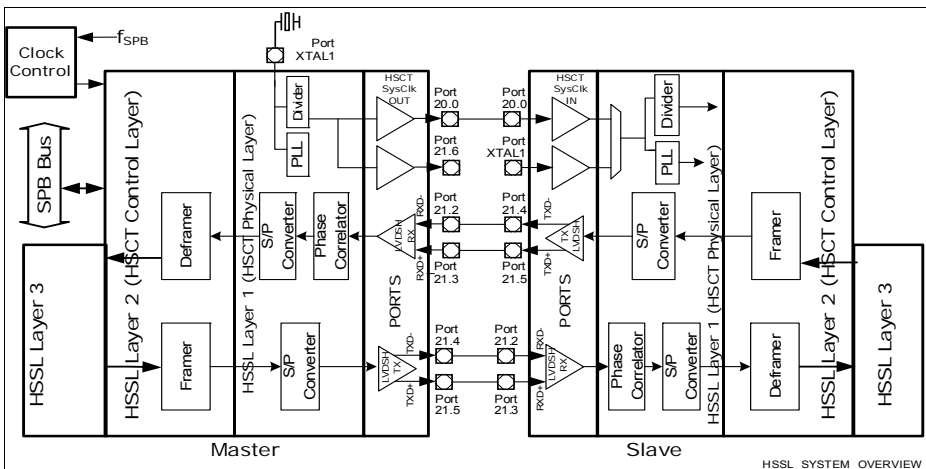


Figure 21-39 Overview of Interface Signals.

Note: The HSCT SysClk path from Master to Slave device can be done using Master Port 20.0 (5 V) or using Port 21.6 (3.3V). Only one of this two signal path shall be used on a PCB design. It is recommended to use P20.0 signaling path. HSCT SysClk connection to XTAL1 is not supported by FlexRay PLL.

21.15.1.1 Features

The HSCT interface is intended to carry:

- Transmit Symbol information from the master interface to the slave interface
- Receive Sample information from the slave interface to the master interface
- System Clock provided by the master interface (crystal owner) to the slave interface
- Control information on Master TX link and RX link
- Clear To Send indication in both directions
- Unsolicited Status in both directions
- Regular data transfer in both directions based on data channels
- Master interface data transfer speed 5 MBaud and 320 MBaud.
- Three slave interface data transfer speeds available based on 20 MHz SysClk:
 - 5 MBaud (low speed)
 - 20 MBaud (medium speed)
 - 320 MBaud (high speed)
- Two slave interface data transfer speeds available based on 10 MHz SysClk:
 - 5 MBaud (low speed)
 - 320 MBaud (high speed)
- The interface is based on IEEE 1596.3 LVDS IO's
- To reduce the voltage swing a configuration option is available

21.15.2 Functional Description

21.15.2.1 Introduction

The first HSCT communication is always established from master interface. The LVDS IO's are based on standard IO technology following IEEE 1596.3 specification. The HSCT LVDS IO common voltage is defined to be a 1.2V. Compared to IEEE 1596.3 the differential voltage level can be reduced via configuration option to reduce EMI. Additionally the HSCT SysClk can be configured to run at Crystal/2 to reduce EMI instead of Crystal frequency used as single ended clock.

The intention is to implement the interface either as a master interface or as a slave interface. So the interface is able to handle both roles (master or slave). The following detailed description should describe both roles. This requires the following definitions:

- TX line or TxData link (TX_DAT, TX_DATX) defines the data transfer from the master interface to the slave interface.
- RX line or RxData link (RX_DAT, RX_DATX) defines the data transfer from the slave interface to the master interface.
- Transmit direction, outbound data (or transmission) describes to send out data independent of the role the interface does have.
- Receive direction, inbound data (or receiving) describes to receive data independent of the role the interface does have.

21.15.2.2 Physical Layer

RX / TX Data Interface

During initialization the associated TxData link starts in low-speed mode, with the high-speed clock generators turned off and the RxData Link disabled. An interface control sequence, initiated by software, allows to switch the interface in high-speed mode.

The data interface can transmit up to 320 Mbit/s in the RX and the TX direction (full-duplex). The data is recovered from RX data stream (RX_DAT, RX_DATX) in the master interface and from TX (TX_DAT, TX_DATX) data stream in the slave interface.

Table 21-7 shows the interconnects of the differential data lines for control and data transfer between the master interface and the slave interface.

In transmit direction a 320 Mbit/s data signal is transmitted from the master interface to the slave interface. Proper phase selection is processed at receiver side via six different phases of a 320 MHz clock.

In RxData direction a 320 Mbit/s differential data signal is supplied from the slave interface to the master interface. A proper phase selection is processed for data recovery at the master receiver path via six different phases of a 320 MHz clock. The slave interface reference clock must be the SysClk, driven by the interface Master. The Slave must receive the SysClk from the master interface to be enabled to transfer data. In the slave interface the 320 MHz data clock is generated internally from the SysClk. The slave interface PLL reference clock is based on the **SysClk** interface signaling or on a clock provided via the Oscillator input. The selection can be done via configuration register setting. Also the slave receiver path is using the six different phases of a 320 MHz clock to recover the data.

Note: A local crystal at the slave interface taken as reference clock for the interface PLL is not allowed as the data recovery based on clock phase shift does work on a common reference clock source! It is allowed to use the oscillator input of the chip and connect the master interface SysClk to it. In this case also other chip internal PLL's can be supplied with the SysClk master interface reference clock.

Table 21-7 Physical Interconnects

Direction	Name	Description
Transmit	TX_DAT, TX_DATX	Channel TX data and control information at a max rate of 320 Mbit/s from the master interface to the slave interface
Receive	RX_DAT, RX_DATX	Channel RX data and control information at a max rate of 320 Mbit/s from the slave interface to the master interface

The interface driver in shutdown mode provides a pull-down to 0V on both differential pins to ensure that the signals are held in a defined state. The RxDAT link enters shutdown mode by Interface control logical channel command and exit shutdown mode by control command, again. The slave transmission line is disabled by default and must be enabled by the master. The master is using control commands to link operation modes.

Payload content is transferred most significant bit first. For example, if a 32-bit payload contains a message that is built in a 32-bit register in the sending interface, b31 of the register (the MSb) shall be the first bit of payload transmitted and b0 (LSb) shall be the last. Larger payloads should be built up according to the same principle.

Differential Signaling Principle Based on LVDS

The voltage at the termination resistor of the receiver is either positive or negative, which depends on the direction of the current flowing through it. This is determined by the state of the switches at the transmit stage. The voltage V_{OD} should be evaluated by a comparator with hysteresis.

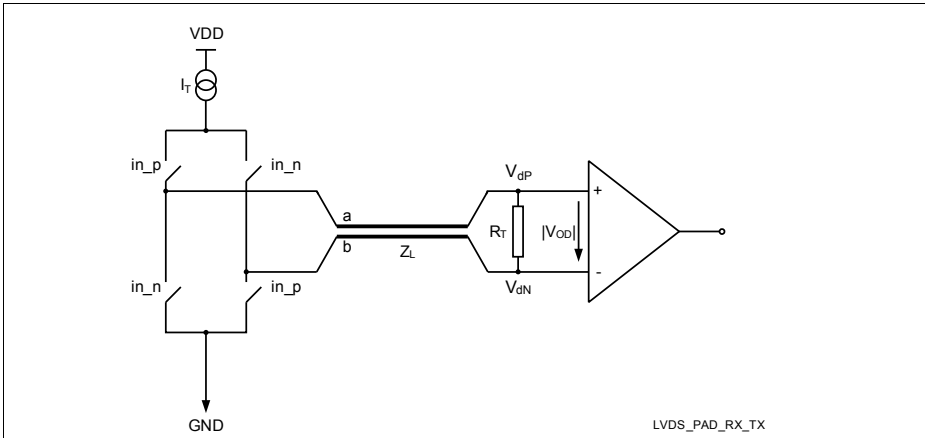


Figure 21-40 Pad Driver and Pad Receive

The HSCT interface does not define specific test for an error-free transmission, although the signal quality of course suffers from attenuation and noise and in particular it depends on proper wave propagation. Therefore the interconnect lines need to have a characteristic wave impedance Z_L equal to $100\ \Omega$ differential, corresponding to the termination resistor R_T at the receiver side.

Note: Due to the nature of wave propagation a termination other than with the characteristic line impedance leads to inevitable signal degradations. Therefore any operation mode other than $100\ \Omega$ termination is only supported from a functional point of view (i.e. the corresponding electrical configuration is possible) without any further guarantee of an error-free transmission.

Note: The interface receiver provide an input termination impedance of $100\ \Omega \pm 20\%$ at the IO. If a more precise termination is required the design offers to disable the internal termination impedance and allow an external termination on the PCB.

The differential output voltage V_{OD} is defined as the difference of the voltages V_{dP} and V_{dN} at the receiver input pins.

$$V_{OD} = V_{dP} - V_{dN} \quad (21.1)$$

The common-mode voltage V_{OS} is defined as the arithmetic mean value of the voltages at the receiver input pins:

(21.2)

$$V_{OS} = \frac{V_{dP} + V_{dN}}{2}$$

V_{OD} , V_{OS} , and the corresponding minimum differential eye opening V_{eye} considering LVDS transmitter jitter are graphically shown in [Figure 21-41](#) and [Figure 21-42](#) for ideal signals.

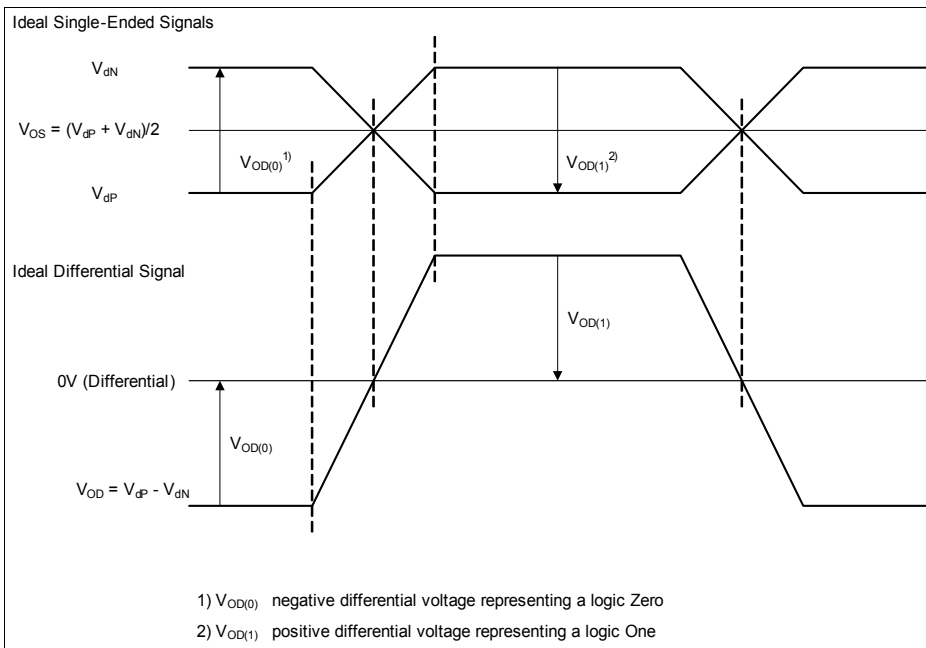


Figure 21-41 Ideal Single-Ended and Ideal Differential Signals

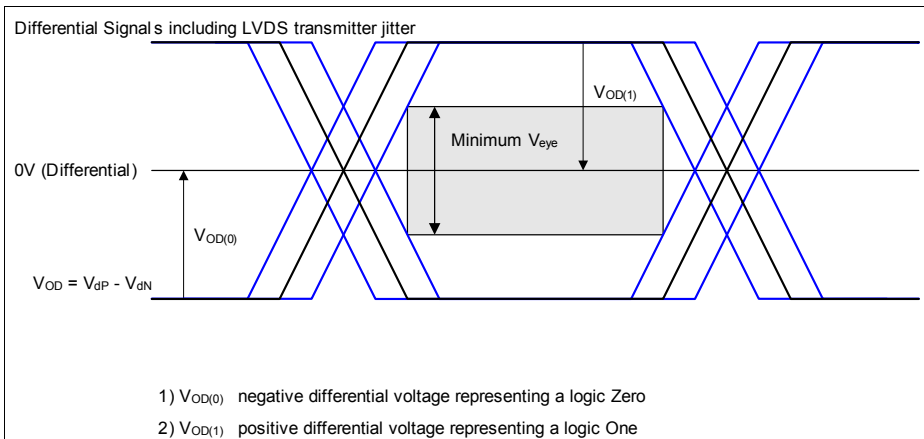


Figure 21-42 Differential Eye Opening

21.15.2.3 Electrical Characteristics Based on LVDS for Reduced Link trace length

The characteristics are based on the LVDS standard IEEE P1596.3-1995. It is adapted for low power consumption and lower supply voltages for short ranges. It is assumed that the master interface and the slave interface may have different supply voltages, but common mode voltage for the differential signaling is based on 1.2V.

Operating Modes

The HSCT interface comprises the following operating modes:

- Terminated Mode:** The data transmitter requires a 100 Ω external termination at the pins **TX_DAT** and **TX_DATX**. The termination is given by a 100 Ω resistor at the receiver input, of the opposite LVDS device, for matching the differential characteristic impedance of the line with the impedance of the receiver. PCB design has to follow guidelines on differential signal routing. The PCB placed termination resistor can be chosen very precise. Via configuration option an 100 Ω LVDS receiver internal resistor can be enabled with a +/- 20% tolerance.
- Power up:** The line receiver shall always be enabled after power up and SysClk is active. The data decoder behind the receiver is not enabled at this point. It is enabled when the first transition on the data lines is detected by the line receiver. The line driver must be enabled on the master side via SW control. On the slave side the line driver must be enabled by sending interface message control commands from the master. After power up the link starts in low speed mode. After line driver enable some time is required to power up the LVDS transmitter line. The LVDS driver will be activated after a startup time of max. 3 ms. The data link in RX and TX direction is ready to transmit and receive data after the defined setup time.

- **Sleep mode:** In terminated operation the differential voltage across the resistor will be reduced to nominally zero (-5mV to 20mV at 100 Ω +/- 20%) and hence, the line driver current requirement will reduce to internal bias currents. The common-mode voltage, however, should be maintained. Also the PLL is not switched off during sleep mode and being in high speed mode.
- **Power down:** When any interface driver supply voltage is removed, the driver shall provide a weak pull-down to 0 V to ensure that the signals are held in a well-defined state. It is a requirement to be free of cross- and bias current during power down.

Note: The line driver in the master interface shall be turned on before the line driver in the slave interface is activated, i.e. before sending the transmitter activation command to the slave interface, in order to avoid receiving undefined data in the slave interface.

The implementation of the output driver is based on a bridge topology of current sources. Accordingly the equivalent differential source impedance is high-ohmic (>10k Ω). This mismatch with regards to the immediate termination at the Microcontroller chip edge (100 Ω line) is common practice among LVDS interface designs.

Due to the strong dependency on link length and layout, as well as the susceptibility to noise and interference, it is recommended, to have the termination available on the PCB:

- Receiver **internal termination mode** not active (PCB termination): can be programmed in register **INIT**. The data transmitter is operated without a termination inside the differential receiver pair.
- Receiver **internal termination mode** active: allows to have a 100 Ω resistor internally with a +/-20% tolerance. Programming via the same register bit as PCB termination (in register **INIT**).

Data Rates

Both, master and slave interface shall be ready to receive frames whenever the SysClk is running. The link shall initialize in low-speed mode with the high-speed clock generators turned off and the RxData link disabled. Subsequent speed and mode changes shall be commanded by the master interface, except sleep mode on the RxData link, which is controlled by the slave interface. Except when using sleep mode, both links shall default to the logic zero state, while dormant and between frames. This is required, to ensure that data will not be lost during a speed change.

Interface data rate change on both the TxData and the RxData link shall be possible without losing data. It is highly recommended to keep the link quiet during the time the interface transfer speed is changed. Changing transfer data rate on an active link does not guarantee stable data transfer.

The HSCT standard defines three different data speed modes:

- Low speed mode
- Medium speed mode
- High speed mode

Additionally the HSCT standard defines the different data speed modes to be derived from the crystal frequency. Taking the Automotive common crystal frequency of 20 MHz the following data rates can be produced based on the available multiplication factors.

SysClk; RefClk = Crystal frequency = 20 MHz:

- 5 MHz = Crystal frequency / 4 (Low Speed)
- 20 MHz = Crystal frequency (Medium Speed)
- 320 MHz = Crystal frequency * 16 (High Speed)

SysClk; RefClk = Crystal frequency / 2 = 10 MHz:

- 5 MHz = SysClk frequency / 2 (Low Speed)
- 20 MHz = Medium Speed - **not available**
- 320 MHz = SysClk frequency * 32 (High Speed)

Low Speed mode and Medium Speed mode frequencies are generated directly out of the Crystal frequency. High Speed mode frequency requires a PLL, which takes the Crystal clock as reference clock.

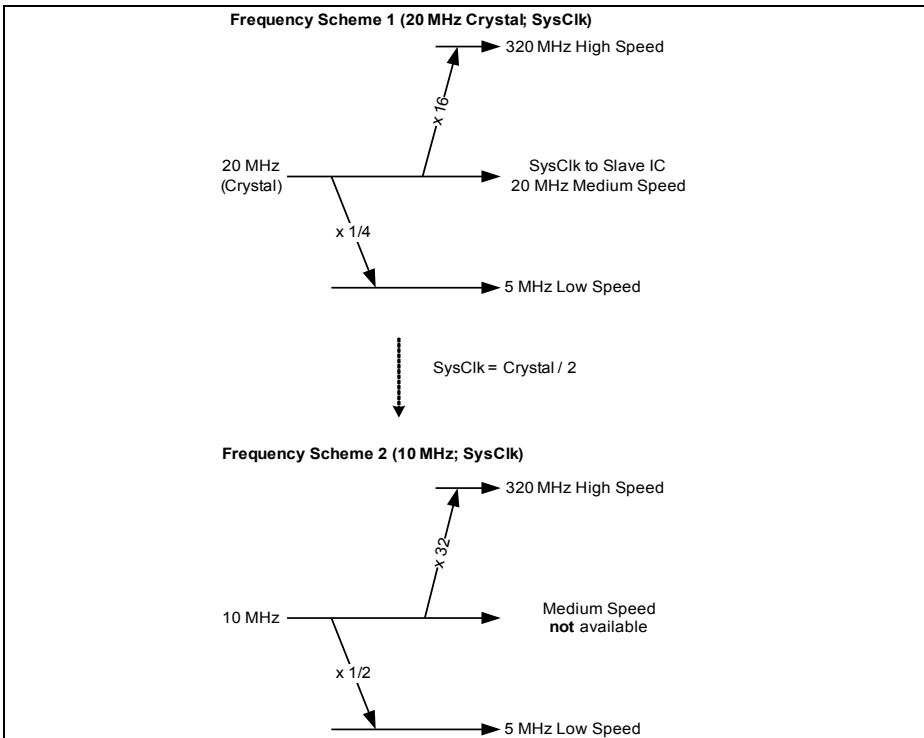


Figure 21-43 Frequency Scheme

Connecting a 20 MHz Crystal to the system allows to generate the Automotive frequency scheme.

The following table (Table 21-8) shows the data transfer direction and the available Automotive transfer speed based on Crystal clock frequency.

Table 21-8 HSCT Data Rates

Direction	Data Rate	Operating Mode
Master interface to Slave interface	5 Mbit/s	“low speed mode” for startup / fallback
Master interface to Slave interface	320 Mbit/s	“high speed mode”
Slave interface to Master interface	5 Mbit/s	“low speed mode” for startup / fallback

Table 21-8 HSCT Data Rates (cont'd)

Direction	Data Rate	Operating Mode
Slave interface to Master interface	20 Mbit/s	“medium speed mode” <i>Note: only available, if reference clock; SysClk is at 20 MHz (Crystal)</i>
Slave interface to Master interface	320 Mbit/s	“high speed mode”

*Note: To fulfill the requirement - having reduced EMI on the single ended **SysClk** - the interface can be configured to operated on a **SysClk = Crystal/2** frequency. Choosing the **crystal/2** mode excludes the medium speed mode available in slave to master data transfer direction (receiver line). Low speed mode devices divide SysClk by 2 to achieve the low speed clock. Another possibility to reduce EMI is reducing SysClk driver strength based on the 20 MHz clock, which is the recommended way as BER is better than using 10 MHz.*

PLL configuration and Baud Rates

The PLL supports different speed modes and frequency modes, which will be described in this section in detail. Each mode and transfer direction can be configured separately.

Table 21-9 High Speed Mode selection

PLL Speed [MHz]	320	320	280	280	240	240	200	200
PLL reference frequency [MHz]	20	10	20	10	20	10	20	10
PLLWMF [hex]	10	20	0E	1C	0C	18	0A	14
TXHD 1/1 [MBaud]	320	320	280	280	240	240	200	200
TXHD 1/2 [MBaud]	160	160	140	140	120	120	100	100
TXHD 1/4 [MBaud]	80	80	70	70	60	60	50	50
TXHD 1/8 [MBaud]	40	40	35	35	30	30	25	25
TXHD 1/16 [MBaud]	20	20	17.5	17.5	15	15	12.5	12.5

PLL reference frequency and PLL frequency control word Multiplication factor (PLLWMF) defines the PLL speed and are used to configure the output frequency of the PLL. The interface baud rate can be configured separately in TX direction (TXHD) and RX direction (RXHD). The table above exemplarily shows the TX direction.

Correlator

Data transfer from the master interface to the slave interface can either be in “high speed mode” or “low speed mode” (**Crystal**/4 at 20 MHz SysClk or SysClk/2 at 10 MHz SysClk). A correlator is used for finding the optimum sample phase for the transfer. The correlator configuration differs in high speed and low speed mode.

Phase Selection

Every frame starts with a 16-bit sync field (see [Figure 21-44](#)) containing a synchronization pattern. Note that the sync pattern starts with 1, so a new frame always starts on the rising edge. In between frames, there is always at least one bit 0 in normal mode. By sending 1 immediately after the last bit in a frame, the master interface controller demands sleep mode as described in Section Sleep Mode. Neither the master interface nor the slave interface may assume that the other side will maintain data phasing between frames. The receiving interface shall use the sync pattern to determine the start time of the frame relative to the local clock.

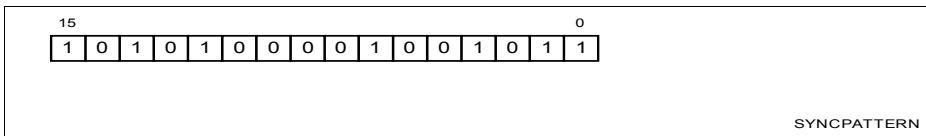


Figure 21-44 Synchronization Pattern

The synchronization pattern is only needed for selecting the optimum sample phase, so it is not feed through to the output register of the correlator.

Payload Size Decoding

Every frame consists of three parts: a 16-bit synchronization pattern, an 8-bit header and the payload field which can be 8, 32, 64, 96, 128, 256 or 512 bits. The header contains information about payload size and logical channel type. Although decoding the header is task of the deframer, the payload size shall also be evaluated inside the correlator for finding the end of a frame.

The header is sent most significant bit first. The first 3 bits (bit5, bit6 and bit7) contain the payload size of this frame (see [Table 21-10](#)).

Table 21-10 Payload Size Coding

b7 ... b5	Payload Size (bits)	Total Frame Size (bits)
000	8	32
001	32	56
010	64	88
011	96	120
100	128	152

Table 21-10 Payload Size Coding (cont'd)

b7 ... b5	Payload Size (bits)	Total Frame Size (bits)
101	256	280
110	512	536 (not supported)
111	288	312

Note: In a harsh Automotive environment a payload of 512 bit may cause increased error bit streams, due to the long distance from one synchronization pattern to the next. This is the reason why this data transfer length is not supported.

The correlator shall decode the payload size field and count the number of received bytes. After the last byte in a frame has been fed through to the correlator output register, the tuning of the multiplexer unit becomes invalid. In between frames at least one bit 0 shall be transmitted. The comparison procedure is started again as soon as there is a 1 at the MSB position in the shift registers.

The 8-bit header and the payload are used as input for the deframer block.

Note: The sync pattern and inter-frame bits are not passed to the deframer. An error in the sync pattern generates an error and is detected by the Physical layer.

Transmission initiated Sleep Mode - bringing receiver path into sleep mode.

By sending a 1 immediately after the last bit in a frame, the interface indicates that its LVDS line driver is sent to sleep mode, and suggests sleep mode to the receiver LVDS.

To leave sleep mode, the master interface shall send 8 bits of 0. This is performed in high speed mode and in low speed mode.

Jitter Budget

The system clock is distributed directly from the master interface to the slave interface.

Subsequently, all jitter investigations considered in the following sections are referenced to the system clock, i.e., 20 MHz as the base oscillator frequency or 10 MHz as the base oscillator frequency divided by 2.

Each system clock distribution has an associated filter function that comprehends the worst case combination of PLL bandwidth/peaking and equivalent jitter. The effective jitter seen at the LVDS receivers clock-data recovery inputs is a function of the difference in LVDS receiver and LVDS transmitter PLL bandwidth and peaking convolved with the jitter spectrum of the system clock.

System Clock variations contain jitter over a wide range of frequencies, some of which will be tracked by the LVDS receiver or otherwise removed by the combination of the LVDS transmitter and the LVDS receiver PLLs. Consequently, the nature of the filter functions is in part dependent on the way of how the system clock is distributed. In general, the system clock jitter is filtered by the PLL difference function which

comprehends the “mismatch” between LVDS transmitter and LVDS receiver PLL and also accounts for the impact of the transport delay.

Direct System Clock Distribution

This architecture implies direct distribution of the crystal clock as system clock from the master interface to the slave interface. Hence, the system clock is directly supplied to both LVDS transmitter PLL and LVDS receiver PLL. Hence, a major part of the clock jitter is sourced equally through LVDS transmitter and LVDS receiver PLLs. The amount of jitter appearing at the data recovery block is then defined by the transfer function difference of the LVDS transmitter and LVDS receiver PLLs, and by the sum of the clock jitter of the LVDS transmitter PLL, $X_{\text{clkMaster}}(s)$, and the LVDS receiver PLL, $X_{\text{clkSlave}}(s)$.

Based on the above clock architecture, a difference function may be defined that describes the “mismatch” between LVDS transmitter and LVDS receiver PLLs. As a first approximation second order transfer functions are assumed.

The jitter of the direct system clock distribution is described by the phase noise power spectral density function as the absolute square of $X_d(s)$, as follows.

$$|X_d(s)|^2 = |X(s)|^2 |H_1(s)e^{-sT} - H_2(s)|^2 + |X_{\text{clkMaster}}(s)|^2 + |X_{\text{clkSlave}}(s)|^2 \quad (21.3)$$

where

$$H_1(s) = \frac{2s \zeta_1 \omega_{n1} + \omega_{n1}^2}{s^2 + 2s \zeta_1 \omega_{n1} + \omega_{n1}^2} \quad (21.4)$$

and

$$H_2(s) = \frac{2s \zeta_2 \omega_{n2} + \omega_{n2}^2}{s^2 + 2s \zeta_2 \omega_{n2} + \omega_{n2}^2} \quad (21.5)$$

Conversion between the natural PLL frequency ω_n and the -3 dB point is given by the following expression.

$$\omega_{3dB} = \omega_n \sqrt{1 + 2\zeta^2 + \sqrt{(1 + 2\zeta^2)^2 + 1}} \quad (21.6)$$

Under the assumption that crystal clock jitter can be ignored, i.e.,

$$X(s) \approx 0 \quad (21.7)$$

and LVDS transmitter to LVDS receiver transport delay can be neglected, i.e.,

$$e^{sT} \approx 1 \quad (21.8)$$

the phase noise power spectral density of the direct system clock distribution can be simplified, as follows.

$$|X_d(s)|^2 = |X_{clkMaster}(s)|^2 + |X_{clkSlave}(s)|^2 \quad (21.9)$$

Note: The slave interface requires a SysClk from the master always as reference clock. It is not supported by the interface to have a separate crystal connected to the SysClk input as reference clock at the slave interface. The master interface generated SysClk needs to be enabled.

Jitter Budgeting

To estimate the total rms jitter of LVDS transmitter PLL and LVDS receiver PLL for data recovery the following procedure has to be followed.

- Piece wise linear integration of the additive SSB (Single Sideband) phase noise for direct clock distribution to obtain the total phase noise power of the LVDS transmitter PLL.
- Calculation of the rms jitter of the LVDS transmitter PLL as shown in [Figure 21-45](#).
- Calculation of the total rms jitter RJ as the square root of the sum of the squares of the rms jitter from the LVDS transmitter PLL and the LVDS receiver PLL.
- Calculation of total jitter, i.e., random and deterministic, considering duty-cycle distortion and other effects to guarantee minimum eye opening at the interface.

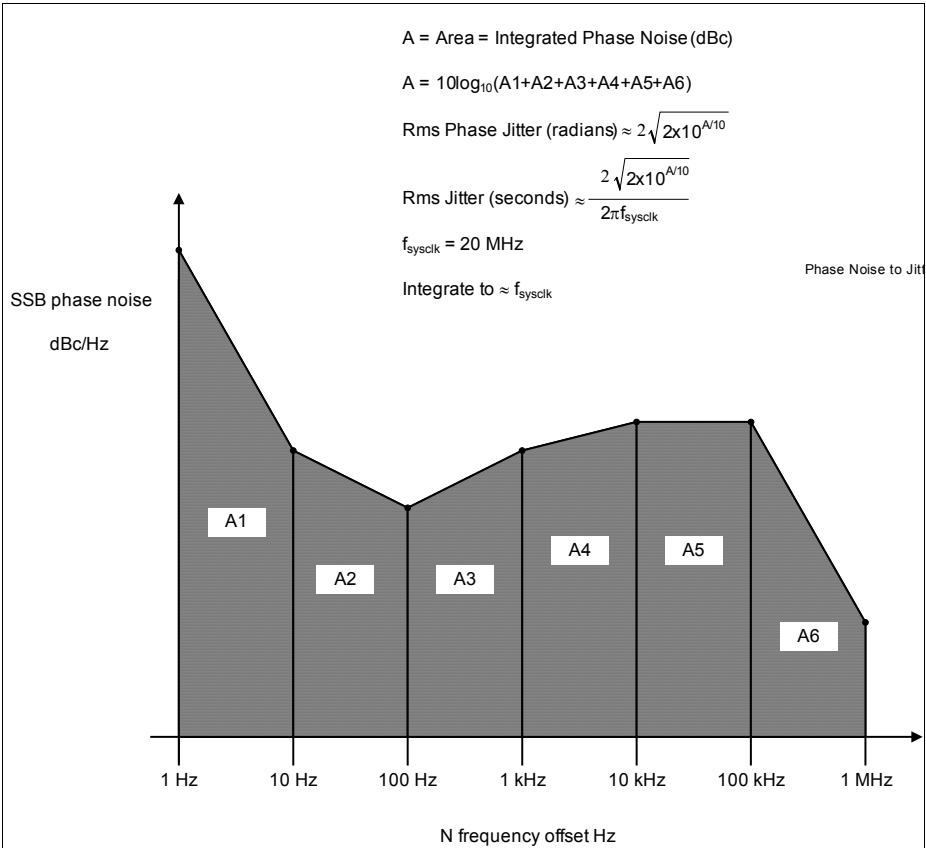


Figure 21-45 Calculation of Jitter from SSB Phase Noise

Calculation of Total Jitter Budget

To estimate the total jitter budget allowed for the random rms jitter *RJ* as described in the sections before, the following procedure for direct clock distribution has to be followed.

- Calculation of the jitter headroom available for the ideal jitter-free case given by the minimum eye opening, i.e., 55% of a 320 MHz period.
- Calculation of the total jitter headroom required under the assumption that the total long-term jitter *TJ* will not exceed the maximum peak-to-peak jitter specified.
- Subtraction of two phases of the oversampling frequency used for data recovery taking into account the data-recovery uncertainty in multiples of phases of the oversampling frequency. The data-recovery uncertainty is implementation specific

and determined by the data-recovery oversampling factor DR_{OS} . **Figure 21-46** depicts the case where the selected sampling phase of the data recovery is almost one sample phase away from the ideal sampling point. Thus, the budget left for the peak jitter is reduced by almost one sample phase and hence, the peak-to-peak jitter is reduced by almost two phases.

- The remaining sum represents the jitter budget which is the upper limit for the peak-to-peak jitter calculated from the random rms jitter distribution RJ described in the sections before. Given a required *bit error rate* $BER = 10^{-12}$ for *SysClk* 20 MHz, and $BER = 10^{-9}$ for *SysClk* 10 MHz.

The jitter budget is given by the following equation.

$$TJ_{pp} = DJ_{pp} + RJ_{pp} = (DJ_{TX} + DJ_{RX}) + 2 * Q * \sqrt{J_{ABS}^2 + J_{ACC}^2} \quad (21.10)$$

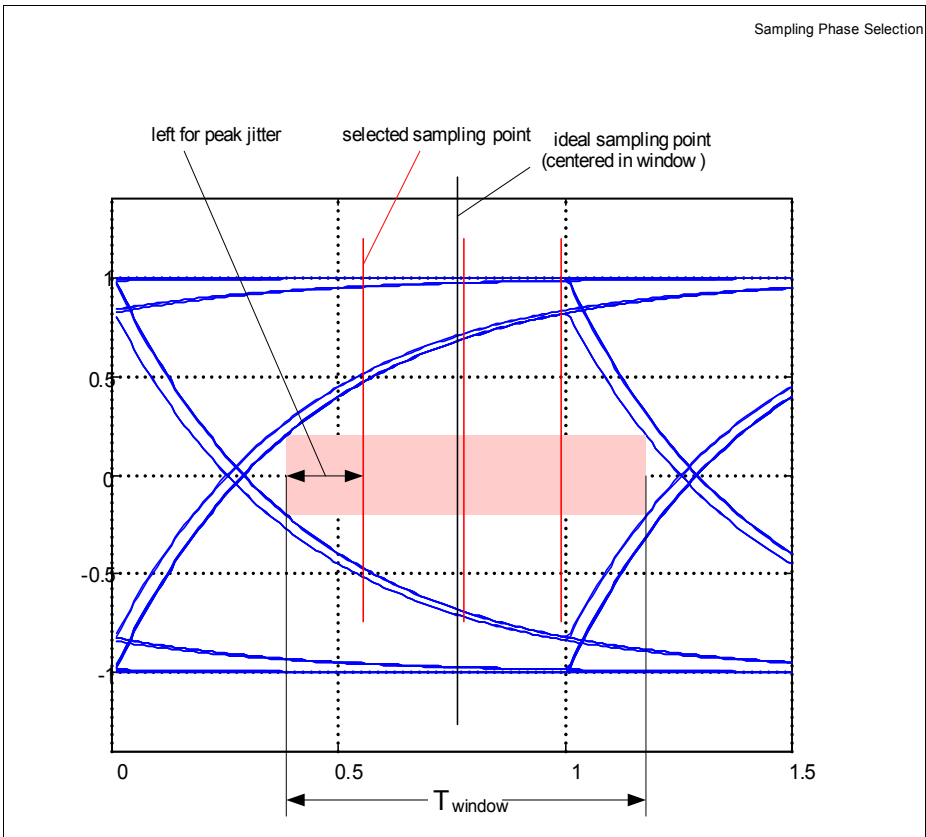


Figure 21-46 Sampling Point Selection

21.15.2.4 Protocol Layer

The protocol layer is between the higher layer protocol (HSSL) and the physical layer. The protocol Layer is performing the frame generation in transmit direction and extracts the payload out of a frame in receive direction, called deframing.

Deframer

The deframer's tasks are decoding the 8-bit header and separating the data stream according to the logical channel type.

During Microcontroller reset, the deframer shall ignore all data coming in from the correlator. This is a precaution measure to eliminate possible malfunctions due to floating HSCT lines.

Additionally the deframer does check the header for potential errors, e.g. size is matching received data or size is allowed and received type is allowed. An error in the header discards the received data.

Header Decoding

The first 8-bit in a frame, after the sync pattern has been filtered out by the correlator, is the header or frame type field. It contains information about the payload size and logical channel type coding (see [Figure 21-11](#)). Please note again that all transfers are most significant bit first.

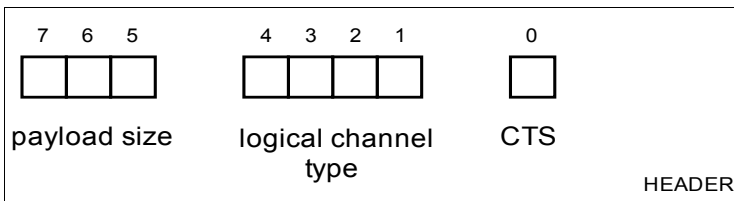


Figure 21-47 Header (Frame Type Field)

- The payload size coding can be seen in [Table 21-10](#) in the previous.
- Bits 1 to 4 define the logical channel type. See [Table 21-11](#) for details on the logical channel type coding. Note that some kinds of transfer are exclusively used from the master interface to the slave interface or from the slave interface to the master interface.
- Bit 0 is used as Clear To Send bit CTS for transfers from the slave interface to the master interface (Requires an enable via configuration). For transfers from the master interface to the slave interface also the master interface can send CTS, which must also be enabled via configuration before. Also CTS from master interface indicates the master to the slave, no more data can be accepted. This feature requires for the master interface as well as for the slave interface that higher layer FIFO provides a configuration watermark allowing to tune the CTS signaling - dependant on the transferred payload size.

Table 21-11 Logical Channel Type Coding

b4 ... b1	Logical Channel Type Master interface to Slave interface (TX Link)	Logical Channel Type Slave interface to Master interface (RX Link)
0000	Interface Control	Interface Control (PING answer, 32-bit value)
0001	Master interface Unsolicited Status (32-bit only)	Slave interface Unsolicited Status (32-bit only)
0010	Slave interface Control (not supported, receiver generates an error) ¹⁾	Slave interface Solicited Read (not supported, receiver error) ²⁾
0011	CTS Transfers	CTS Transfer
0100	Data Channel A	Data Channel A
0101	Data Channel B	Data Channel B
0110	Data Channel C	Data Channel C
0111	Data Channel D	Data Channel D
1000	Data Channel E (not supported, receiver generates an error interrupt)	Data Channel E (receiver error interrupt)
1001	Data Channel F (not supported, receiver error interrupt)	Data Channel F (receiver error interrupt)
1010	Data Channel G (not supported, receiver error interrupt)	Data Channel G (receiver error interrupt)
1011	Data Channel H (not supported, receiver generates an error interrupt)	Data Channel H (receiver error interrupt)
1100	Reserved - receiver error interrupt	Reserved - receiver error interrupt
1101	Reserved - receiver error interrupt	Reserved - receiver error interrupt
1110	Reserved - receiver error interrupt	Reserved - receiver error interrupt
1111	Reserved - receiver error interrupt	Reserved - receiver error interrupt

1) Solicited control command for slave interface control is not supported - instead a higher layer two-way protocol is available.

2) Solicited read command for slave interface read is not supported - instead a higher layer two-way protocol is available.

Note: A logical channel type TX transfer request of not supported or reserved types is not stopped or prevented to be send out. Receiver side is expected to react with an error indication.

*Note: The received header information can be checked by SW reading Header status register (**STAT**). An error in the header is a severe error in a running system. Application SW must handle the situation by interface link checks (PING). Additionally it is suggested to restart a running stream, if data loss cannot be tolerated. Received header frames are discarded, the same as the CTS information inside this error frame.*

Payload Size for transmission Data

For a transmission data transfer it is recommended to use max 288-bit payload size.

Logical Channel Separation

Reflecting the logical channel type to the higher level interface, the deframer sends the frame payload. The payload, as all other parts in a frame, is sent most significant bit first.

- All data received on and decoded as channel A to D data are sent via an 32-bit data interface to the RX-FIFO.
- Data from one HSCT interface to the other can be sent using the 32-bit, 64-bit, 96-bit or 288-bit payload size.

Note: Receiving a 8-bit header size on a logical channel results in an header error.

Framer

The framer combines data and control information from the higher layer protocol to the frame payload. Note that the payload again is sent most significant bit first. The framer codes the logical channel type and the payload size in the header according to the HSCT header format (see [Figure 21-47](#)). Then a complete frame consisting of 16-bit synchronization pattern (see [Figure 21-44](#)), 8-bit header and payload field is put together.

The following logical channels are used for transmissions between master interface and slave interface:

- Data from the transmit Unit: The frame will be sent to the receiving interface side as soon as the transmit FIFO contains enough data for the required payload size. The appropriate values will be signaled with the payload size signal and in the logical channel type field in the header. For data transfer, only the logical channel types Data Channel A to D are be used! Receiving a header type reflecting the type Data Channel E to H generates an error.
- Slave interface control (at TX link) and Slave interface read (at RX link), identified by logical channel type encoding of Hex 2, is not supported.
- Unsolicited status information can be send by master interface and slave interface based on a 32-bit payload size. An unsolicited status can be used to send an unexpected and urgent message to the other side. Receiving an unsolicited message results in an interrupt at the receiving side.

Note: The Slave interface (RX link) can send out a control command (channel type encoding Hex 0), which is the predefined PING answer having a defined 32-bit payload. All other control commands are not available for the RX link to send. Slave interface solicited read (RX link, channel type encoding Hex 2) and Master interface control (TX link, channel type encoding Hex 2) are not supported by the HSCT automotive module. Receiving a header type reflecting these types on the RX link or TX link generates an error.

The handling of all frame parts is done by the framer. For determining the end of a frame, the framer shall count the number of payload bits sent to the receiver side. In between frames bits 0 shall be transmitted.

Clear to Send (CTS)

The HSCT interface is able to use “data pull” in both directions (master interface to slave interface and slave interface to master interface) to perform flow control on the transmission FIFO's. Therefore the CTS bit in the header is asserted, when the other side is allowed to send more data, and it is negated when the other side should stop transferring data.

A CTS deactivation can be send out in a “normal” transmit frame, indicated by the header bit. The following payload is not discarded at receiving interface side and processed normally. CTS deactivation can be send out in a CTS type transmit frame, if no actual transmit data is available. The clear to send information is indicated by the header CTS bit deactivation. Following payload is discarded at receiving side. A received clear to send deactivation requires to stop the transmission immediately after the actual frame being in transmission or prevent to send a new frame, if currently no transmission is active. Control frames can still be send.

The clear to send is independent of the channel having caused the back pressure. It simply stops the data channel transfer. As soon as the traffic jam is resolved the CTS bit in the next transmitted frame is activated again, or a separate CTS frame is send with the CTS bit activated.

Higher Layer software must analyze which channel caused the traffic jam and can send the information to the other side.

The usage of the CTS signaling is optional. CTS signaling can be disabled by disabling the CTS_FRAME **CTSCTRL**.

Per default the CTS bit in the header is set to 1. When the threshold value in the receive FIFO is reached, the CTS bit shall be reset to 0 in the header of the next frame sent. If there are currently no data in the higher layer module transmission FIFO to be sent in a new frame, a dedicated CTS transfer according to the logical channel type in **Table 21-11** is generated. This transfer always uses the 8-bit payload size containing dummy bits. The payload in a CTS transfer will be discarded by the receiving side.

Generation of dedicated CTS frames can be disabled by resetting bit **CTS_FRAME** in register **CTSCTRL** to 0.

The CTS bit is set to 1 again when the receive FIFO fill level in the higher layer module is below the threshold value. This is reflected by activating the CTS signal again.

The CTS signals are full duplex signals and are side information in the transferred data frame. The deframer filters the TXCTS information from the received frame header. The TXCTS is presented to the higher layer TX interface frame based. Changes within a frame is not intended. At the higher layer receiver side the higher layer does activate the RXCTS signal, if no more data can be received. The Framers sends the RXCTS information with the header of the next frame.

Sleep Mode for line driver path

The HSCT interface offers an optional transmission sleep mode in between frames. Whether the move to the transmission sleep mode can be performed, depends on the actual symbol rate, as well as the protocol payload size, because it takes the LVDS line driver about 60 ns to wake up from sleep mode. In sleep mode, the transmitting line suggests sleep mode to the other side by sending an 1 immediately after the last bit in a frame. Setting the SLPEN bitfield to 1 enables Transmission sleep mode.

If enabled, Transmission sleep mode shall be triggered after every frame, in case there is no data or CTS frame due to be sent. When Transmission sleep mode shall be triggered to the opposite receiving side, the P/S converter inserts a 1 after the last bit in a frame, and asserts the sleep signal to the transmission LVDS driver.

The sleep signal is reset as soon as the transmission FIFO wake-up threshold value is reached, or when control information shall be sent to the slave interface from the master interface. These events trigger the wake-up counter which ensures that the LVDS line driver becomes ready for operation within 60 ns. After the wake-up time the framer starts sending eight bits 0 for signalling the start of a new frame to the receiving side. Subsequently the framer generates the new frame to be sent to the receiving side.

The transmission FIFO threshold value at the higher level protocol module can be programmed depending on the payload size and data rate in order to ensure that no additional latency in the data stream is generated due to the wake-up time for the LVDS line driver.

Interface Control

The master interface is configured via a CPU based configuration. The Slave interface can be configured by sending an interface control command from the master interface. The interface control command is used to configure the slave interface into the defined mode. The receiving slave interface shall take no action, if a reserved payload value is received. Instead it generates an error interrupt. HW reacts on the received control command.

After startup the interface is enabled to work at low speed (**SysClk/4** or **SysClk/2**) and the PLL is off, no matter of the mode it had been before. Therefore shutting down the interface by chip internal control mechanism causes the interface to leave loopback-mode, clock test mode and sleep mode, if it was previously in any of these modes, and forces the receiver and transmitter to power down mode.

The interface transmitter path is disabled per default. It shall be enabled by using the appropriate interface control, which is a payload command for the slave interface and a control via SW for the master interface.

Note: No data can be transferred to the other side as long as the transmitter is disabled. In this case all data in TX direction are discarded!

The Interface Control Logical Channel transfer command always uses the 8-bit payload size and is send by the master interface only. Payload is send MSb first. The interface control payload values are shown in **Table 21-12**. All other payload values are reserved for future use. Whenever a reserved payload value is received it shall be ignored and reported by an error.

Table 21-12 Interface Control Payload Values

Value (hex)	Function
00 _H	“ping” (send by master interface. Slave interface sends back a fixed 32-bit payload result.)
01 _H	Reserved
02 _H	Slave interface clock multiplier start (in preparation for high speed mode)
04 _H	Slave interface clock multiplier stop (after fallback from high speed mode)
08 _H	Select low speed mode for transfers from the Master interface to the Slave interface
10 _H	Select high speed mode for transfers from the Master interface to the Slave interface
20 _H	Select low speed mode for transfers from the Slave interface to the Master interface
40 _H	Select medium speed mode for transfers from the Slave interface to the master interface <i>Note:</i> Not allowed, if SysClk = Crystal/2 (10 MHz) at Slave interface.
80 _H	Select high speed mode for transfers from the Slave interface to the master interface
31 _H	Enable Slave interface transmitter
32 _H	Disable Slave interface transmitter

Table 21-12 Interface Control Payload Values (cont'd)

Value (hex)	Function
34 _H	Turn on clock test mode (Send 101010... on Rx line continuously using currently configured Rx line rate; cancelled by issuing the “test mode off” command. Internal transmit and receive path is disabled. Received data is discarded. The only exception to this is the “test mode off” function, which in should not loop back and re-initialize itself for normal operation, retaining whichever interface clock rate are currently configured.)
38 _H	Turn off test mode (cancel clock test mode and payload loopback)
FF _H	Turn on payload loopback (incoming frames at Slave interface are looped back until cancelled by a Frame containing “test mode off”.) <i>Note: Requires same speed configuration for RX- and TX-link.</i>

Note: Before changing the speed mode SW has to take care that data Transmission on both links is stopped and the interface remains quiet.

As a general principle, after a mode change the master interface should request a data transfer from the slave interface to stimulate a frame, which confirms the correct reception and the mode change was successful. The PING function, available in the interface control logical channel shall be used to verify a successful speed change. The master interface generated PING has to be send in system defined time intervals, to verify the link is still alive. Sending out a ping requires system SW to activate a system timer. Not receiving the PING answer within the system defined time, the link cannot be considered as stable and a new initialization is required. The Slave interface sends back a fixed, known response based on a control command, at the currently configured link rate on RxData line.

Slow to Fast:

The slow to fast interface speed change can apply to only one of the interfaces data transfer directions or to both at the same time. At both sides Clock multipliers are needed, when either interface is to run at high speed. If the RXData link is not enabled, but needs to be, the master interface sends “Enable Slave Interface transmitter”. If the slave interface clock multiplier is not already running the master interface sends “Slave interface clock multiplier start” to the slave interface at low speed. The master interface enables its own clock multiplier, if it is not already running. After the proper settling time, the master interface sends “select high speed” to the slave interface at the currently configured speed for the TXData link. The slave interface switches the requested interface to high speed. The master interface switches the requested interface to high speed. The master interface sends a frame to the slave interface requiring a response at the currently configured speed for the TxData Link. The slave interface responds with a frame to confirm the mode change has worked. In order to select high speed for both

RXData and TXData links the master interface sends “select high speed” for the RxData link first and then for the TxData link. As described before it is suggested to send the PING for verifying a successful speed change.

Fast to Slow:

The fast to slow interface speed change is the converse of the slow-to-fast transition. Both clock multipliers must be running for either interface to operate in high-speed mode. Once both interfaces no longer require the clock multipliers they can be turned off, using the appropriate interface control payload values to command the switch in the slave interface.

Before entering high speed mode it shall be assured that the high speed clock is available by sending the interface control value 0x02.

Figure 21-48 shows the TX transmission state machine, which controls transfers from the master interface to slave interface. The interface control value `int_ctrl` represents the received interface control payload value according to **Table 21-12**.

Note: Application Software has to ensure, that no data communication is active on RX-link or TX-link during the time the interface speed is changed. Otherwise a save speed change without data loss can not be guaranteed interface.

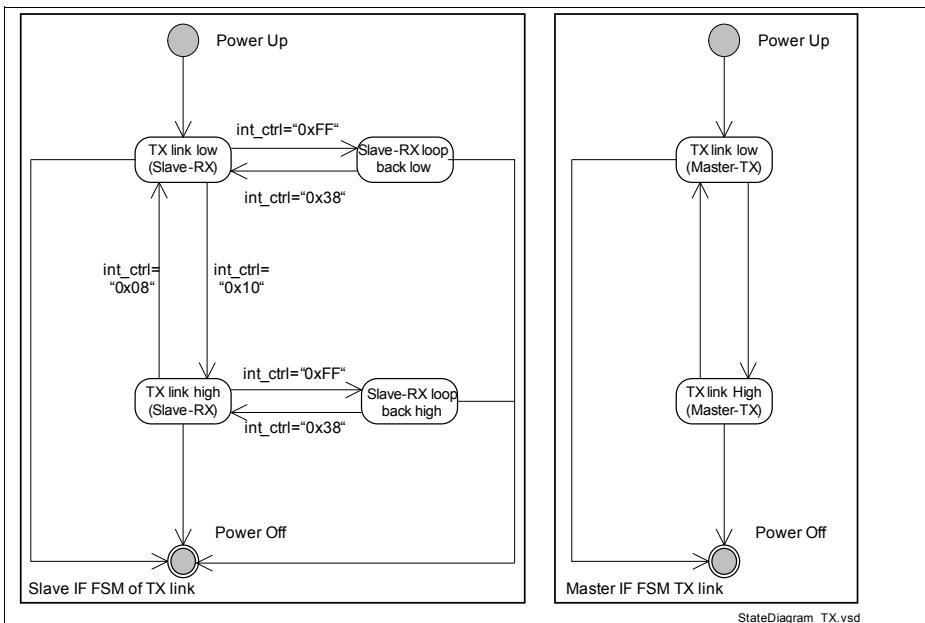


Figure 21-48 TX link State Machine

The loopback mode is used for test and verification to identify BER. When the master interface demands loopback mode, the slave interface internally loops back all received payload values in the following frames using the currently configured clock speed on the receive data link, without applying further processing internally. (Transmit and receive chains internally shall be disabled.) The header configuration is the same as in the incoming frame and the CTS bit in the header shall be asserted. Payload contents are ignored at slave interface in loopback mode, except for the “turn off test mode” interface control value, which ends loopback mode. The “turn off test mode” frame shall not be looped back to the master interface. After leaving loopback mode, the interface is prepared for normal operation at the previously used clock speed.

Note: The “loopback on” function has been intentionally coded at a large Hamming distance from the other codes (minimum of 5, usually 7, thus requiring at least five bit-errors in the Header field for another function to be corrupted into “loopback on”.

The RX link state machine, which is responsible for controlling transfers from slave interface to the master interface, can be seen in [Figure 21-49](#). The state machine is implemented on slave interface at transmitting path. The interface control value `int_ctrl` represents the received interface control payload value according to [Table 21-12](#).

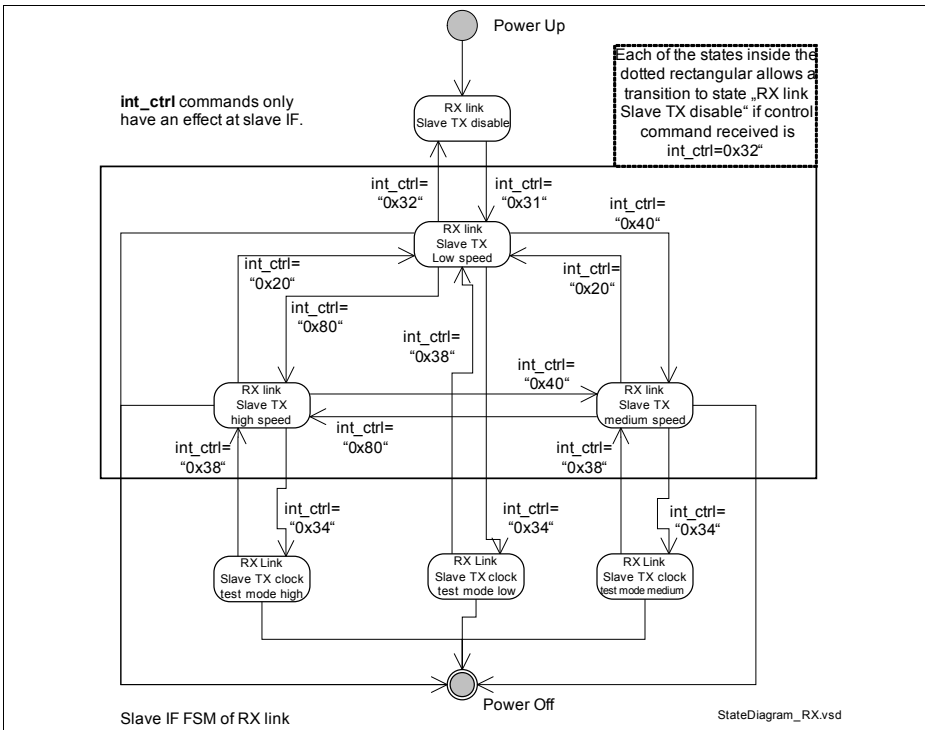


Figure 21-49 RX link State Machine

*Note: The **Medium speed mode** is only available, if the reference clock (SysClk) is running at **Crystal speed (20 MHz)**. For Crystal/2 medium speed mode is not available. A configuration of medium speed at Crystal/2 = SysClk causes low speed mode.*

In clock test mode the transmitter sends '101010 ...' continuously using the currently configured clock speed. Clock test mode can be left by sending the interface control value "turn off test mode".

The ping function in the slave interface always uses the 32-bit payload size sending back the value 0xABCDEF01 (see [Figure 21-50](#)) using the logical channel type Hex 0 (PING Status - on RX link).

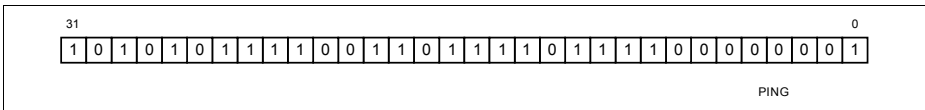


Figure 21-50 The Slave interface “ping” Response

The Interface control commands to configure the slave interface have to be written into the master interface register **IFCTRL.IFCVS**. For sending the control frame a logic 1 needs to be written to the **IFCTRL.SIFCV** register to trigger the frame transfer. The 8 register bits **IFCTRL.IFCVS** represent the same functionality as defined for the interface control payload values. Coding is as shown in **Table 21-12**. The register field **IFCTRL.IFCVS** shall be written with the appropriate interface control payload value for every state transition shown in **Figure 21-48** and **Figure 21-49**. The interface control in the slave interface can alternatively be performed by programming the **IFCTRL.IFCVS**. This allows controllability of the Slave interface besides the frame based configuration. The Master interface is controlled by register configuration of physical layer only.

In **Table 21-13** the processing times for the time consuming interface control transitions (defined in **Table 21-12**) are listed.

Table 21-13 Processing Times for Interface Control

IFCVS (hex)	Transition Description	Processing Time
02 _H	Slave interface clock multiplier off to Slave interface high speed clock avail.	80 µs
04 _H	Slave interface clock multiplier on to Slave interface clock multiplier off	2 µs
08 _H	TX high speed mode to TX low speed mode	600 ns
10 _H	TX low speed mode to TX high speed mode	400 ns
20 _H	RX high or medium speed to RX low speed mode	400 ns
40 _H	RX high or low speed to RX medium speed mode	400 ns
80 _H	RX low or medium speed to RX high speed mode	400 ns
31 _H	RX disabled to RX low speed mode	500 ns

Note: The physical layer requires about 20 µs to power up, including bandgap. Afterwards the PLL Reset is allowed to be release. After PLL reset removal it takes about 50 µs to get the PLL into a locked state.

Power up sequence

The power up sequence of the HSCT Physical Layer and the LVDS IO requires to have the BandGap enabled and stable before the Bias distributor and LVDS power can be enabled. Bias distributor and LVDS power on are enabled in Master interface mode by disabling the TX_PD at LVDS PAD Control register. The required time to power up is about 500 ns. In Slave interface mode the TX_PD at LVDS PAD Control register needs to be enabled by startup software during initialization process.

21.15.3 Use Cases

The following section describes the chip to chip communication requirements based on a HSCT interface connection. The Micro Controller (MC) is from a system point of view intended to be the master interface of the chip to chip communication. Application SW has to consider power up time of companion chip during the bringup of the link.

21.15.3.1 MC to ASIC

The MC (master) is connected to the ASIC (slave) which allows to perform data streaming in both directions and register control (read/write) by the MC, which allows to use ASIC defined ASIC functions.

21.15.3.2 MC to FPGA

The MC (master) is connected to the FPGA (slave), which allows to perform data streaming in both directions.

21.15.3.3 MC to MC

This use case allows to connect two MC's together, whereas one of them gets, from the Interface point of view, the master function and the other the slave function. Potential application would be register control or data streaming. After the interface role (master or slave) definition, the role is not allowed to change. In this case only the master interface does have the crystal connected. The slave interface does use the SysClk as reference clock for the existing PLL's. The reference clock at the slave interface is used to clock the HSCT PLL and additionally as clock source for the system PLL.

The HSCT analog part is per default in power down mode. Application SW is required to enable power supply for the analog part. Only after the power is on for the analog part, communication via the HSCT interface is possible.

21.15.4 HSCT Register Description

21.15.4.1 Registers Definition

This section describes the kernel registers of the HSCT module and the System related register in single kernel configuration. All HSCT kernel register will be referenced in other parts of the TC26x B-Step User's Manual by the module name prefix "HSCT_".

HSCT Kernel Register Overview

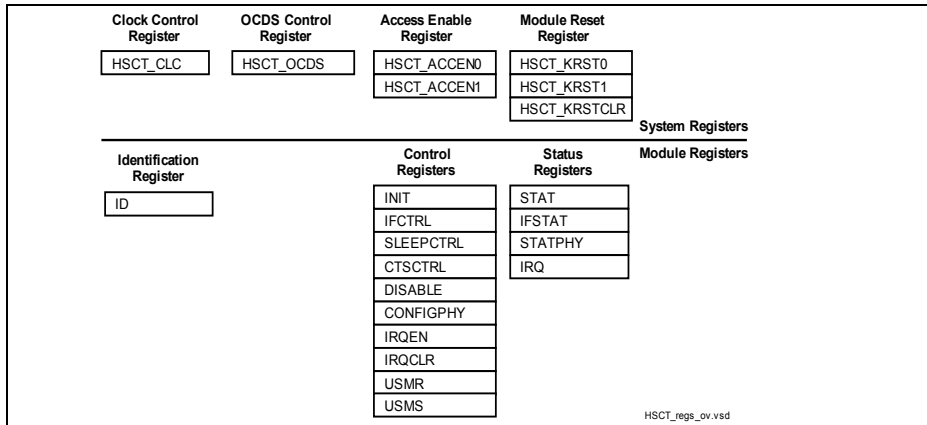


Figure 21-51 HSCT Register Overview

The complete and detailed address map of the HSCT modules is described in the next chapter.

Address Map
Table 21-14 Registers Address Space

Module	Base Address	End Address	Note
HSCT	F0090000 _H	F009FFFF _H	-

Table 21-15 Registers Overview

Short Name	Description	Offset Addr.	Access Mode		Reset Class	Description See
			Read	Write		
CLC	Clock Control Register	00 _H	U, SV	SV,E,P	3	Page 21-15 8
ID	Identification Register	08 _H	U, SV	BE	3	Page 21-13 2
INIT	Initialization Register	10 _H	U, SV	U,SV,P	3	Page 21-13 3
IFCTRL	Interface Control Register	14 _H	U, SV	U,SV,P	3	Page 21-13 5
SLEEPCTRL	Sleep Control Register	18 _H	U, SV	U,SV,P	3	Page 21-13 8
CTSCTRL	Clear To Send Register	1C _H	U, SV	U,SV,P	3	Page 21-13 9
DISABLE	Transmission Disable Register	20 _H	U, SV	U,SV,P	3	Page 21-14 1
STAT	Status Register	24 _H	U, SV	U,SV,P	3	Page 21-14 3
IFSTAT	Interface Status Register	28 _H	U, SV	U,SV,P	3	Page 21-14 5
CONFIGPHY	Physical Layer Configuration Register	30 _H	U, SV	U,SV,P	3	Page 21-14 6
STATPHY	Physical Layer Status Register	34 _H	U, SV	U,SV,P	3	Page 21-14 8
IRQ	Interrupt Registers	40 _H	U, SV	U,SV,P	3	Page 21-14 9
IRQEN	Interrupt Enable Registers	44 _H	U, SV	U,SV,P	3	Page 21-15 3

Table 21-15 Registers Overview

Short Name	Description	Offset Addr.	Access Mode		Reset Class	Description See
			Read	Write		
IRQCLR	Clear Interrupt Registers	48 _H	U, SV	U,SV,P	3	Page 21-15 5
USMR	Unsolicited Message Received Registers	50 _H	U, SV	U,SV,P	3	Page 21-15 6
USMS	Unsolicited Message Send Registers	54 _H	U, SV	U,SV,P	3	Page 21-15 7
OCS	OCDS Control and Status Register	FFE8 _H	U, SV	SV,P	1	Page 21-15 9
KRSTCLR	Reset Status Clear Register	FFEC _H	U, SV	SV, E,P	3	Page 21-16 5
KRST1	Reset Control Register 1	FFF0 _H	U, SV	SV, E,P	3	Page 21-16 4
KRST0	Reset Control Register 0	FFF4 _H	U, SV	SV, E,P	3	Page 21-16 3
ACCEN1	Access Enable Register 1	FFF8 _H	U, SV	SV, SE	3	Page 21-16 2
ACCEN0	Access Enable Register 0	FFFC _H	U, SV	SV, SE	3	Page 21-16 1

Note: The absolute register address is calculated as follows: Module Base Address ([Table 21-14](#)) + Offset Address

A register is addressed word wise.

Note: Register bits marked reserved in the following register description are virtual registers and do not contain flip-flops. They are always read as 0.

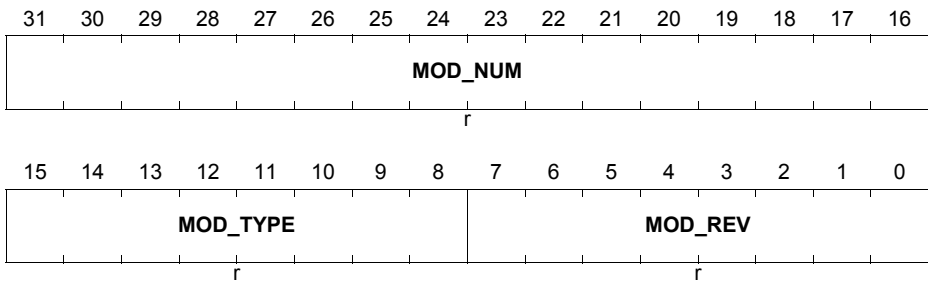
HSCT Kernel Register Definitions

ID

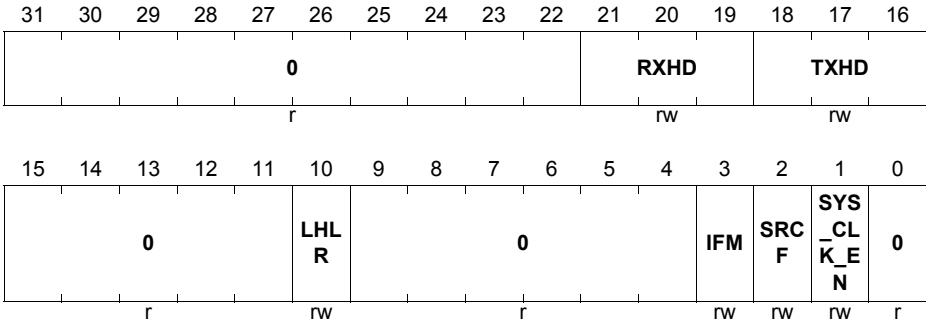
The Module Identification Register ID contains read-only information about the module version. For a register overview, see the [Table 21-15](#).

ID

Module Identification Register (08_H) **Reset Value: 00B4 C001_H**



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number 01 _H The value of a module revision starts with 01 _H (first revision).
MOD_TYPE	[15:8]	r	Module Number Type C0 _H 32 bit peripheral
MOD_NUM	[31:16]	r	Module Number for module identification 00B4 _H Is the module identification number for the HSCT interface.

INIT
INIT
Initialization register (10_H) Reset Value: 0000 0000_H


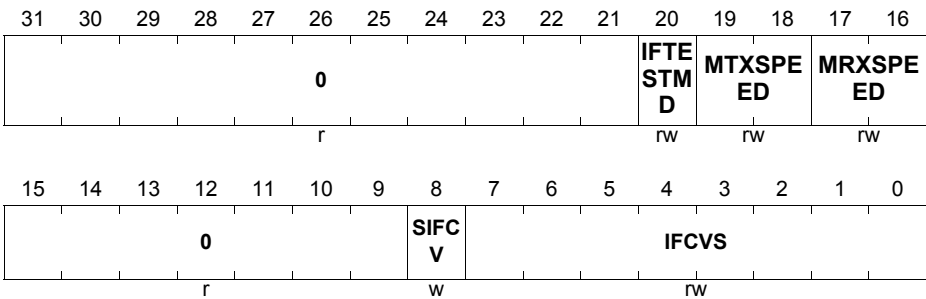
Field	Bits	Type	Description
SYS_CLK_EN	1	rw	Enable SysClk in Master interface SysClk enable activates the SysClk. 0 _B disable (default) 1 _B enabled <i>Note: This feature is only available in the master interface. In slave interface mode the register setting does not have an effect.</i>
SRCF	2	rw	Select SysClk / Reference Clock Frequency rate For master interface defines Reference frequency. For slave interface frequency defines SysClk frequency. Has an effect for Low Speed only. 0 _B SysClk; RefClk is 20 MHz (Divider 1/1) 1 _B SysClk; RefClk is 10 MHz (Divider 1/2)
IFM	3	rw	Select Interface Mode Select the Interface Mode (master IF or slave IF). 0 _B master IF 1 _B slave IF

Field	Bits	Type	Description
LHLR	10	rw	<p>Loopback path at Slave interface side at higher layer system RAM.</p> <p>Disable of the LVDS IO loopback when the bit is set. The function allows to loop back data at a higher layer in HW or by SW using a RAM area.</p> <p>0_B Loopback path performed at LVDS IO 1_B Loopback path at higher layer RAM area</p>
TXHD	[18:16]	rw	<p>Transmit High Speed Divider.</p> <p>The Transmit High Speed data rate can be reduced by dividing factors. The Transmit High Speed data rate is separated from the Receive High Speed data rate.</p> <p>000_B Divider 1/1 001_B Divider 1/2 010_B Divider 1/4 011_B Divider 1/8 100_B Divider 1/16 101_B For future use. 110_B For future use. 111_B For future use.</p>
RXHD	[21:19]	rw	<p>Receive High Speed Divider.</p> <p>The Receive High Speed data rate can be reduced by dividing factors. The Receive High Speed data rate is separated from the Transmit High Speed data rate.</p> <p>000_B Divider 1/1 001_B Divider 1/2 010_B Divider 1/4 011_B Divider 1/8 100_B Divider 1/16 101_B For future use. 110_B For future use. 111_B For future use.</p> <p><i>Note: For future use configuration leads to no output clock delivered to the correlator.</i></p>
0	[31:22], [15:11], [9:4], 0	r	Reserved

IFCTRL

This register allows to configure the transfer Speed of the Master Interface and is also used to send the command frame to the slave.

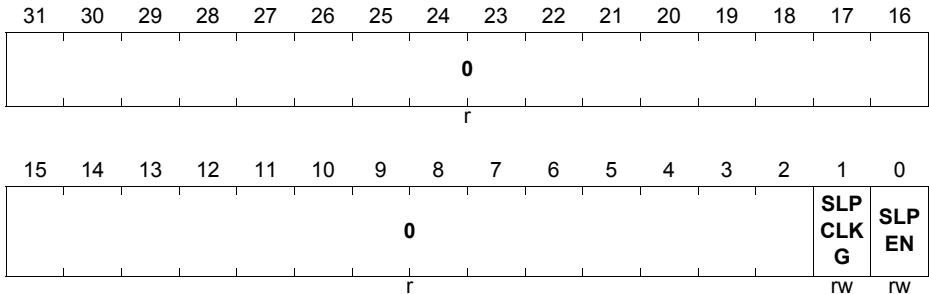
Note: The Master interface link speed configuration immediately takes place. Therefore the Master interface speed transition must not be changed with a Slave interface command send activity.

IFCTRL
CPU transfer control register
(14_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
IFCVS	[7:0]	rw	Master Mode - Interface Control Value to be send to Slave interface See Table 21-12 . <i>Note:</i> 7. <i>Master IF: The value is taken as control frame value send as payload to the slave IF.</i> 8. <i>Slave IF: The value is a new configuration of the slave IF.</i>

Field	Bits	Type	Description
SIFCV	8	w	<p>Master Mode - Slave IF control frame trigger</p> <p>0_B The IFCVS field configured value does not have an effect (default).</p> <p>1_B Writing a one to the register bit sends the control frame configured in register field IFCVS, if the interface is configured as master. In slave interface configuration the trigger has an effect and takes the IFCTRL.IFCVS value. In slave mode this is not the usual control flow. A frame based configuration shall be used instead. A control frame has higher priority than a register control. At a simultaneous occurrence of both configuration source the register configuration in slave mode is lost.</p> <p><i>Note: Changing the interface configuration, software must guarantee not having transfers active on the interface.</i></p> <p><i>Note: The register bit reads back a zero always.</i></p>
MRXSPEED	[17:16]	rw	<p>Master Mode RX speed</p> <p>00_B Low Speed</p> <p>01_B Medium Speed</p> <p>10_B High Speed</p> <p>11_B for future use</p> <p><i>Note: Register setting only valid in interface master mode.</i></p>
MTXSPEED	[19:18]	rw	<p>Master Mode TX speed</p> <p>00_B Low Speed</p> <p>01_B for future use</p> <p>10_B High Speed</p> <p>11_B for future use</p> <p><i>Note: Register setting only valid in interface master mode.</i></p>

Field	Bits	Type	Description
IFTESTMD	20	rw	<p>Master Mode Interface Test Mode</p> <p>0_B Test Mode disabled</p> <p>1_B Test Mode enabled - send out 101010101.. test pattern continuously.</p> <p><i>Note: Register setting only valid in interface master mode. Slave interface does send out the 101010101.. sequence on RX link by receiving the Turn clock test mode on interface Control logical channel command.</i></p>
0	[15:9],[31:21]	r	Reserved

SLEEPCTRL
SLEEPCTRL
Sleep Control Register
(18_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SLPEN	0	rw	Sleep mode enabled Sleep mode is enabled and performed after receiving a 1 at the end of a received frame or in transmission direction, if a empty is received from HSSL FIFO. 0 _B Sleep mode disabled. 1 _B Sleep mode enabled.
SLPCLKG	1	rw	Clock Gating in Sleep Mode During transmission in sleep mode the clock for HSCT (framer, deframer) can be gated in order to minimize power consumption. <i>Note: Clock gating: Receiving path and transmitting path is separated.</i> 0 _B Clock gating in transmission sleep mode disabled. 1 _B Clock gating in transmission sleep mode enabled.
0	[31:2]	r	Reserved

CTSCTRL
CTSCTRL
Clear To Send Control Register
(1C_H)
Reset Value: 0000 0001_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												HSS L_C TS_F BD	CTS _RX D	CTS _TX D	CTS _FR AME
r												rw	rw	rw	rw

Field	Bits	Type	Description
CTS_FRAME	0	rw	Transmit CTS Frame Generation Dedicated CTS frames are generated after the receive FIFO threshold value is reached, indicated by the CTS header bit in case there is currently no data to be transferred. 0 _B Generation of dedicated CTS frames disabled. 1 _B Generation of dedicated CTS frames enabled.
CTS_TXD	1	rw	Disable TX CTS signaling If this bit is set to 1, CTS signaling is not performed at the interface and the status remains at the clear to send for every frame send. 0 _B Enable CTS signaling (default). 1 _B Disable CTS signaling.
CTS_RXD	2	rw	Disable RX CTS detection If this bit is set to 1, CTS detection is not performed at the receiver and the status remains internally at clear to send for every frame received. 0 _B Enable CTS detection (default). 1 _B Disable CTS detection.

Field	Bits	Type	Description
HSSL_CTS_FBD	3	rw	Disable HSSL interface CTS Frame Blocking If this bit is set to 1, CTS signaling is not performed at the interface and the status remains at the clear to send for every frame send. 0 _B Enable CTS frame blocking (default). 1 _B Disable CTS frame blocking.
0	[31:4]	r	Reserved

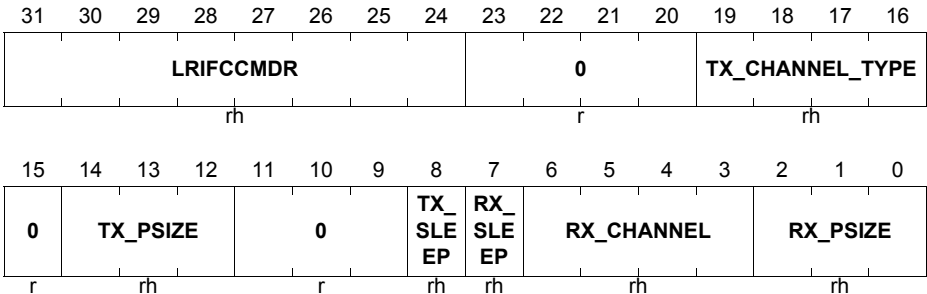
DISABLE
DISABLE
Transmission Disable Register (20_H) **Reset Value: 0000 0003_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													RX_	RX_	TX_
													HEP	DIS	DIS
													D	D	D
r													rw	rw	rw

Field	Bits	Type	Description
TX_DIS	0	rw	Disable HSCT Transmit path in Master interface Disable the transmit path of the HSCT interface. If this bit is set to 1 - no transfer can be initiated and the LVDS driver is disabled. 0 _B enable 1 _B disable
RX_DIS	1	rw	Disable HSCT Receive path in Master interface Disable the receive line path of the HSCT interface. If this bit is set to 1 - no transfer from the other side can be received and the Master RX path is in a low power state. 0 _B enable 1 _B disable <i>Note: This feature is only available in the master interface. Slave interface RX path can not be disabled and a write to the register has no effect.</i>
RX_HEPD	2	rw	Disable RX Header Error Discard Payload data. Instead of discarding the Payload data at a header error the payload data is passed to the higher layer (HSSL). Only channel data to HSSL is affected. 0 _B Header error received data is discarded 1 _B Header error received data is passed to the higher hardware layer.

Field	Bits	Type	Description
0	[31:3]	r	Reserved

STAT
STAT
Status Register
(24_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RX_PSIZE	[2:0]	rh	RX (receiving) Payload Size Contains the payload size of the last received frame.
RX_CHANNEL	[6:3]	rh	RX (receiving) Logical Channel Type Contains the logical channel type of the last received frame. See Logical Channel Type Coding .
RX_SLEEP	7	rh	RX (receiving) Sleep Mode Status 0 _B HSCT is in receive direction active 1 _B HSCT is in receive direction in sleep mode
TX_SLEEP	8	rh	TX (transmitting) Sleep Mode Status 0 _B HSCT is in transmit direction active 1 _B HSCT is in transmit direction in sleep mode
TX_PSIZE	[14:12]	rh	Transmission Payload Size Coding of the logical channel type is according to the Table: Payload Size Coding . This value was used in the logical channel type field in the header for the actual transfer in transmit direction.
TX_CHANNEL_TYPE	[19:16]	rh	Transmission Logical Channel Type Coding of the logical channel type is according to the Table: Logical Channel Type Coding . This value was used in the logical channel type field in the header for the actual transfer in transmit direction.

Field	Bits	Type	Description
LIFCCMDR	[31:24]	rh	Last Interface Control Command Received The bit value reflects the last control command received. The bit is active in Slave interface mode only. In master mode it reflects logic 0 always. (See Table 21-12)
0	[23:20], 15,[11: 9]	r	Reserved

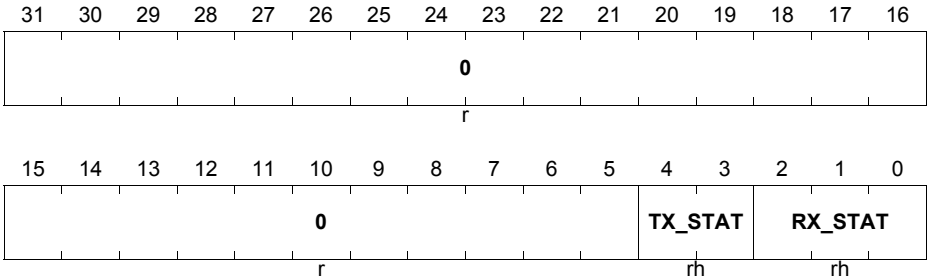
IFSTAT

IFSTAT

Interface Status Register

(28_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RX_STAT	[2:0]	rh	<p>HSCT slave interface Status for RX link</p> <p>000_B Interface is disabled in RX link direction. 001_B Interface runs at low speed on RX link. 010_B Interface runs at medium speed on RX link. 011_B Interface runs at high speed on RX link direction. 100_B Clock test mode and RX link disabled 101_B Clock test mode and low speed on RX link 110_B Clock test mode and medium speed on RX link 111_B Clock test mode and high speed on RX link.</p> <p><i>Note: slave interface transmitter only</i></p>
TX_STAT	[4:3]	rh	<p>HSCT slave interface Status for TX link</p> <p>00_B Interface runs at low speed on TX link. 01_B Interface runs at high speed on TX link. 10_B Loopback mode low speed. 11_B Loopback mode high speed.</p> <p><i>Note: slave interface receiver only</i></p>
0	[31:5]	r	Reserved

CONFIGPHY
CONFIGPHY

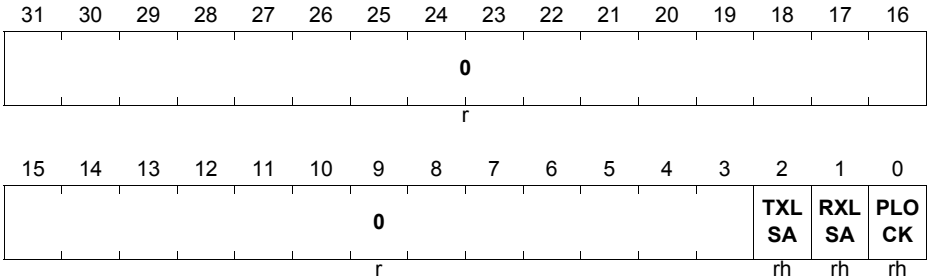
 Configuration physical layer register (30_H)

 Reset Value: 0000 80FC_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		OSC CLK EN		0	PLLIVR			PLLKI			PLLKP				
r		rw		r	rw			rw			rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PHY RST	PLL PKI	PLLWMF					PLLPE					PLL PON	0		
rw	rw	rw					rw					rw	r		

Field	Bits	Type	Description
PLLON	1	rw	PLL Power On (Master Mode only) Power on the PLL. Additionally the PLL logic receives an reset going from power of in power on state. 0 _B PLL power off. (default) 1 _B PLL power on. <i>Note: A deactivation during an active High Speed Mode has no effect.</i>
PLLPE	[7:2]	rw	PLL phase enable - allows to enable/disable each of the 6 Phase outputs.
PLLWMF	[13:8]	rw	PLL frequency control word multiplication factor Output frequency of PLL can be configured following the formula: $f_{pll} = f_{ref} * PLLWMF$ 00000 _B word multiplication factor disabled other , word multiplication factor enabled using the configured value

Field	Bits	Type	Description
PLLKPKI	14	rw	KP/KI Setting Enable of KP/KI setting by register programming 0 _B use default alpha, beta 1 _B use pll_alpha_i and pll_beta_i values supplied at macro pins
PHYRST	15	rw	Physical Layer Reset Reset of Physical layer digital logic. 0 _B Reset not enabled 1 _B Reset active (default) <i>Note: For SysClk generation and interface control deactivate this bit.</i> <i>Note: A new Physical Layer Reset (changing the PHYRST bit value from 0 to 1) shall only be done by a complete HSCT module kernel reset using a class 3 reset or the kernel reset register KSRT0 and KRST1 to avoid communication issues.</i>
PLLKP	[18:16]	rw	KP of PLL - Configuration of PLL beta coefficients of proportional part of loop filter
PLLKI	[21:19]	rw	KI of PLL - Configuration of PLL alpha coefficients of integral part of loop filter
PLLIVR	[25:22]	rw	Adjustment for integrated voltage regulator
OSCCLKEN	28	rw	Enable Oscillator Clock as PLL reference clock 0 _B PLL reference clock is HSCT SysClk_i 1 _B PLL reference clock is Oscillator input <i>Note: Reference clock path needs to be configured correctly, if the interface is configured as Master or Slave.</i>
0	31,30, 29, 27, 26, 0	r	Reserved

STATPHY
STATPHY
STATPHY
(34_H)
Reset Value: 0000 0006_H


Field	Bits	Type	Description
PLOCK	0	rh	PLL locked 0 _B PLL out of lock (default) 1 _B PLL locked
RXLSA	1	rh	Receiver in Low speed 0 _B Receiver in High speed 1 _B Receiver in Low speed
TXLSA	2	rh	Transmitter in Low speed 0 _B Transmitter in High speed 1 _B Transmitter in Low speed
0	[31:3]	r	Reserved

IRQ

Interrupt status register. Read only and updated by HW. The bit's remain active until cleared. The clear pulse is coming from the IRQCLR register. Only an interrupt source not being already active in the interrupt status register does generate an interrupt pulse to the SoC interrupt controller.

IRQ
Interrupt register
(40_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SFU	SFO	TXTE	PAR	USM	PLER	USMSF	SMER	IFCSF	CER	PYER	HER	0	
r		rh	rh	rh	rh	rh	rh	r	rh	r	rh	rh	rh	rh	r

Field	Bits	Type	Description
HER	1	rh	Header error detected Not supported size: <ul style="list-style-type: none"> Received command at slave interface 8-bit size only - other command sizes generate an error. Received command ping answer at master interface 32-bit size only - other command sizes generate an error unsolicited data 32-bit only logical data channel size 8-bit Not supported logical channel type Logical channel type not supported: <ul style="list-style-type: none"> 0b1xxx 0010 (Slave interface control and Slave interface read) 0 _B Header OK 1 _B Header issue detected.

Field	Bits	Type	Description
PYER	2	rh	Payload error detected Payload does not fit the header size 0 _B Payload size OK 1 _B Received payload size wrong
CER	3	rh	HSCT command error Control command not valid or control payload size bigger than 8-bit. 0 _B No command error 1 _B HSCT command error
IFCFS	4	rh	HSCT interface control frame send The scheduled interface control command is send. 0 _B No interface control command send 1 _B Interface control command send.
SMER	5	rh	Speed Mode Switch Error (Master Mode only) Speed mode change did not work. Received PING payload not valid. 0 _B Interfaces runs at defined speed 1 _B Ping payload error
USMSF	6	rh	Unsolicited message frame send finished Interrupt is indicated after the unsolicited message send is finished. 0 _B No unsolicited message send. 1 _B Unsolicited message send finished.
PLER	7	rh	PLL lost lock error After the PLL locked, the PLL may loose lock, which is reflected by the error 0 _B PLL lock 1 _B PLL lock loss
USM	8	rh	Unsolicited Message Received Unsolicited message received indication. Unsolicited message indicates a system event to the other interface side. 0 _B no unsolicited message available 1 _B unsolicited message available

Field	Bits	Type	Description
PAR	9	rh	<p>PING Answer Received</p> <p>The received message was identified as PING.</p> <p>0_B no PING message available 1_B PING message received</p>
TXTE	10	rh	<p>TX transfer error occurred on a disabled TX channel.</p> <p>A disabled TX triggers an error interrupt, if:</p> <ul style="list-style-type: none"> • TX disabled on a pending or active data transfer. • TX CTS configuration change on a active CTS frame. <p>0_B no error situation occurred 1_B error situation occurred</p>
SFO	11	rh	<p>Synchronization FIFO overflow (in RX direction)</p> <p>Physical layer to Controller data synchronization FIFO in RX transfer direction hit an overflow situation.</p> <p>0_B RX synchronization is running well. 1_B RX FIFO overflow</p> <p><i>Note: This interrupt is an indication about a to slow SRI clock compared to Physical layer clock, which results in a overflow situation. (Minimum SRI frequency 40 MHz.)</i></p>
SFU	12	rh	<p>Synchronization FIFO underflow (in TX direction)</p> <p>Controller to Physical layer data synchronization FIFO in TX transfer direction hit an underflow situation.</p> <p>0_B TX synchronization is running well. 1_B TX FIFO underflow</p> <p><i>Note: This interrupt is an indication about a to slow SRI clock compared to Physical layer clock, which results in a underflow situation. (Minimum SRI frequency 40 MHz.)</i></p>

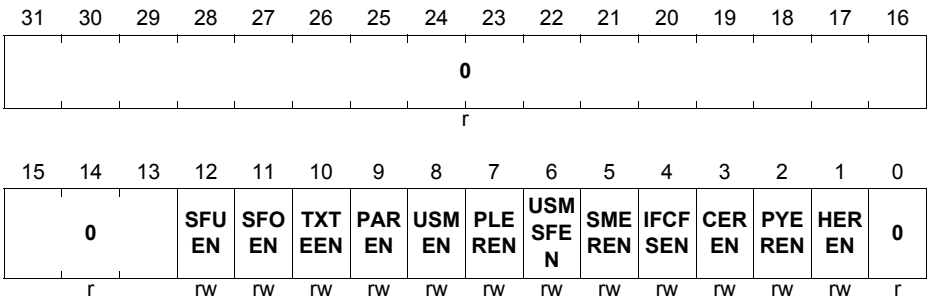
Field	Bits	Type	Description
0	[31:13], 0	r	Reserved

IRQEN

Interrupt enable register. An enabled register generates an pulsed interrupt. On a disabled interrupt the interrupt pulse does not come through on the interrupt line.

IRQEN

Interrupt enable register (44_H) **Reset Value: 0000 0000_H**

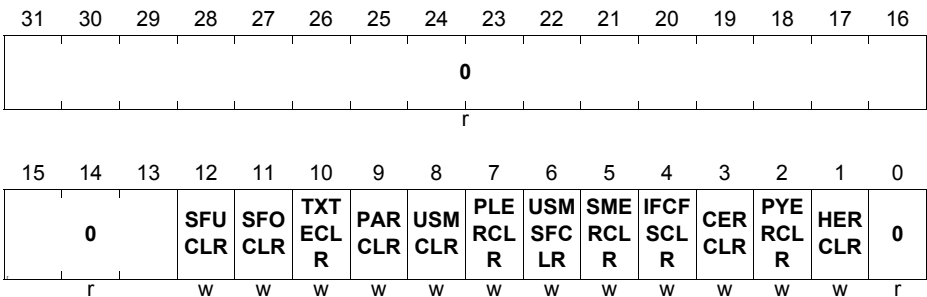


Field	Bits	Type	Description
HEREN	1	rw	Header error detected interrupt enable 0 _B Interrupt disabled 1 _B Interrupt enabled
PYEREN	2	rw	Payload error detected interrupt enable 0 _B Interrupt disabled 1 _B Interrupt enabled
CEREN	3	rw	HSCT command error interrupt enable 0 _B Interrupt disabled 1 _B Interrupt enabled
IFCFSEN	4	rw	HSCT interface control command send enable 0 _B Interrupt disabled 1 _B Interrupt enabled
SMEREN	5	rw	Speed Mode Switch Error interrupt enable 0 _B Interrupt disabled 1 _B Interrupt enabled
USMSFEN	6	rw	Unsolicited message frame send finished 0 _B Interrupt disabled 1 _B Interrupt enabled

Field	Bits	Type	Description
PLEREN	7	rw	PLL lost lock error interrupt enable 0 _B Interrupt disabled 1 _B Interrupt enabled
USMEN	8	rw	Unsolicited Message received enable 0 _B Interrupt disabled 1 _B Interrupt enabled
PAREN	9	rw	PING Answer Received enable 0 _B Interrupt disabled 1 _B Interrupt enabled
TXTEEN	10	rw	TX disable error interrupt enable 0 _B Interrupt disabled 1 _B Interrupt enabled
SFOEN	11	rw	Synchronization FIFO overflow (in RX direction) interrupt enable 0 _B Interrupt disabled 1 _B Interrupt enabled
SFUEN	12	rw	Synchronization FIFO underflow (in TX direction) interrupt enable 0 _B Interrupt disabled 1 _B Interrupt enabled
0	[31:13], 0	r	Reserved

IRQCLR

This register is the Interrupt clear register. By writing a logic 1 to the register the interrupt in the IRQ register is cleared by a clear pulse. This means the interrupt clear methodology is write one to clear. Reading from this register presents the logic value 0 to software.

IRQCLR
Interrupt clear register
(48_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
HERCLR	1	w	Header error detected interrupt clear
PYERCLR	2	w	Payload error detected interrupt clear
CERCLR	3	w	HSCT command error interrupt clear
IFCFSCCLR	4	w	HSCT interface control command send interrupt clear
SMERCLR	5	w	Speed Mode Switch Error interrupt clear
USMSFCLR	6	w	Unsolicited message frame send finished interrupt clear
PLERCLR	7	w	PLL lost lock error interrupt clear
USMCLR	8	w	Unsolicited Message received clear
PARCLR	9	w	PING Answer received clear
TXTECLR	10	w	TX disable error interrupt clear
SFOCLR	11	w	Synchronization FIFO overflow (in RX direction) interrupt clear
SFUCLR	12	w	Synchronization FIFO underflow (in TX direction) interrupt clear

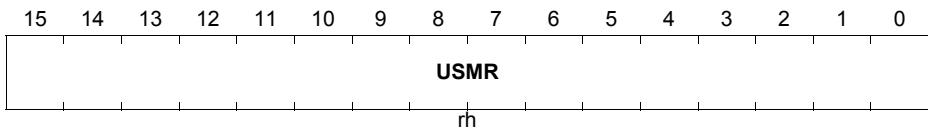
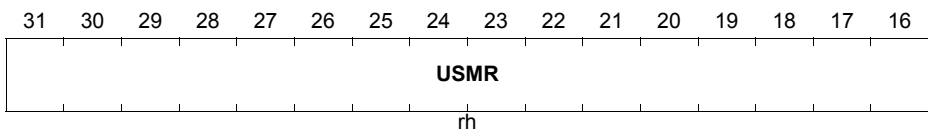
Field	Bits	Type	Description
0	[31:13], 0	r	Reserved

USMR

The Unsolicited Status message register is available to capture unsolicited messages received with the logical channel command encoding in the header of the frame (Logical Channel Type Hex 1).

USMR

Unsolicited Status Message Received (50_H) **Reset Value: 0000 0000_H**



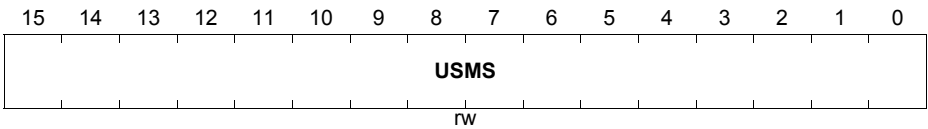
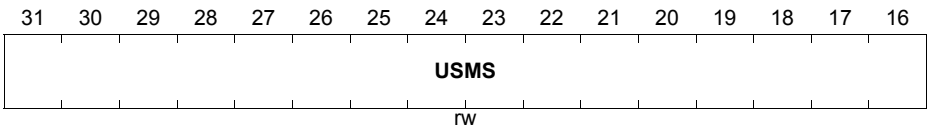
Field	Bits	Type	Description
USMR	[31:0]	rh	Unsolicited status message received The register contains the last received unsolicited status message.

USMS

A write to the register does activate the interface to send out an unsolicited status message to the other interface side. The Logical Channel Type reflected in the header is Hex 1 and the Payload is 32-bit always. An unsolicited frame has a higher priority than a Frame based on a data channel. In between frames the USMS gets higher priority than the data transfer requested by the HSSL.

USMS

Unsolicited Status Message Send (54_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
USMS	[31:0]	rw	Unsolicited status message send Writing to the register triggers a unsolicited status message to be send to the other interface side.

BPI_FPI Module Registers (Single Kernel Configuration)

Clock Control Register (CLC)

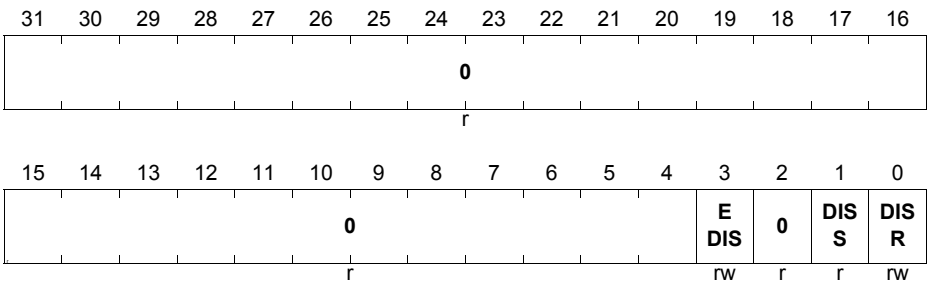
The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI_FPI for the HSCT module. Where a module kernel is connected to the CLC clock control interface, CLC register controls the module clock, sleep mode and disable mode for the module.

CLC

Clock Control Register

(00_H)

Reset Value: 0000 0003_H



Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control (clock and local access) of the module. The disable is performed after an acknowledge signal from the core logic is received.
DISS	1	r	Module Disable Status Bit Bit indicates the current status of the module and is set after the requested disable is active.
EDIS	3	rw	Sleep Mode Enable Control Used to control module's sleep mode. A system sleep mode request only becomes active, if enabled by this bit. If sleep request is active, no writes to the module kernel are possible anymore.

Field	Bits	Type	Description
0	[31:4], 2	r	Reserved Read as 0; should be written with 0.

OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The OCS register can only be written when the OCDS is enabled.

If OCDS is being disabled, the OCS register value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32

The OCS register includes the module related control bits for the OCDS Trigger Bus (OTGB).

The register can only be written and the OCS control register bits are only effective while the OCDS is enabled (OCDS enable = '1'). While OCDS is not enabled, OCS reset values/modes are effective.

OCS

OCDS Control and Status (FFE8_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SUS STA	SUS _P	SUS				0								
r	rh	w	rw				r								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TG _P	TGB	TGS	
r												w	rw	rw	

Field	Bits	Type	Description
TGS	[1:0]	rw	Trigger Set for OTGB0/1 0 _H No Trigger Set output 1 _H TS16_HSCT others , reserved (no Trigger Set selected)
TGB	2	rw	OTGB0/1 Bus Select 0 _B Trigger Set is output on OTGB0 1 _B Trigger Set is output on OTGB1
TG_P	3	w	TGS, TGB Write Protection TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.

Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately (read from registers still possible). <i>Note: After a Hard suspend the HSCT and higher layer protocol module must be reset. A new initialization sequence is required afterwards</i> others , reserved
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:4], [31:30]	r	Reserved Read as 0; must be written with 0.

The HSCT has one OCDS Trigger set available which gives a maximum of 16 triggers. The next table describes the available triggers.

Table 21-16 TS16_HSCT OCDS Trigger Set

Bits	Name	Description
0	HRXRQ	HSSL Interface RX transfer direction request
1	HRXCTS	HSSL Interface RX transfer direction CTS
2	HTXCTS	HSSL Interface TX transfer direction CTS
3	HTXRQ	HSSL Interface TX transfer direction request
4	TXF	First Byte transfer of transfer frame in TX transfer direction at Controller to Physical Layer interface
5	TXL	Last Byte transfer of transfer frame tin TX transfer direction at Controller to Physical Layer interface
6	TXW	Transmit direction wake-up at controller to physical layer interface
7	TXS	Transmit direction sleep indication at controller to physical layer interface

Table 21-16 TS16_HSCT OCDS Trigger Set

Bits	Name	Description
8	RXFS	Receive direction frame start at physical layer to controller interface.
9	DFSMI	Deframer state machine not in idle.
10	IFCT	Interface frame command trigger.
[15:11]		Reserved (value is 0)

Access Enable Register 0 (ACCEN0)

The Access Enable Register 0 controls write access for transactions with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... ,EN31 -> TAG ID 011111B.

ACCEN0
Access Enable Register 0
(FFFC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Field	Bits	Type	Description
ENn (n = 0-31)	n	r/w	Access Enable for Master TAG ID n This bit enables read/write access to the protected resources for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register 1 (ACCEN1)

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID master peripheral mapping). The BPI_FBI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

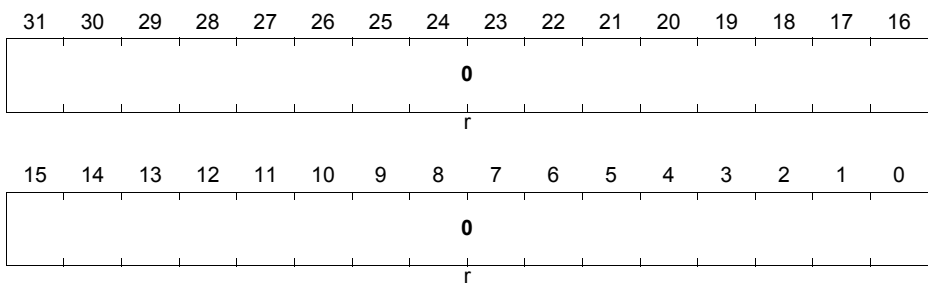
Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000B, EN1 -> TAG ID 100001B, ..., EN31 -> TAG ID 111111B.

ACCEN1

Access Enable Register 1

(FFF8_H)

Reset Value: 0000 0000_H

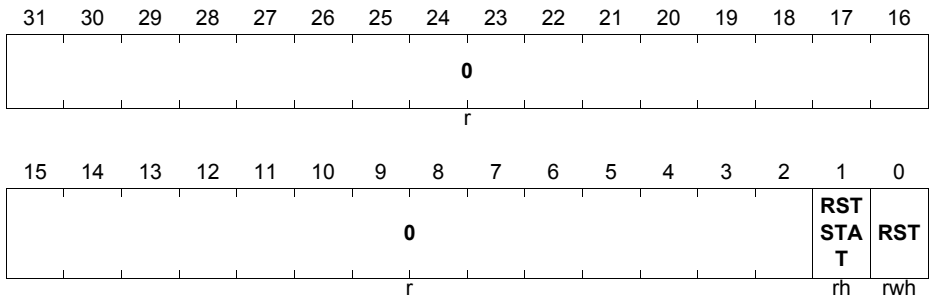


Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLE.CLR register bit.

KRST0
Reset Register 0
(FFF4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set.</p> <p>0_B No kernel reset was requested 1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p> <p><i>Note: It is strongly recommended to reset the HSCT and HSSL in sequence to avoid communication issues.</i></p>
RSTSTAT	1	rh	<p>Kernel Reset Status</p> <p>This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits.</p> <p>0_B No kernel reset was executed 1_B Kernel reset was executed</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.</p>
0	[31:2]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Kernel Reset Register 1(KRST1)

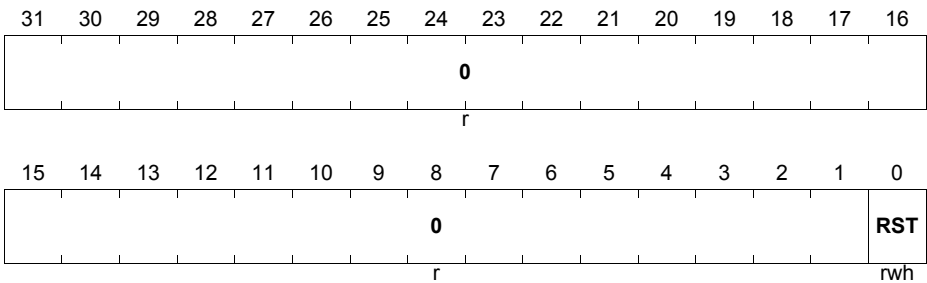
The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

KRST1

Reset Register 1

(FFF0_H)

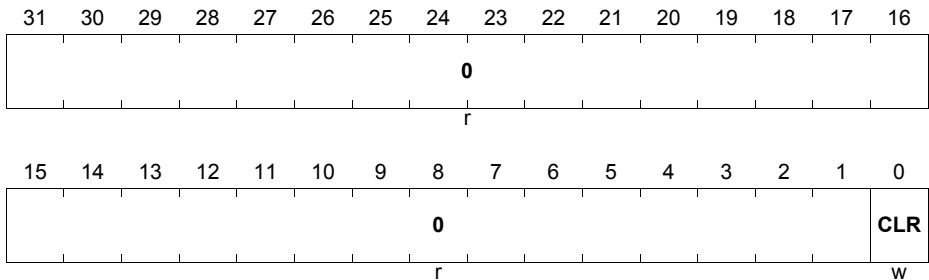
Reset Value: 0000 0000_H



Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set.</p> <p>0_B No kernel reset was requested</p> <p>1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p> <p><i>Note: It is strongly recommended to reset the HSCT and HSSL in sequence to avoid communication issues.</i></p>
0	[31:1]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Kernel Reset Status Clear Register (KRSTCLR)

The Kernel Reset Register Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

KRSTCLR
Reset Status Clear Register
(FFEC_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

21.15.5 Suspend, Sleep and Power-Off Behavior

Generally, the HSCT module transmits and receives data and control information via the differential signal interface.

A power down of the module can be done in any situation of the transmission. After Power up the interfaces is in low speed mode, again. It is strongly recommended to have the transfers stopped from the higher layer protocol before sending the HSCT layer into power off state. In order to power down the module in a well defined way, the user software must take care that the power down request is issued after ongoing transfers or stop the transfers before. Issuing an power down within a transfer - all data is lost.

21.15.5.1 OCDS Suspend

OCDS suspend request, for debugging purposes, simply freezes the HSCT module in the current state. This corresponds to the hard suspend of the whole module, that is both the transmitter and the receiver. After the end of the suspend state, the receive data will be most probably corrupted. Therefore, the whole interface must be reset, re initialized and reactivated.

21.15.5.2 Sleep Mode

Going into sleep mode implies that the HSCT module has finished all activities and that the HSSL TX FIFOs are empty and the last data of the frame are send. It is responsibility of the user software to enable sleep mode SLEEPCTRL.SLPEN.

21.15.5.3 Disable Request (Power-Off)

Going into power off implies that the HSCT module has finished all activities.

21.15.6 References

- [1] IEEE Std 1596.3-1996 - IEEE Standard for Low-Voltage Differential Signals (LVDS) for Scalable Coherent Interface (SCI)
- [2] LVDSH C65 ACDC specification from Infineon.

22 Micro Second Channel (MSC)

This chapter describes the Micro Second Channel Interfaces , MSC0 and MSC1 of the TC26x B-Step. It contains the following sections:

- Functional description of the MSC kernel (see [Page 22-3](#))
- MSC kernel register descriptions (see [Page 22-50](#))
- TC26x B-Step implementation-specific details and registers of the MSC module (port connections and control, interrupt control, address decoding, and clock control, see [Page 22-92](#))
- >> [Registers Overview](#)

Note: The MSC kernel register names described in [Section 22.3](#) are also referenced in the TC26x B-Step User's Manual by the module name prefix "MSC0_" for the MSC0 module and by "MSC1_" for the MSC1 module.

Micro Second Channel (MSC)

MSC Applications

The MSC is a serial interface that is especially designed to connect external power devices to the TC26x B-Step. The serial data transmission capability minimizes the number of pins required to connect such external power devices. Parallel data information (coming from the timer units) or command information is sent out to the power device via a high- speed synchronous serial data stream (downstream channel). The MSC receives data and status back from the power device via a low-speed asynchronous serial data stream (upstream channel).

Parallel requests from on chip bus masters to a module will be executed sequentially via the on chip bus system. Read-modify-write feature provides an atomic read/write sequence where no other master can access the module in between. Module hardware semaphores are not supported.

Figure 22-1 shows a typical TC26x B-Step application in which an MSC interface controls two power devices. Output data is provided by the module.

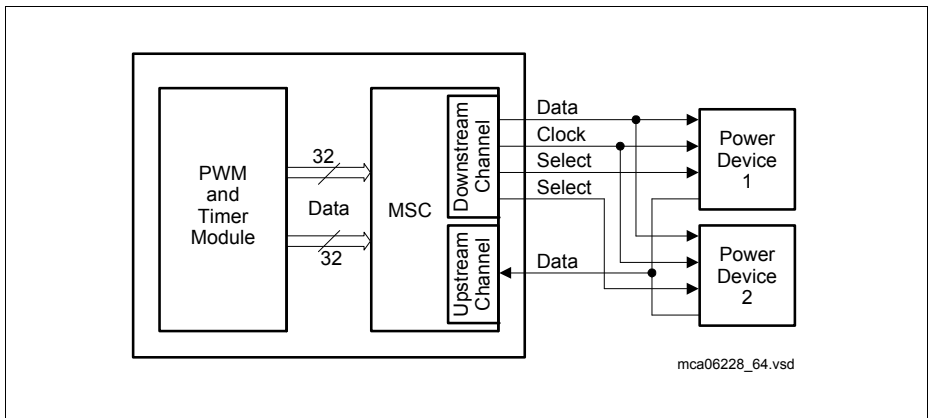


Figure 22-1 MSC to External Power Device Connection

Some applications are:

- Control of the external power switching unit via the downstream channel
- Receiving information back from power switching unit
- Serial connections of the TC26x B-Step to other peripheral devices

22.1 MSC Kernel Description

This section describes the functionality of the MSC kernel.

22.1.1 Overview

The MSC interface provides a serial communication link typically used to connect power switches or other peripheral devices. The serial communication link includes a fast synchronous downstream channel and a slow asynchronous upstream channel. **Figure 22-2** shows a global view of the MSC interface signals.

The MSC module features functional extensions, but is compatible to the MSC interface of the TC17xx devices.

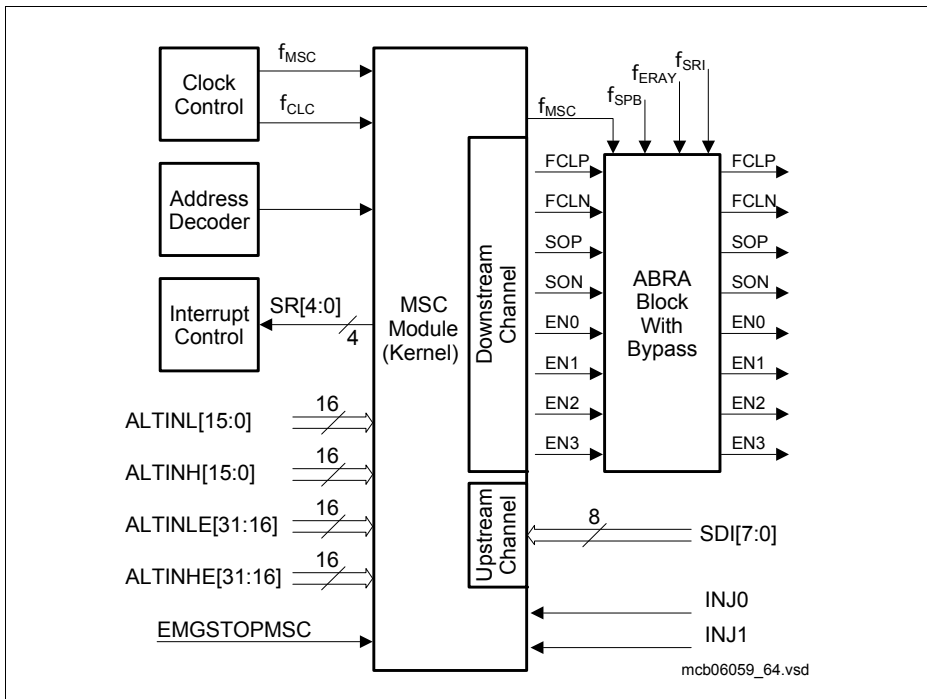


Figure 22-2 General Block Diagram of the MSC Interface

The downstream and upstream channels of the MSC module communicate with the external world via nine I/O lines. Eight output lines are required for the serial communication of the downstream channel (clock, data, and enable signals). One out of eight input lines $SDI[7:0]$ is used as serial data input signal for the upstream channel. The source of the serial data to be transmitted by the downstream channel can be MSC

Micro Second Channel (MSC)

register contents or data that is provided at the ALTINL/ALTINH input lines. These input lines are typically connected to other on-chip peripheral units (for example with a timer unit like the GTM). An emergency stop input signal makes it possible to set bits of the serial data stream to dedicated values in emergency case.

Clock control, address decoding, and interrupt service request control are managed outside the MSC module kernel. Service request outputs are able to trigger an interrupt or a DMA request.

Features

- Fast synchronous serial interface to connect power switches in particular, or other peripheral devices via serial buses
- High-speed synchronous serial transmission on downstream channel
 - Serial output clock frequency: $f_{FCL} = f_{MSC}/2$ ($f_{MSCmax} = 100$ MHz)
 - Fractional clock divider for precise frequency control of serial clock f_{MSC}
 - Command, data, and passive frame types
 - Start of serial frame: Software-controlled, timer-controlled, or free-running
 - Transmission with or without SEL bit
 - Flexible chip select generation indicates status during serial frame transmission
 - Emergency stop without CPU intervention
- Low-speed asynchronous serial reception on upstream channel
 - Baud rate: f_{MSC} divided by 4, 8, 16, 32, 64, 128, or 256 ($f_{MSCmax} = 100$ MHz)
 - Standard asynchronous serial frames
 - Programmable upstream data frame length (16 or 12 bits)
 - Parity error checker
 - 8-to-1 input multiplexer for SDI lines
 - Built-in spike filter on SDI lines
 - Programmable delay of the receive interrupt after the last stop bit (0 or 1 bit time)
- Selectable pin types of downstream channel interface:
four LVDS differential output drivers or four digital GPIO pins
- Module reset available
- Advanced features:
 - Asynchronous Baud Rate Adjustment Block
 - one extra interrupt
 - 64-bit data frames extension
 - fully compatible with the 32-bit data frames module version

22.1.2 Downstream Channel

The downstream channel performs a high-speed synchronous serial transmission of data to external devices. Its 64-bit shift register is divided into two 32-bit parts, SRL and SRH. Each bit of SRL and SRH can be selected to be delivered by the downstream data register DD, by the Downstream Command Register DC, or by two 16-bit wide input signal buses ALTINL and ALTINH.

Figure 22-3 is a diagram of the MSC downstream channel.

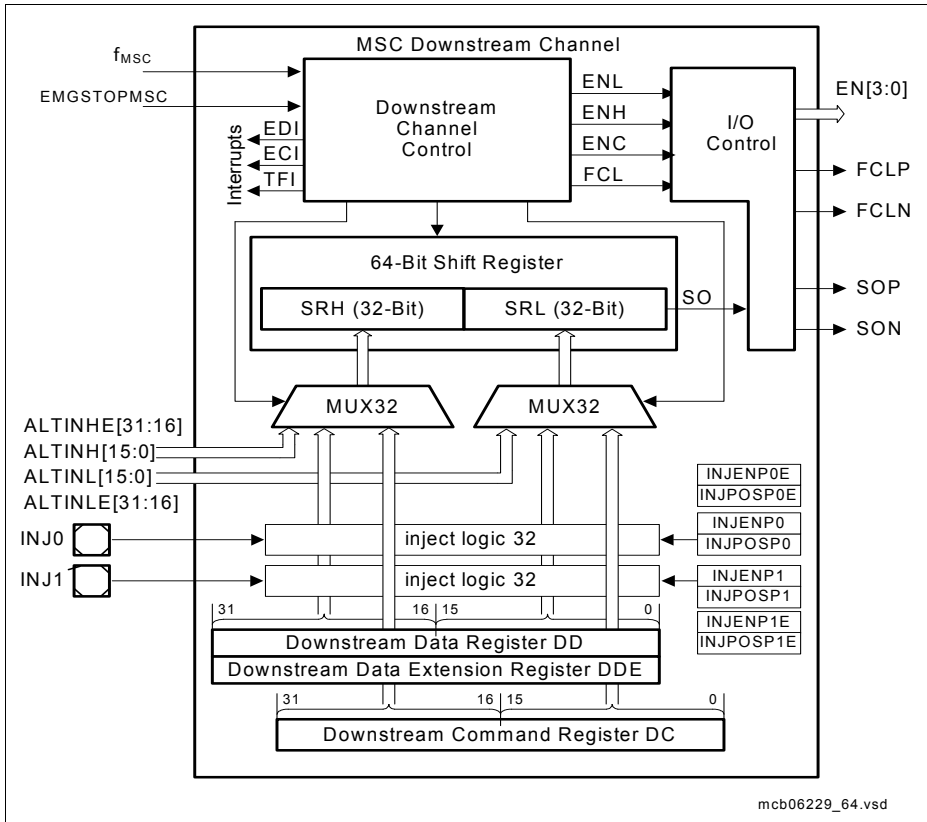


Figure 22-3 Downstream Channel Block Diagram

The enable signals ENL, ENH, and ENC indicate certain phases of the serial transmission in relation to the serial clock FCL. In the I/O control logic, these signals can be combined to four enable/select outputs EN[3:0]. For supporting differential output

Micro Second Channel (MSC)

drivers, the serial clock output FCL and the serial data output SO are available in both polarities, indicated by the signal name suffix “P” and “N”.

The emergency stop input line EMGSTOPMSC is used to indicate an emergency stop condition of a power device. In emergency case, shift register bits can be loaded bit-wise from the downstream data register instead from the ALTINL and ALTINH buses.

22.1.2.1 Frame Formats and Definitions

This section describes the frame formats and definitions of the MSC.

Basic Definitions

Figure 22-4 shows the layout and definitions of a downstream frame. A downstream frame is composed of an active phase and a passive phase. During the active phase, data transmission takes place and during the passive phase no data is transmitted at SO. The active phase is split into two parts: The SRL active phase in which the content of the shift register low part SRL is transmitted, and the SRH active phase in which the content of the shift register high part SRH is transmitted. At the beginning of the SRL and SRH active phase, a selection bit (SELL) can be optionally inserted into the serial data stream. In the frame shown in **Figure 22-4**, SELL is generated at the beginning of the SRL active phase (not for the SRH active phase). The least significant bits of SRL and SRH are sent out first.

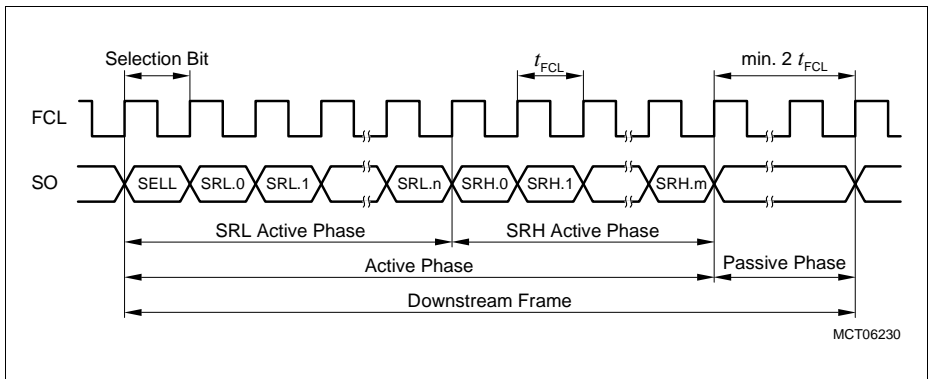


Figure 22-4 Downstream Channel Frame

The MSC downstream channel uses three types of frame formats for operation:

- Command frames, indicated by SELL = 1
- Data frames, indicated by SELL = 0 or SELL bit insertion disabled
- Passive time frame, indicated by ENL = ENH = 0

Micro Second Channel (MSC)

Command Frames

A command frame has two active phase parts, SRL active phase and SRH active phase. The command frame always starts with a high-level selection bit, independently whether the selection bit insertion (as defined by bit DSC.ENSELL) is enabled or not. The number of the bits transmitted during SRL and SRH active phases (except the selection bit) is defined by bit field DSC.NBC. SRL and SRH are combined to a value up to 64-bit whose length can be selected from 0 up to 32 bits. In other words, whenever bits of SRH are transmitted, they are always preceded by the transmission of the complete SRL content.

During the active phase of a command frame, the enable output signal ENC becomes active. The enable output signals ENL and ENH remain inactive.

The passive phase of a command frame always has a fixed length of $2 \times t_{FCL}$. The diagram shown in **Figure 22-5** assumes that the FCL clock is only generated during the active phase of the command frame (OCR.CLKCTRL = 0).

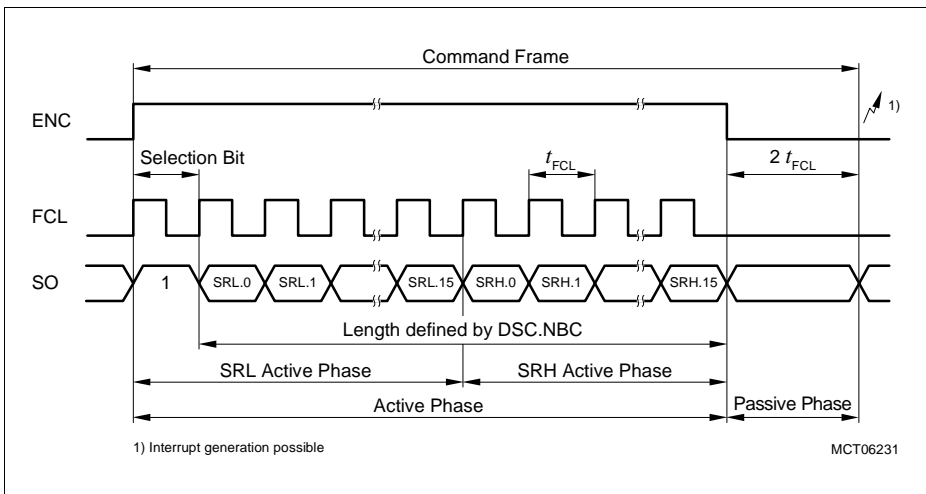


Figure 22-5 Command Frame Layout

Micro Second Channel (MSC)

Table 22-1 shows the programming of the bits to be transmitted and the resulting length of the complete command frame.

Table 22-1 Command Frame Length

DSC.NBC	SRL/SRH Bits that are Transmitted in Active Phase	Command Frame Length in t_{FCL} Periods
000000 _B	No bit shifted out	$1 + 0 + 2 = 3$
000001 _B	SRL[0] shifted out	$1 + 1 + 2 = 4$
000010 _B	SRL[1:0] shifted out	$1 + 2 + 2 = 5$
000011 _B	SRL[2:0] shifted out	$1 + 3 + 2 = 6$
...
001111 _B	SRL[14:0] shifted out	$1 + 15 + 2 = 18$
010000 _B	SRL[15:0] shifted out	$1 + 16 + 2 = 19$
010001 _B	SRL[15:0] and SRH[0] shifted out	$1 + 17 + 2 = 20$
010010 _B	SRL[15:0] and SRH[1:0] shifted out	$1 + 18 + 2 = 21$
010011 _B	SRL[15:0] and SRH[2:0] shifted out	$1 + 19 + 2 = 22$
...
011111 _B	SRL[15:0] and SRH[14:0] shifted out	$1 + 31 + 2 = 34$
100000 _B	SRL[15:0] and SRH[15:0] shifted out	$1 + 32 + 2 = 35$
Other NBC combinations	Reserved; do not use these bit combinations.	

Micro Second Channel (MSC)

Data Frames

A data frame has two active phase parts, SRL active phase and SRH active phase. The number of bits that are transmitted can be programmed separately for each of these two phases. Bit field DSC.NDBL determines the number of SRL bits that are transmitted during the SRL active phase and DSC.NDBH determines the number of SRH bits that are transmitted during the SRH active phase.

SRL and SRH active phases can start with a low-level selection bit when enabled by bits DSC.ENSELL or DSC.ENSELH.

During the SRL active phase of a data frame, the enable output signal ENL becomes active and during the SRH active phase of a data frame, the enable output signal ENH becomes active. The enable output signal ENC remains inactive.

The length of the data frame's passive phase is variable and is defined by bit field DSC.PPD. It can be within a range of $2 \times t_{FCL}$ up to $31 \times t_{FCL}$. The diagram shown in Figure 22-6 assumes that the FCL clock is only generated during the active phase of the data frame (OCR.CLKCTRL = 0).

Table 22-2, Table 22-3, and Table 22-5 show the definitions of the five data frame parameters that determine the layout of the data frame.

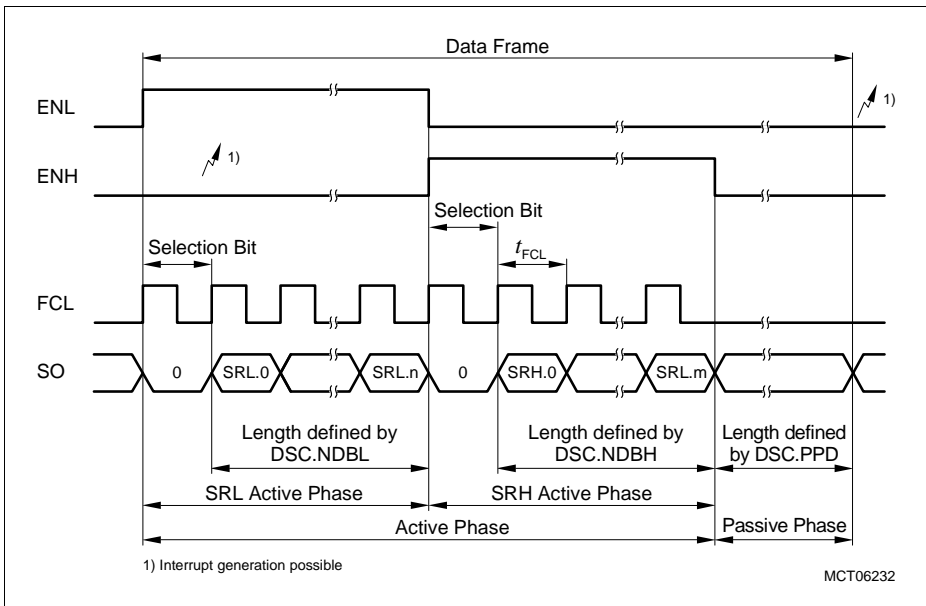


Figure 22-6 Data Frame Layout

Table 22-2 Data Frame Selection Bit Parameters

DSC.ENSELL	Selection Bit	DSC.ENSELH	Selection Bit
0	No selection bit inserted at the beginning of the SRL active phase	0	No selection bit inserted at the beginning of the SRH active phase
1	A low level selection bit is inserted at the beginning of the SRL active phase	1	A low level selection bit is inserted at the beginning of the SRH active phase

*Note: The MSC module can operate in two modes: standard (up to 32 data bits) or extended (up to 64 data bits). The mode is selected by using the bit **DSCE.EXEN**.*

Table 22-3 Data Frame SRL/SRH Length Parameters, Standard Mode

DSCE.NDBLE=0 and DSC.NDBL	SRL Bits Transmitted in SRL Active Phase	DSCE.NDBHE=0 and DSC.NDBH	SRH Bits Transmitted in SRH Active Phase
0 0000 _B	No SRL bit transmitted	0 0000 _B	No SRH bit transmitted
0 00001 _B	SRL[0]	0 00001 _B	SRH[0]
0 00010 _B	SRL[1:0]	0 00010 _B	SRH[1:0]
0 00011 _B	SRL[2:0]	0 00011 _B	SRH[2:0]
...
0 01111 _B	SRL[14:0]	0 01111 _B	SRH[14:0]
0 10000 _B	SRL[15:0]	0 10000 _B	SRH[15:0]
Other bit combinations	Reserved; do not use these bit combinations.	Other bit combinations	Reserved; do not use these bit combinations.

Table 22-4 Data Frame SRL/SRH Length Parameters, Extended Mode

DSCE.NDBLE=1 and DSC.NDBL	SRL Bits Transmitted in SRL Active Phase	DSCE.NDBHE=1 and DSC.NDBH	SRH Bits Transmitted in SRH Active Phase
1 00000 _B	No SRL bit transmitted	1 00000 _B	No SRH bit transmitted
1 00001 _B	SRL[16:0]	1 00001 _B	SRH[16:0]
1 00010 _B	SRL[17:0]	1 00010 _B	SRH[17:0]
1 00011 _B	SRL[18:0]	1 00011 _B	SRH[18:0]
...
1 01111 _B	SRL[30:0]	1 01111 _B	SRH[30:0]

Micro Second Channel (MSC)

Table 22-4 Data Frame SRL/SRH Length Parameters, Extended Mode (cont'd)

DSCE.NDBLE=1 and DSC.NDBL	SRL Bits Transmitted in SRL Active Phase	DSCE.NDBHE=1 and DSC.NDBH	SRH Bits Transmitted in SRH Active Phase
1 10000 _B	SRL[31:0]	1 10000 _B	SRH[31:0]
Other bit combinations	Reserved; do not use these bit combinations.	Other bit combinations	Reserved; do not use these bit combinations.

Table 22-5 Data Frame Passive Phase Length

DSC.PPD	Passive Phase Length
00000 _B	$2 \times t_{FCL}$
00001 _B	$2 \times t_{FCL}$
00010 _B	$2 \times t_{FCL}$
00011 _B	$3 \times t_{FCL}$
...	...
11110 _B	$30 \times t_{FCL}$
11111 _B	$31 \times t_{FCL}$

The following formula determines the number of t_{FCL} cycles of an up to 32-bit data frame: All parameters (bits and bit fields) are located in register DSC.

$$N_{cycles} = ENSELL + NDBL + ENSELH + NDBH + PPD \quad (22.1)$$

Note that in the formula above, PPD must be set to 2 when $DSC.PPD \leq 00010_B$.

The following formula determines the number of t_{FCL} cycles of an extended, up to 64-bit data frame, in case DSC.NDBL and DSC.NDBH are not equal to zero:

$$N_{cycles} = ENSELL + NDBL + NDBLE \cdot EXEN \cdot 16 + ENSELH + NDBH + NDBHE \cdot EXEN \cdot 16 + PPD + PPDE \cdot 32 \quad (22.2)$$

Otherwise, the corresponding factor $NDBLE \cdot EXEN \cdot 16$ or $NDBHE \cdot EXEN \cdot 16$ must be taken as zero (see [Table 22-4](#)).

Note that in the formula above, PPD must be set to 2 when $DSC.PPD \leq 00010_B$.

The parameters (bits and bit fields) are located in registers DSC and DSCE.

Attention: *In order to calculate the time between two consecutive TRPs, always use $ENSELL=1$ in the formulas above. $ENSELL$ does not influence the time between two consecutive TRPs, but only the length of a data*

Micro Second Channel (MSC)

frame. The difference in the starting point of a frame with ENSELL=0 and 1 is shown in the [Figure 22-13](#) and [Figure 22-14](#).

Passive Time Frames

A passive time frame has the length defined by the five data frame parameters according [Equation \(22.1\)](#). They are generated only in Data Repetition Mode. Under special conditions (command frame insertion), passive time frames can be shortened (see [Figure 22-10](#)).

During passive time frames, the data output SO have to be considered as invalid at the receiving device and the clock output FCL may toggle or not (as selected by bit OCR.CLKCTRL). The ENL and ENH enable signals remain at low level during a passive time frame.

22.1.2.2 Shift Register Operation

This section describes the SRL and SRH shift register loading.

SRL Shift Register Loading

During the SRL/SRH shift register load operation at the beginning of each downstream frame transmission, several parameters determine which information is loaded into the bits of the shift register. **Figure 22-7** shows the logic that is implemented for the SRL shift register loading operation. The logic for the SRH shift register loading operation is equivalent to the one for the SRL register. Its differences in data sources and register controls are described later in this section.

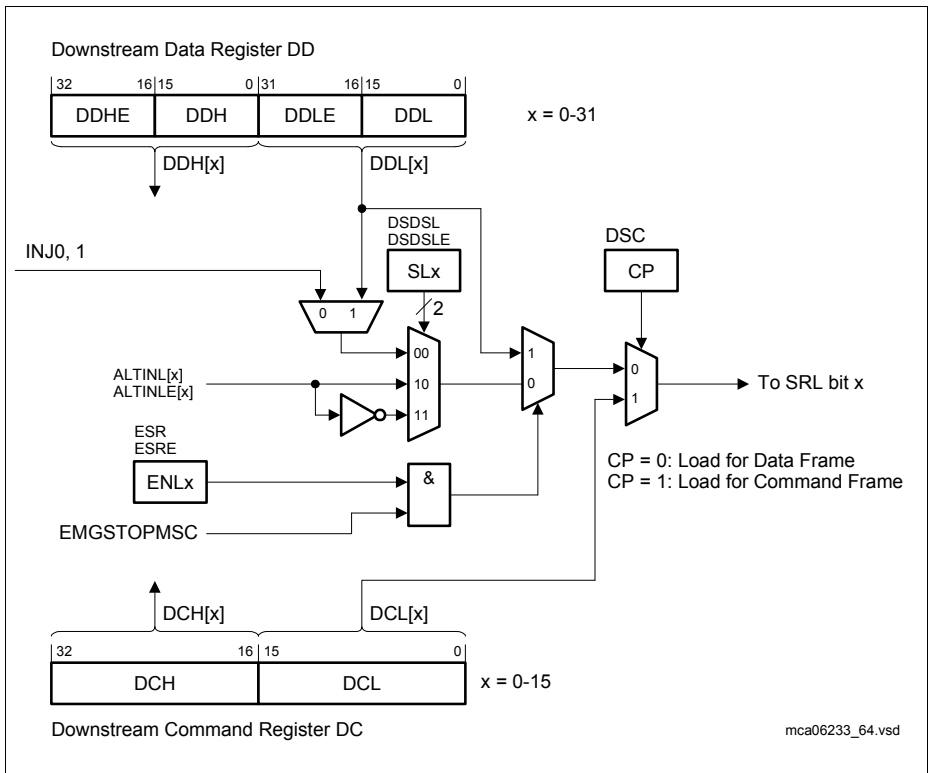


Figure 22-7 SRL Shift Register Data Loading Control

Micro Second Channel (MSC)

Four data sources can be selected for each SRL bit by using several control bits and one control signal:

- ALTINL input line (non-inverted)
- ALTINL input line (inverted)
- Bit of DD.DDL (downstream data register)
- Bit of DC.DCL (downstream control register)

When SRL is loaded for data frame transmission (DSC.CP = 0), bit fields DSDSL.SLx determine bit-wise which data is loaded into SRL bit x. The data source selection as controlled by DSDSL.SLx will only be effective when EMGSTOPMSC is inactive (at low level). When EMGSTOPMSC = 1 (active) during the load operation, all SRL[x] bits that are enabled for the emergency stop feature (bit ESR.ENLx = 1) are loaded directly with the corresponding bit DDL[x] of the downstream data register DD.

When SRL is loaded for command frame transmission (DSC.CP = 1), always the lower 16-bit part DCL of the downstream control register is loaded completely into SRL.

Table 22-6 summarizes all SRL data source selection capabilities (x = 0-31):

- in standard mode, **DSCE.EXEN** = 0, the selection configuration of up to 16 bits (x = 0-15) is taken into account
- in extended mode, **DSCE.EXEN** = 1, the selection configuration of up to 32 bits (x = 0-31) is taken into account

Table 22-6 SRL Data Source Selection Capabilities

DSC. CP	DSDSL. SLx DSDSLE.SLx	ESR. ENLx	EMGSTOP MSC	Selection
0	00 _B	0	–	Bit DD.DDL[x] is loaded into SRL[x].
	01 _B			Reserved.
	10 _B			State of ALTINL[x] input is loaded into SRL[x].
	11 _B			Inverted state of ALTINL[x] input is loaded into SRL[x].
	XX _B	1	1	Bit DD.DDL[x] is loaded into SRL[x].
1	XX _B	X	X	Bit fields DCL and DCH are completely loaded into SRL and SRH, respectively.

*Note: The data signals can be overruled by injected pin signals INJ0 and INJ1. See **DSCE** register.*

SRH Shift Register Loading

The SRH shift register load operation is equivalent to the SRL shift register load operation. The following differences must be taken into account for SRH shift register loading:

- Input lines ALTINH are connected instead of ALTINL input lines.
- DSDSH register bits control data source selection instead of DDSL register.
- Emergency stop is enabled by ESR.ENHx bits instead of ESR.ENLx bits.
- Bits of the downstream data register high part DDH are selected instead of DDL.
- Downstream control register high part DCH is selected instead of DCL.

22.1.2.3 External Signal Injection

Input signals from two pins can be independently injected at two data bit positions in a data frame. The injection takes place both in normal and in extended mode. The external signals are always injected at the defined position, and if the data frame is shorter than this position, then these signals remain unused. For example, if **DSCE**.INJPOS0 defines injection the position 14 of the SRL, and the **DSC**.NBDL=12, then all the data bits from position 12 and up remain unused, including the injected signal.

In normal mode the bits are counted from 0 to 31 from LSB to MSB, and in extended mode the bits are counted from 0 to 63 from LSB to MSB, see [Figure 22-8](#). The injection position is always counted as if the frame has the maximal length. In the first step of the preparation for a transmission the injection takes place, and then in the second step, if some fields are configured to be shorter or omitted (for example DDLE or DDH or any other), the unused bits are discarded and only the remaining valid bits are shifted out.

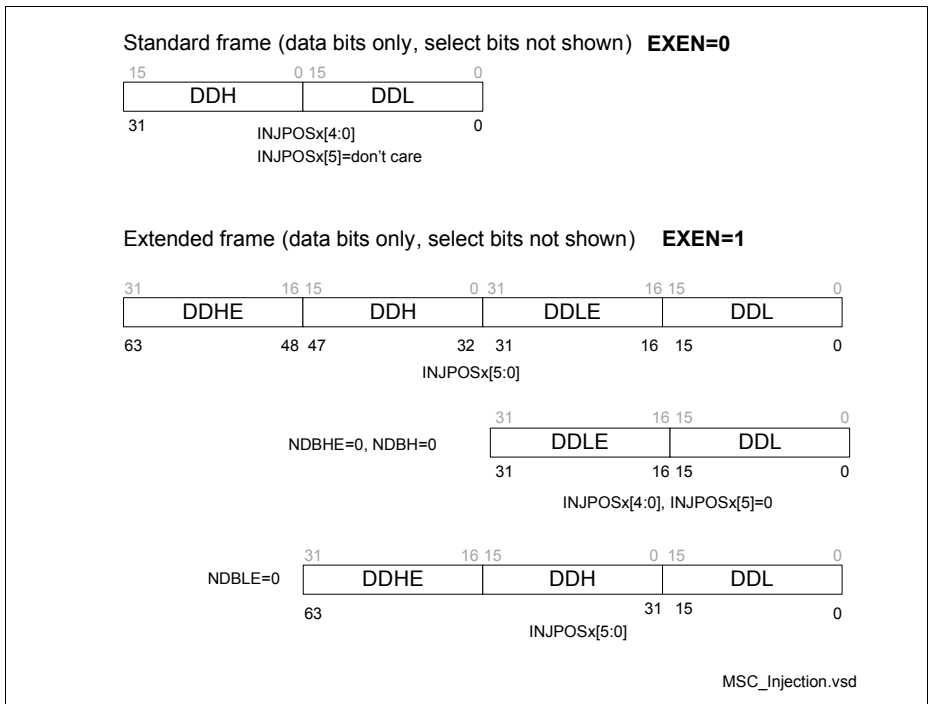


Figure 22-8 External Signal Injection

22.1.2.4 Transmission Modes

The downstream channel of the MSC makes it possible to select between two transmission modes:

- Triggered Mode, selected by $DSC.TM = 0$, or
- Data Repetition Mode, selected by $DSC.TM = 1$

Triggered Mode

In Triggered Mode, command frames or data frames are sent out as a result of a software event. When a frame transmission has been finished and no further frame transmission has been requested, the downstream channel returns to idle state and waits for the next frame transmission to be triggered by software.

Setting the $DSC.TM$ bit triggers immediately a frame transmission using the values existing at that moment.

When the Downstream Command Register DC is written, the command pending bit $DSC.CP$ becomes set and a command frame will be immediately started and sent out if the downstream channel is idle. If a data or command frame is currently processed and output, the command frame transmission is delayed, and started when the active downstream frame has been finished. The command pending bit $DSC.CP$ becomes cleared by hardware when the first bit of the command frame is sent out.

If the downstream channel is idle and the data pending bit $DSC.DP$ is set by writing $ISC.SDP$ with 1, a data frame will be immediately started and sent out if the downstream channel is idle. If a data frame or a command frame is currently processed and output, the data frame transmission is delayed and started when the active downstream frame has been finished. The data pending bit $DSC.DP$ becomes cleared by hardware when the first bit of the data frame is sent out.

A command frame always has priority over the data frame. This means that if both frame pending bits are set ($DSC.DP = DSC.CP = 1$), the command frame will always be sent first. Therefore, a pending data frame transmission will be delayed as long as no further command frame transmission is running or requested.

Figure 22-9 is a flow diagram of the Triggered Mode. This diagram especially shows the behavior of the data and command pending bits $DSC.DP$ and $DSC.CP$. If both frame pending bits are set ($DSC.DP = DSC.CP = 1$), the command frame will always be sent first, followed by the data frame (assuming no further command frame has been requested).

The type of the active frame that is currently processed and output is indicated by two status flags: $DSS.DFA$ is set during a data frame transmission and $DSS.CFA$ is set during a command frame transmission. Further, the downstream counter $DSS.DC$ indicates the number of shift clock periods that have been elapsed since the start of the current frame.

Micro Second Channel (MSC)

In Triggered Mode, the shift register loading event as described in [Section 22.1.2.2](#) occurs just before a command or data frame transmission is started.

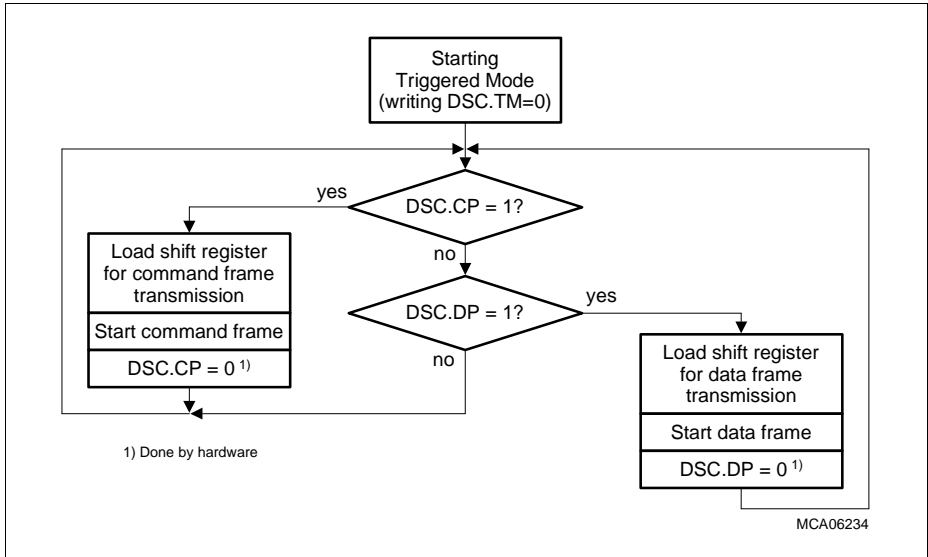


Figure 22-9 Triggered Mode Flow Diagram

Data Repetition Mode

In Data Repetition Mode, data frames are sent out continuously without any software interaction. In the time gap between two consecutive data frames, passive time frames can be inserted. The number of passive time frames to be inserted (0 to 15) is defined by bit field DSS.NPTF. The duration of data frame (t_{DF}) and passive time frames (t_{PTF}) is determined by the five data frame parameters (see [Equation \(22.1\)](#)). These parameters determine the time reference points (TRP) at which a data, command or passive time frames is started (see diagram A in [Figure 22-10](#)).

The automatic data frame generation is controlled by the data pending bit DSC.DP. This bit is set at the end of the last passive data frame if a command frame is running or starts at this TRP, overruling a data frame. DSC.DP is cleared by hardware when the pending data frame starts with transmission at the next TRP. Data Frames are always aligned to time reference points. This means they always start at a TRP. Passive time frames can be shortened. This is especially the case when command frames are inserted.

Continuous data frame transmission can be interrupted by insertion of command frames. Command frames are initiated by software. When the downstream command register DC is written, the command pending bit DSC.CP is set by hardware. CP = 1 indicates that the MSC starts a command frame at the next TRP, independently of whether a data

Micro Second Channel (MSC)

frame (indicated by DSC.DP = 1) or passive time frame should be started with the next TRP. This means also that command frames are always aligned to time reference points.

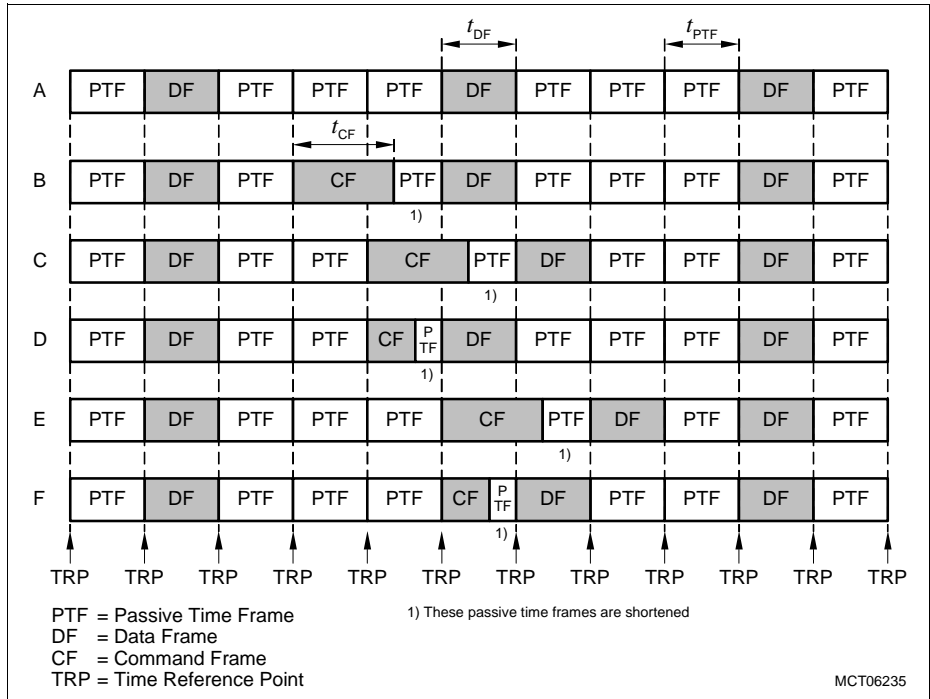


Figure 22-10 Data Repetition Mode Frame Examples with DSS.NPTF = 0011_B

Diagrams B to F in Figure 22-10 show the command frame insertion in Data Repetition Mode.

In diagram B, a command frame has been requested during the first passive time frame after the data frame, and is inserted at the next TRP. In diagrams C and D, a command frame has been requested during the second passive time frame, and is inserted at the time reference point of the last nominal passive time frame.

When the command frame and data frame is not of the same length (this is the case in diagram B to F), a shortened passive time frame is inserted until the next TRP is reached. This ensures that the next data or normal passive time frame is again aligned to a TRP.

Figure 22-11 is a flow diagram of the Data Repetition Mode. This diagram especially shows the behavior of the data and command pending bits DSC.DP and DSC.CP. If both frame pending bits are set (DSC.DP = DSC.CP = 1), the command frame will always be

Micro Second Channel (MSC)

sent first, followed by the data frame when the next TRP is reached (assuming no further command frame has been requested).

When the last passive frame is transmitted, DSC.DP becomes set by hardware. This triggers the start of a data frame when the next TRP is reached.

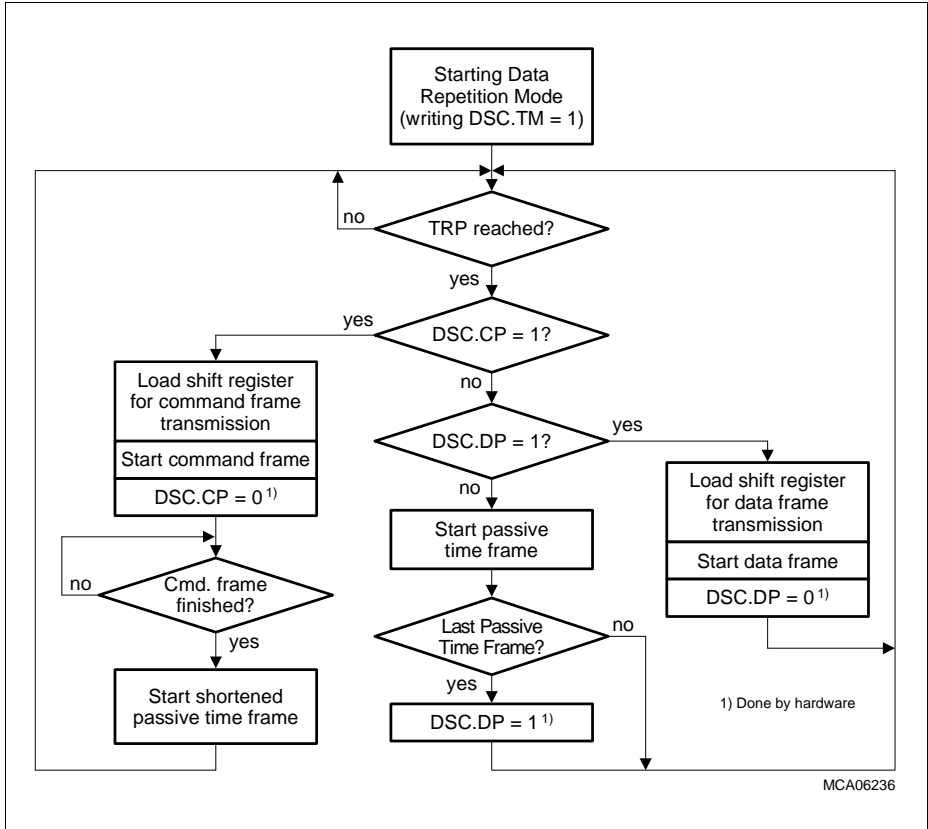


Figure 22-11 Data Repetition Mode Flow Diagram

The type of the active frame (data or command frame) that is currently processed and output is indicated by two status flags: DSS.DFA is set during a data frame transmission and DSS.CFA is set during a command frame transmission. Further, the downstream counter DSS.DC indicates the number of shift clock periods that have been elapsed since the start of the current data, command, or passive time frame.

Micro Second Channel (MSC)

As in Triggered Mode, the shift register loading event as described in [Section 22.1.2.2](#) occurs in Data Repetition Mode just before a TRP, this means shortly before a command or data frame transmission is started.

Passive Frame Counter in Data Repetition Mode

In Data Repetition Mode, a passive time frame counter DSS.PFC indicates how many time frames have been already transmitted after the last regular data frame occurrence. The passive time frame counter counts up from 0000_B to the value which has been written into bit field DSS.NPTF (number of passive time frames). DSS.PFC = 0000_B indicates that a data frame is requested for transmission.

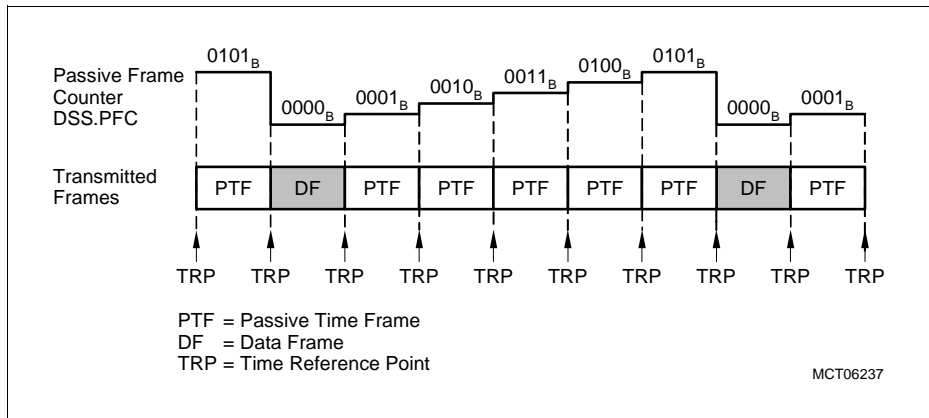


Figure 22-12 Passive Frame Counter Operation (with DSS.NPTF = 0101_B)

22.1.2.5 Downstream Counter and Enable Signals

During downstream channel operation, a 8-bit downstream counter DSS.DC is counting FCL shift clock periods. With the loading of the shift register, the downstream counter is reset to 00_H and started for counting up to the end of the downstream frame (end of passive phase).

In Triggered Mode, the downstream counter stops counting at the end of the passive phase and waits until a new downstream frame is started.

In Repetition Mode, the downstream counter does not stop at the end of the passive phase but is reset and starts counting up again with the next frame, independently whether a data frame, command frame, or passive time frame is started as next frame.

Figure 22-13 shows an example of downstream channel data frame transmission. In this example, the selection bit for the SRL active frame is enabled (ENSELL = 1), and the selection bit for the SRH active frame is disabled (ENSELH = 0). With loading of the shift register SRL/SRH, the downstream counter is reset and then starts counting up with each FCL clock until the end of the passive phase. ENL is set to high level at the beginning of the SRL active frame selection bit.

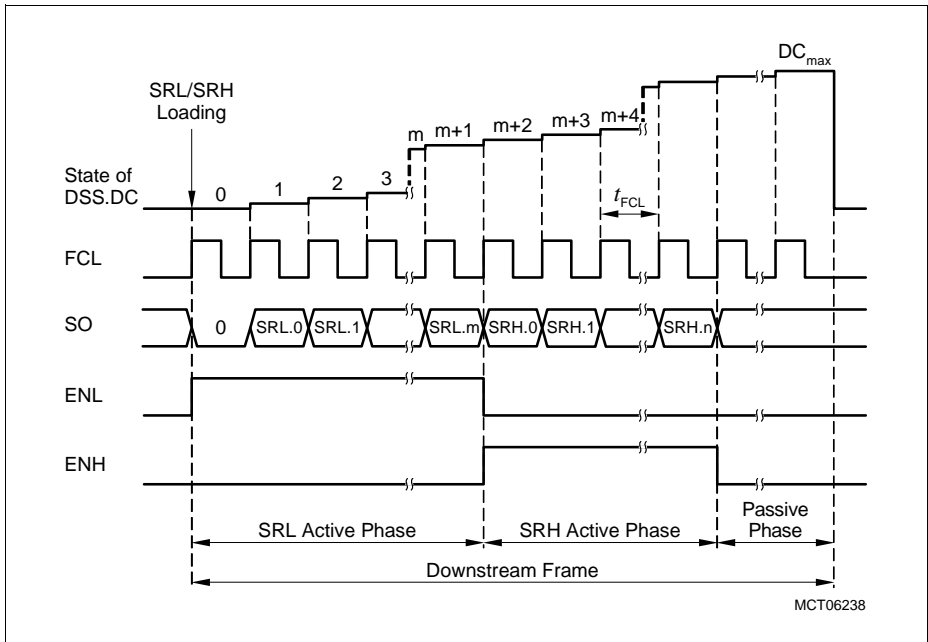


Figure 22-13 Shift Clock Counting: Data Frame with ENSELL = 1 and ENSELH = 0

Micro Second Channel (MSC)

When the selection bit for the SRL active frame is disabled (ENSELL = 0, see [Figure 22-14](#)), the loading of the shift register SRL/SRH (and reset of the downstream counter) occurs one FCL clock cycle before the first data bit SRL.0 is output. ENL is set to high level with the beginning of the first data bit SRL.0.

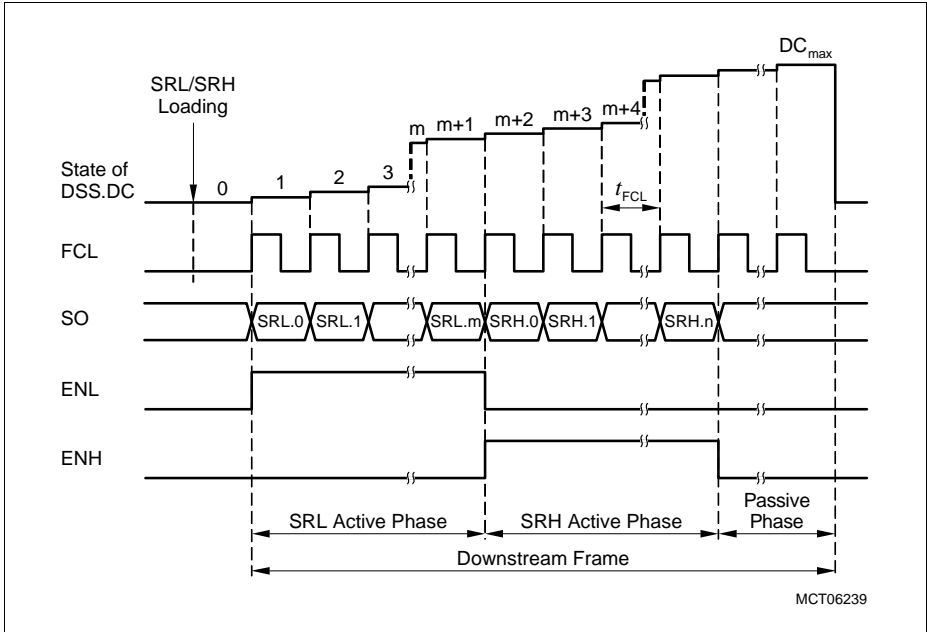


Figure 22-14 Shift Clock Counting: Data Frame with ENSELL = 0 and ENSELH = 0

22.1.2.6 Baud Rate

The baud rate of the downstream channel's serial transmission is defined by the frequency of the serial clock FCL, and is always $f_{MSC}/2$. The f_{MSC} generation is device specific and depends on the implementation of the MSC module. The TC26x B-Step specific clock generation is described on [Page 22-103](#).

22.1.2.7 Abort of Frames

Only a reset condition of the device can abort a current transmission. The MSC module does not start a new frame transmission when the downstream channel becomes disabled, the suspend mode is requested, or the sleep mode is entered. If one of these three conditions becomes active during a running frame transmission, the frame transmission is completely finished before the requested abort state is entered. Note that in this case no time frame finished interrupt is generated any more.

Micro Second Channel (MSC)

22.1.3 Upstream Channel

The MSC upstream channel is an asynchronous serial receiver based on the standard asynchronous data transfer protocol. It is dedicated to receive a serial data stream from a peripheral device via its serial data input SDI, using two specific data frame formats.

Figure 22-15 is a block diagram of the MSC upstream channel.

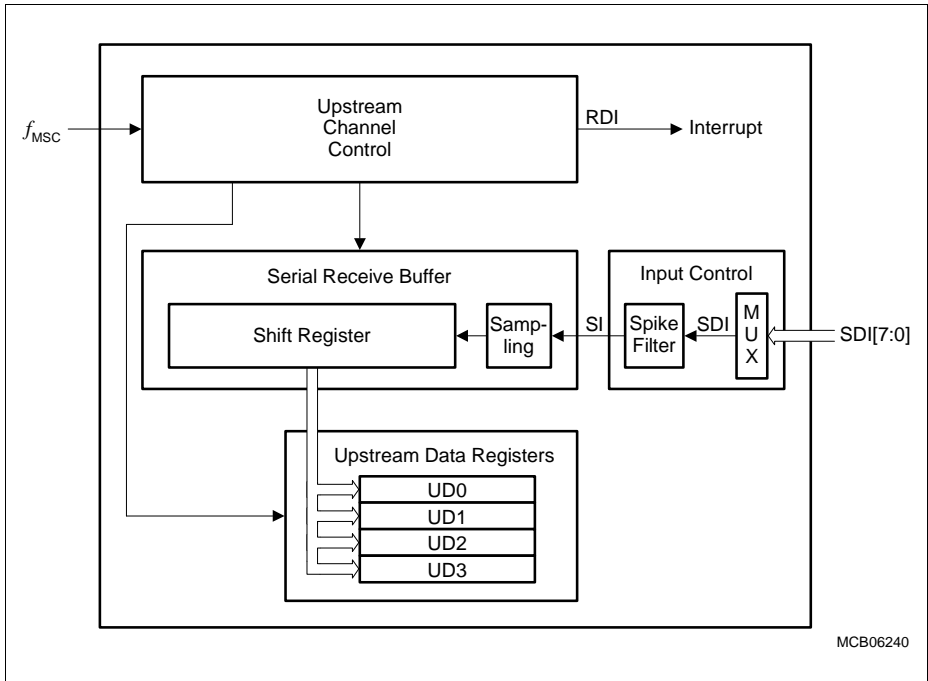


Figure 22-15 Upstream Channel Block Diagram

The incoming data at SI is sampled after it has been filtered for spikes. The detected logic states of the serial input are clocked into a shift register. After the complete reception of the serial data frame, the content of the shift register is transferred into one of the four data registers, and an interrupt can be generated optionally.

The reception baud rate is directly coupled to the module clock f_{MSC} , and can be within a range of $f_{MSC}/4$ up to $f_{MSC}/256$.

*Note: If the ABRA block is used with a fractional divider with a fractional ratio generating the upstream baud rate, and in order to compensate the additional fractional divider jitter, the reception baud rate must be less than $f_{MSC}/10$, or $f_{MSC}/(8 \text{ to } 256) * n/1024 < f_{MSC}/10$.*

22.1.3.1 Data Frames

The asynchronous data frames used by the upstream channel include four basic parts:

1. One start bit, always at low level
2. An 8-bit data field D[7:0] with LSB first
3. An optional 4-bit address field A[3:0] with LSB first
4. One parity bit and two stop bits, that are always at high level

As shown in [Figure 22-16](#), the 16-bit upstream data frame includes an additional 4-bit address field. The upstream frame type is selected by bit USR.UFT.

- USR.UFT = 0: 12-bit upstream data frame selected
- USR.UFT = 1: 16-bit upstream data frame with 4-bit address field selected

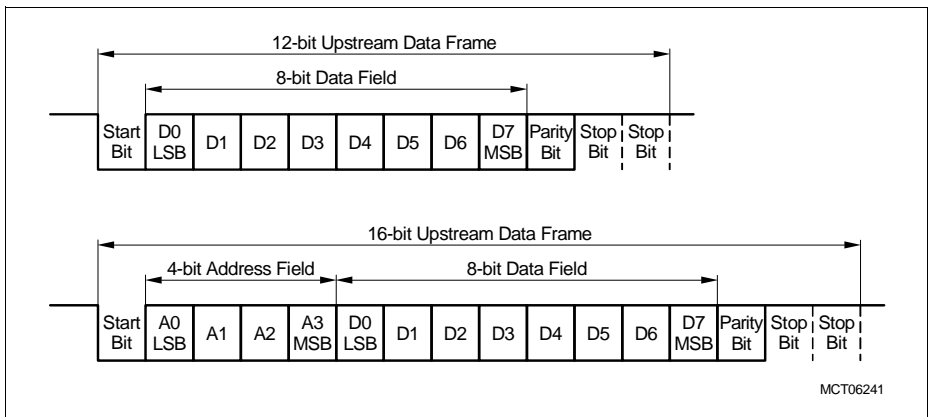


Figure 22-16 Upstream Channel Frame Types

22.1.3.2 Parity Checking

The incoming parity bit of the data frames can be checked by the upstream channel. When a parity error is detected, the parity error flag PERR in the related Upstream Data Register UDx is set. Note that a setting of the parity error flag PERR does not generate an interrupt. The PERR bits must be checked by software. The UDx registers also store the parity bit of the incoming data frame (UDx.P) and the parity bit that is generated internally (UDx.IPF).

Bit USR.PCTR determines the parity mode, even or odd, that is selected for parity checking. With USR.PCTR = 0, even parity mode is selected. Even parity means that the parity bit is set on an odd number of 1s in the data field (12-bit upstream data frame) or in the address plus data field (16-bit upstream data frame). With USR.PCTR = 1, odd parity mode is selected. In odd parity mode, the parity bit is set on an even number of 1s of the related data.

Micro Second Channel (MSC)

The parity checking logic in the upstream channel also controls whether start bit and the two stop bits of the upstream data frame are at correct logic level. If the start bit is not at low level and the two stop bits are not at high level at the end of the frame reception, the parity error flag UDx.PERR is set, too.

22.1.3.3 Data Reception

The reception of the upstream frame is started with a falling edge (1-to-0 transition) on the SI line. When the start bit is detected, serial reception is enabled and the receive circuit begins to sample the incoming serial data and to buffer it in the receive buffer. After the second stop bit has been detected, the content of the receive buffer is transferred to one of four upstream data registers UDx. The receive circuit then waits for the next start bit (1-to-0 transition) at the SI line. When the content of the receive buffer has been transferred to UDx, the valid bit UDx.V is set by hardware, and a receive interrupt can be generated.

The data reception mechanism does not involve destructive read mechanisms.

Note: The SI input line is the filtered non-inverted (OCR.ILP = 0) or inverted (OCR.ILP = 1) SDI input signal. The SI input signal selection is described on [Page 22-34](#).

Frame Reception with Address Field

Frame reception for a 16-bit data frame (see [Figure 22-17](#)) is selected by USR.UFT = 1. When the content of the receive buffer has been received completely, it is transferred to one of the four UDx registers. The two most significant address bits A[3:2] of the received 4-bit address field select the number x of register UDx in which the received frame content is stored. Register UDx is loaded with the two least significant address bits A0 and A1 (UDx.LABF), the 8-bit data (UDx.DATA), the received parity bit (UDx.P), the calculated parity bit (UDx.IPF), and the parity checking result (UDx.PERR). Finally, the valid bit UDx.V is set to indicate that the UDx register contains valid data.

The current state of the frame reception is indicated by the content of an upstream counter that is readable via bit field USR.UC. The upstream counter is a 5-bit counter that counts the upstream frame bits during reception. As shown in [Figure 22-17](#), the upstream counter is loaded with 10000_B at the detection of a start bit. It counts down and is again at 00000_B when the second stop bit has been detected and the frame reception is finished.

The state of the serial input data line SI is sampled in the middle of a bit cell and shifted into the receive buffer at the end of the bit cell. The frequency of the shift clock f_{SHIFT} depends the selected baud rate (see [Page 22-28](#)).

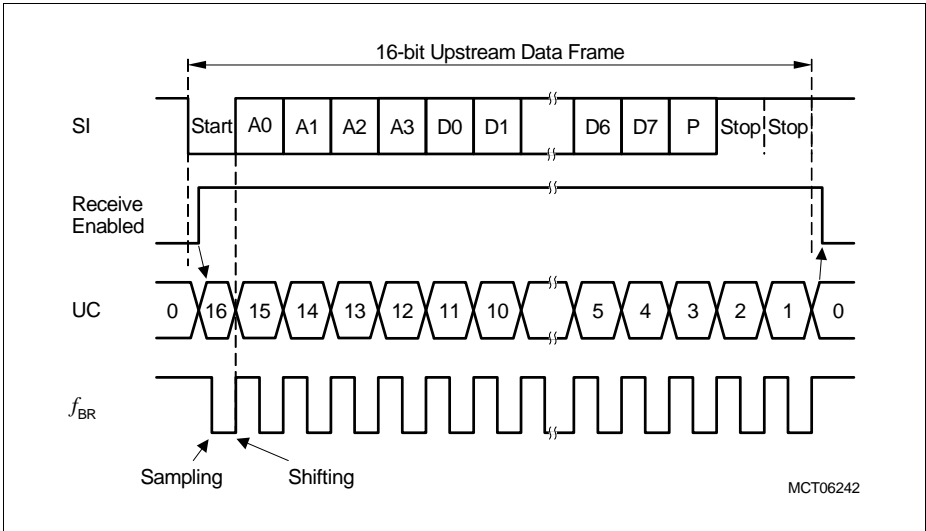


Figure 22-17 16-bit Upstream Reception

Data Reception without Address Field

Frame reception for a 12-bit data frame is selected by $USR.UFT = 0$. The reception scheme is comparable with that of the 16-bit data frame reception but there are a few differences:

- The upstream counter is initially loaded with 01100_B .
- The received frame content is always stored in register UD0.
- Bit field UD0.LABF is always loaded with 00_B when the frame is stored.

Micro Second Channel (MSC)

22.1.3.4 Baud Rate

The baud rate of the upstream channel is derived from the MSC module clock f_{MSC} . **Figure 22-18** shows the configuration of the upstream channel clock circuitry.

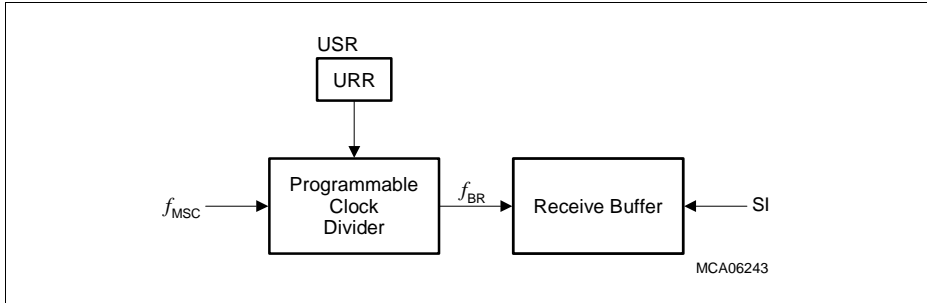


Figure 22-18 Upstream Channel Clock Circuitry

The serial data input SI is evaluated with the baud rate clock f_{BR} in the middle of each bit cell, and latched in case of a data bit. The baud rate clock f_{BR} is derived from f_{MSC} by a programmable clock divider. The frequency of f_{BR} determines the width of a received bit cell and therefore the baud rate for the received data. The content of bit field USR.URR selects the baud rate according **Table 22-7**. The resulting baud rate formula is:

$$\text{Baud rate}_{\text{MSC Upstream Channel}} = \frac{f_{\text{MSC}}}{\text{DF}} \quad (22.3)$$

Table 22-7 Upstream Channel Divide Factor DF Selection & Baud Rate

USR.URR	Divide Factor DF	Baud Rate
000 _B	reception disabled	–
001 _B	4	$f_{\text{MSC}}/4$
010 _B	8	$f_{\text{MSC}}/8$
011 _B	16	$f_{\text{MSC}}/16$
100 _B	32	$f_{\text{MSC}}/32$
101 _B	64	$f_{\text{MSC}}/64$
110 _B	128	$f_{\text{MSC}}/128$
111 _B	256	$f_{\text{MSC}}/256$

Note: With the USR.URR = 000_B the upstream channel is disabled and data reception is not possible.

Micro Second Channel (MSC)

The content of bit field USR.URR determines the operation of an internal sampling reload counter that is clocked with f_{MSC} . **Figure 22-19** shows the operation of the sampling counter at the beginning of an upstream frame with a divide factor DF of 8 (USR.URR = 010_B is equal to DF = 8) which means eight sampling clocks per each frame bit cell.

When the upstream channel is in idle state, it waits for a falling edge (1-to-0 transition) at SI. Therefore, the sample counter starts counting up and is reset when the selected divide factor DF as shown in **Table 22-7** is reached. In the middle of the sampling counter's count range, the logic state at SI is evaluated and, in case of a data bit, latched in the receive buffer's shift register. With the reload of the sampling counter, the shift register is shifted by one bit position.

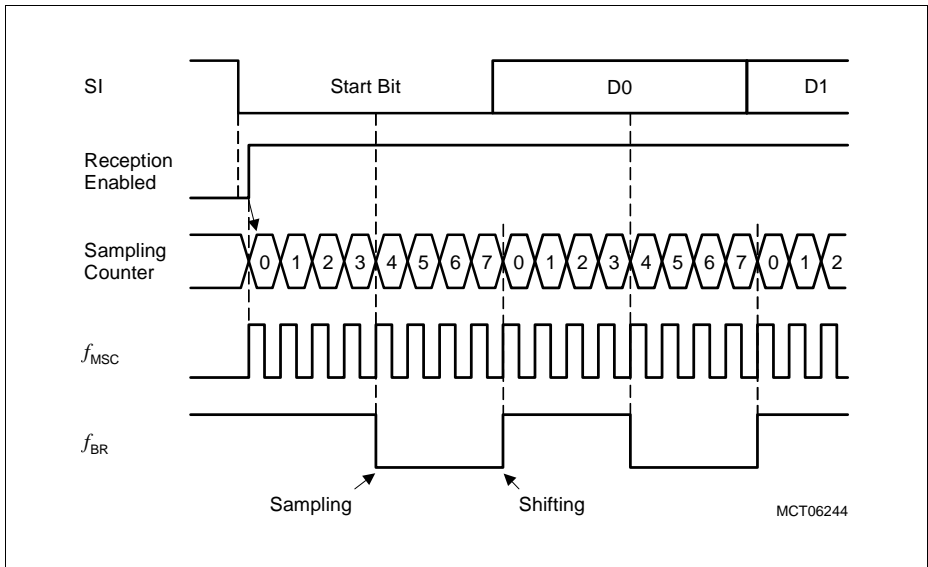


Figure 22-19 Upstream Channel Sampling with URR = 010_B

22.1.3.5 Spike Filter

The upstream channel input line SDI is sampled using a built-in spike filter with synchronization stage, both clocked with f_{MSC} . The spike filter is a chain of flip-flops with a majority decision logic (2 out of 3). A sampled value that is found at least twice in three samples is taken as data input value for SI.

22.1.3.6 Upstream Timer

The upstream timer monitors the time interval between two subsequent upstream frames. The timer reloads its starting values from the register **USCE** and starts counting downwards. If the time interval between two frames is longer than allowed, the counter reaches the value of zero, raises an upstream timer interrupt and stops counting. The counter reloads and starts counting on four events:

- Enabling of the MSC module after reset via **CLC.DISR**
- Resetting the module
- Writing the bit fields **USCE.USTOPRE** or **USCE.USTOVAL** per software
- Detecting the falling edge of the start bit of a frame

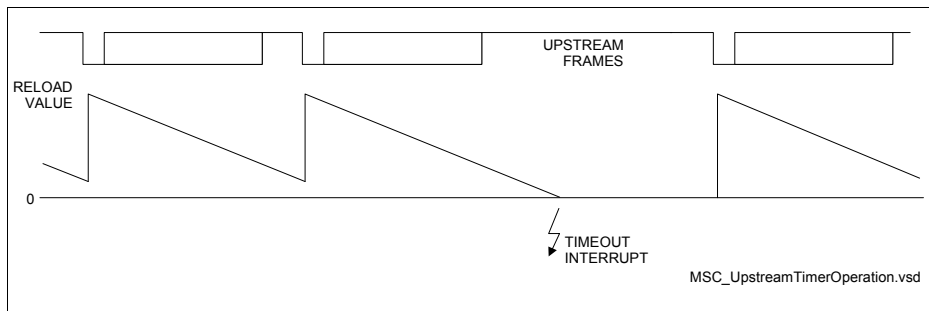


Figure 22-20 Upstream Timer Operation

The upstream watchdog timer counts bit times. The time window in which the next upstream frame must occur can be calculated according to the formula:

$$\text{TimeOut} = \text{BitTime} * 2^{(\text{USTOPRE}+1)} * (\text{USTOVAL}+1)$$

22.1.4 I/O Control

The types of I/O control logic for the MSC module I/O lines are shown in [Figure 22-21](#). The downstream channel generates five output signals that control eight MSC module outputs, split into four chip select outputs, two clock outputs, and two serial data outputs. The upstream channel has one input signal.

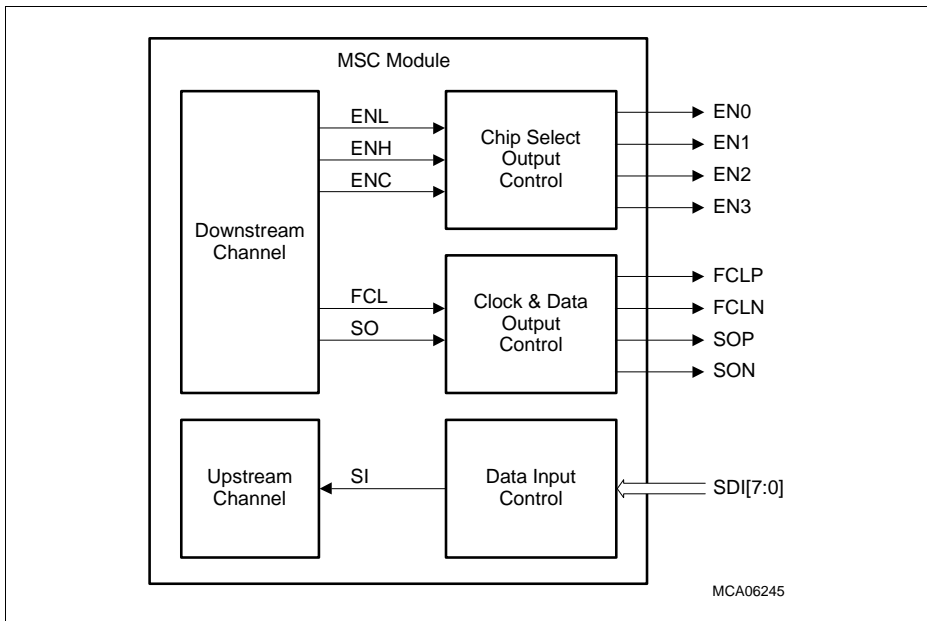


Figure 22-21 I/O Control

The MSC module I/O signals is controlled by bit fields that are located in the Output Control Register OCR.

22.1.4.1 Downstream Channel Output Control

As shown in [Figure 22-5](#) and [Figure 22-6](#), the active phases during downstream channel operation are indicated by three enable signals:

- ENL indicates the SRL active phase of a data frame
- ENH indicates the SRH active phase of a data frame
- ENC indicates the active phase of a command frame

The chip select output control logic of the MSC uses a signal compressing scheme (similar to the interrupt request compressing scheme in [Figure 22-29](#)) that allows each of the three enable signals to be directed via a 2-bit selector to one of the four chip enable

Micro Second Channel (MSC)

outputs EN[3:0]. This also makes it possible to connect more than one internal enable signal (ENL, ENH, ENC) to one chip enable output ENx. Three bit fields in register OCR (CSL, CSH, and CSC) determine which chip enable output becomes active on a valid internal enable signal.

In the MSC, enable signals are high-level active signals. If required in a specific application, all chip enable outputs ENx can be assigned for low-level active polarity by setting bit OCR.CSLP.

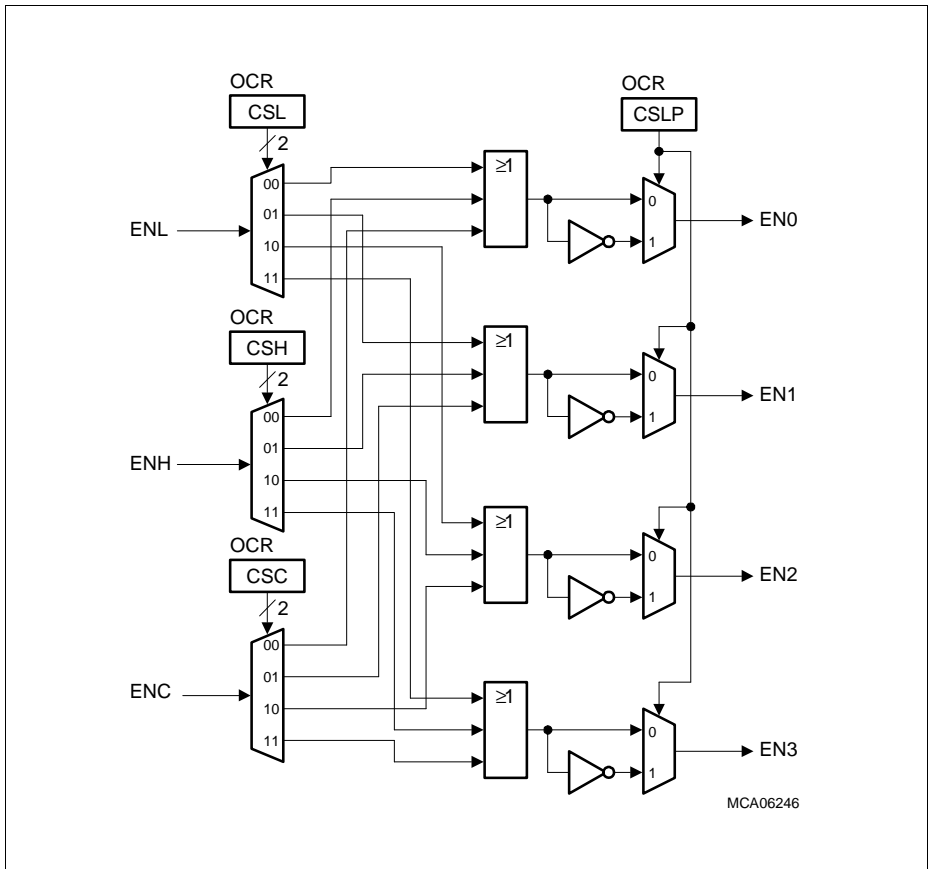


Figure 22-22 Downstream Channel: Chip Enable Output Control

Micro Second Channel (MSC)

At the MSC downstream channel, the internal serial clock output FCL and data output line SO are available outside the MSC module as two signal pairs with inverted signal polarity, FCLP/FCLN and SOP/SON. Both, clock and data outputs, are generated from the module internal signals FCL and SO according to [Figure 22-23](#).

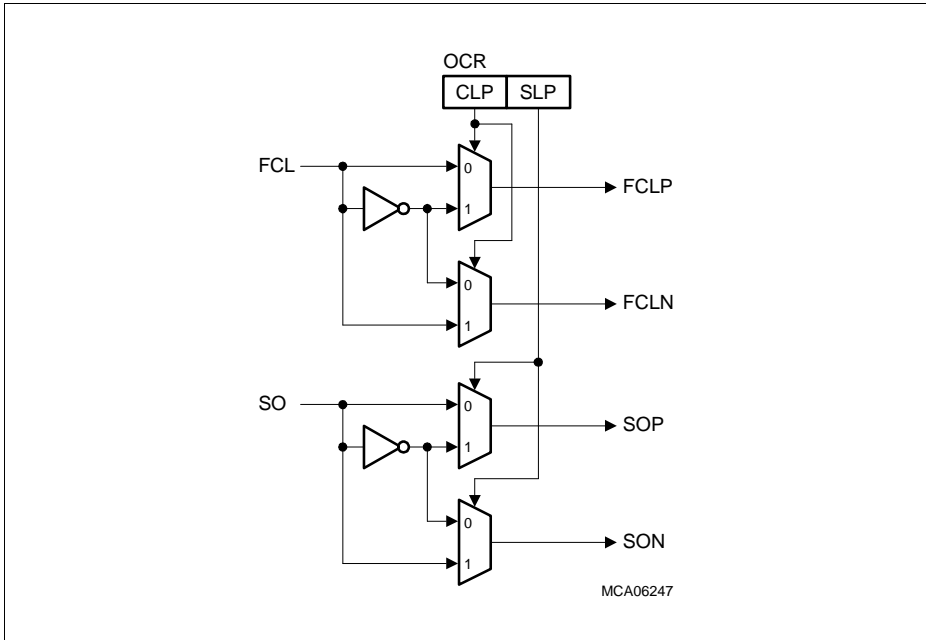


Figure 22-23 Downstream Channel: Clock and Data Output Control

With `OCR.CLP = 0`, FCLP has identical and FCLN has inverted polarity compared to FCL. Setting `OCR.CLP`, exchanges the signal polarities of FCLP and FCLN. An equivalent control capability is available for the SOP and SON data outputs (controlled by `OCR.SLP`).

One additional control capability not shown in [Figure 22-23](#) is available for the FCL signal. With `OCR.CLKCTRL = 1`, the FCL clock signal will always be generated, independently whether a downstream frame is currently transmitted or not. If `OCR.CLKCTRL = 0`, FCL becomes only active during the active phases of data or command frames (not during passive time frames).

22.1.4.2 Upstream Channel

As shown in **Figure 22-24**, the MSC upstream channel can be connected to up to eight SDI[7:0] serial inputs. Bit field OCR.SDISEL selects one out of these input lines (input signal SDI). If OCR.ILP = 0, SDI is directly connected to the serial receive buffer input SI. If OCR.ILP = 1, SDI is connected to input SI via an inverter.

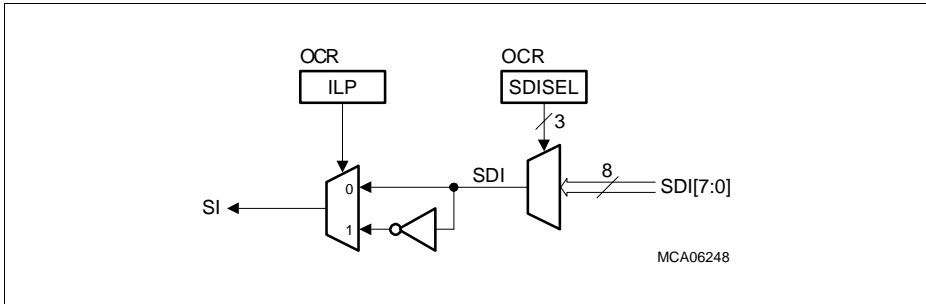


Figure 22-24 Upstream Channel Serial Data Input Control

22.1.5 MSC Interrupts

The MSC module has seven interrupt sources and five service request outputs. A service request output is able to generate interrupts (controlled by a service request control register) or DMA requests. The service request output assignment, interrupt or DMA request, is specific for each microcontroller that is using the MSC. In this section, the term “interrupt request” has the meaning of “service request” that is able to handle interrupt or DMA requests.

Each interrupt source is provided with a status flag, enable bit(s) with software set/clear capability, and an interrupt node pointer. An interrupt event, internally generated as a request pulse, is always stored in an interrupt status flag that is located in the Interrupt Status Register ISR. All interrupt status flag can be set or cleared individually by software via the interrupt Set Clear Register ISC. Software-controlled interrupt generation can be initiated by setting the interrupt status flag of the corresponding interrupt. Each interrupt source can be enabled or disabled individually. When an interrupt event is enabled, a 2-bit interrupt node pointer determines which of the service request outputs will be activated. See [Figure 22-29](#).

[Table 22-8](#) shows the seven MSC interrupt sources.

Table 22-8 MSC Interrupt Sources

Interrupt Type	Generated by
Data frame interrupt	Downstream Channel
Command frame interrupt	
Time frame finished interrupt	
Receive data interrupt	Upstream Channel
Upstream Timeout Interrupt	
Sync FIFO overflow	ABRA Block
Sync FIFO underflow	

22.1.5.1 Data Frame Interrupt

A data frame interrupt can be generated when either the first or the last data bit of the downstream channel is shifted out and becomes available at the SO output line (see also [Figure 22-6](#)). Bit ICR.EDIE selects which case is selected.

Note: If ICR.EDIE = 10_B, an interrupt at the first data bit is only generated if DSC.NDBL is not equal 00000_B. This means, at least one SRL bit must be shifted out for the first data bit shifted interrupt to become active.

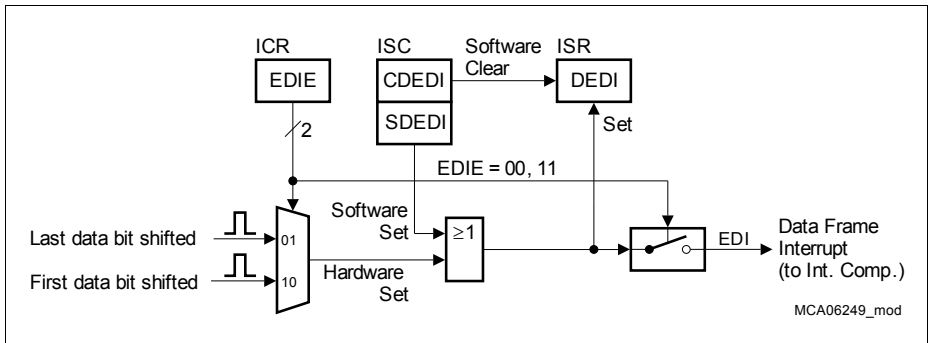


Figure 22-25 Data Frame Interrupt Control

22.1.5.2 Command Frame Interrupt

A command frame interrupt can be generated at the end of a downstream channel command frame (see also [Figure 22-5](#)).

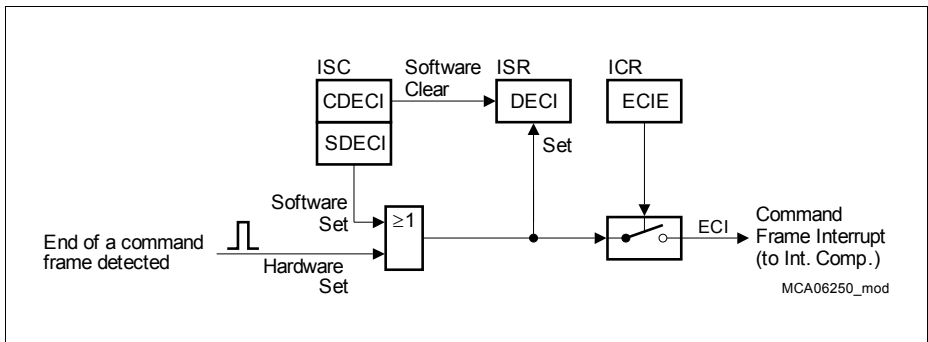


Figure 22-26 Command Frame Interrupt Control

22.1.5.3 Time Frame Finished Interrupt

A time frame finished interrupt can be generated at the end of a downstream channel passive time phase.

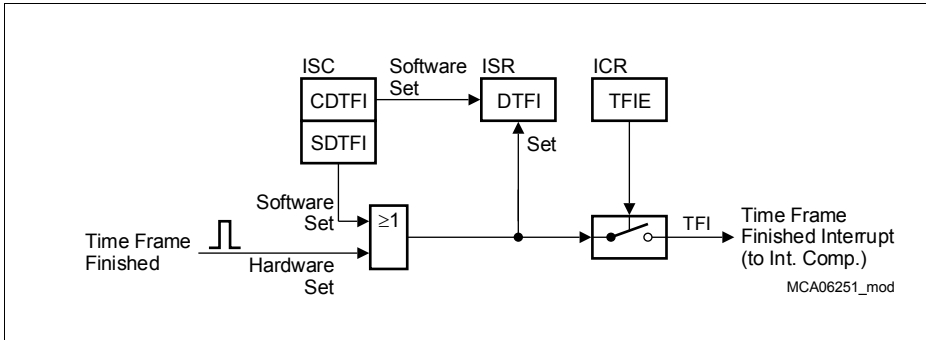


Figure 22-27 Time Frame Interrupt Control

22.1.5.4 Receive Data Interrupt

Whenever the upstream channel receives data in registers UDx (x = 0-3), the MSC is able to generate an interrupt. Three interrupt generation conditions can be selected for the receive data interrupt:

- Each update of UDx (x = 0-3) generates a receive data interrupt.
- Each update of UDx (x = 0-3) generates a receive data interrupt when the updated value is not equal 00_H.
- Only an update of register UD3 generates a receive data interrupt.

The selection of the interrupt generation condition is controlled by bit field ICR.RDIE. Setting ICR.RDIE = 0 disables the receive data interrupt in general. ISR.URDI is the interrupt status flag that can be set or clear when writing bits ISC.CURDI or ISC.SURDI with a 1. If the fractional divider is used, then the software should clear the URDI flag by using ISC.CURDI with a delay of one bit time after the interrupt signal has been generated. The same applies to clearing the UDx.V flag via UDx.C. Otherwise, due to the latencies introduced by the fractional divider, the clear may fail.

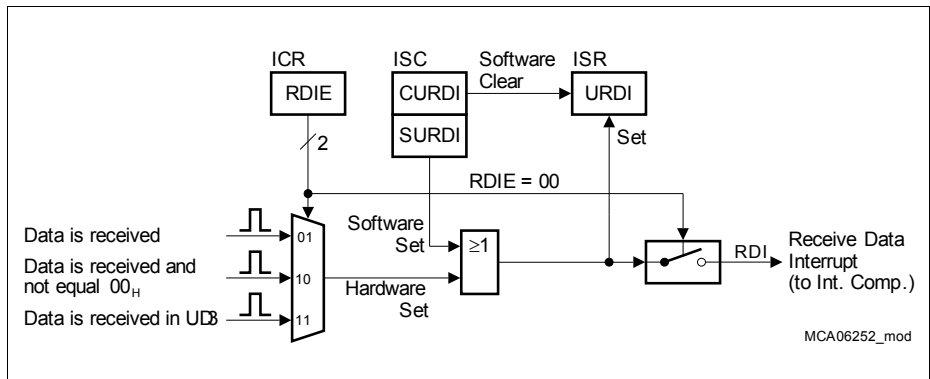


Figure 22-28 Receive Data Interrupt Control

The upstream receive interrupt can be optionally delayed for 1 bit time by setting the **USR.SRDC** bit.

Micro Second Channel (MSC)

22.1.5.5 Interrupt Request Compressor

The interrupt control logic of the MSC uses a flexible interrupt compressing scheme. Seven interrupt sources can be directed via a 2-bit interrupt node pointer to one of the four service request outputs SR[3:0], and three of them additionally to SR4. This makes it possible to connect more than one interrupt source to one interrupt output SRx.

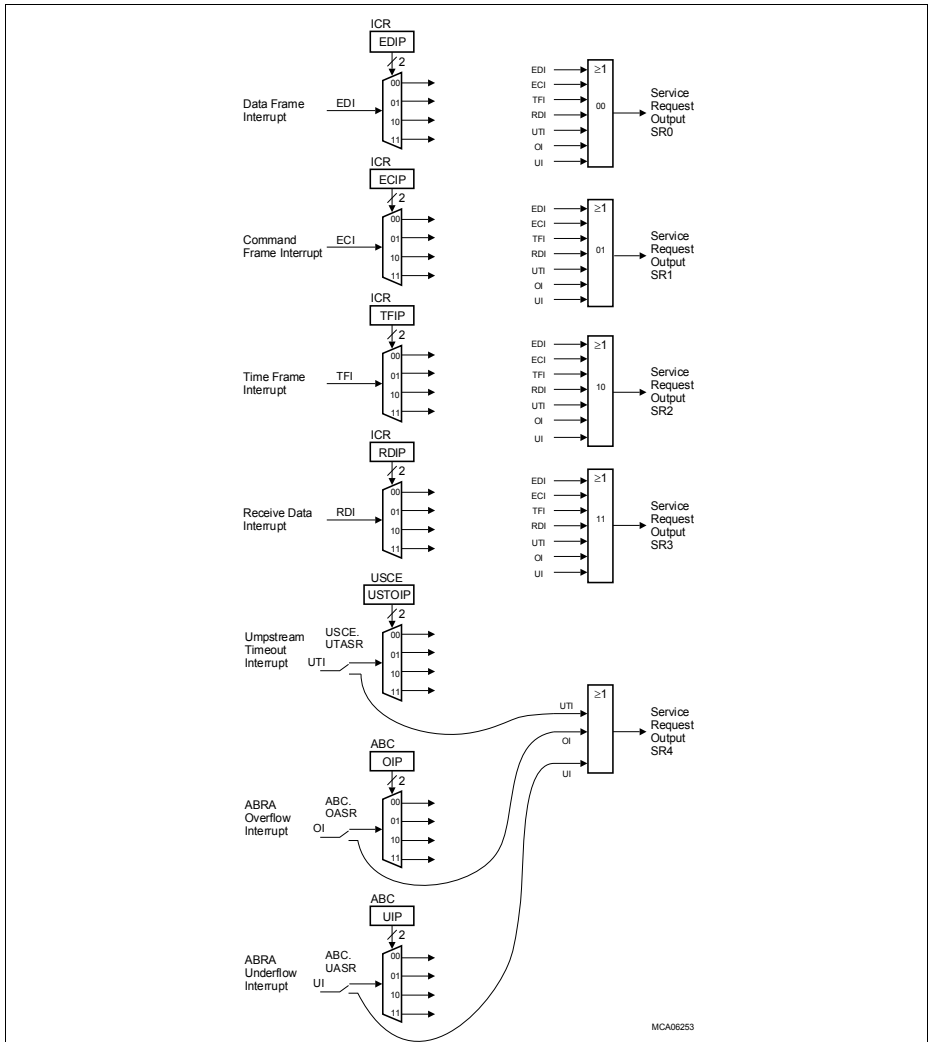


Figure 22-29 MSC Interrupt Request Compressor

22.2 ABRA (Asynchronous Baud Rate Adjustment Block)

The Asynchronous Baud Rate Adjustment Block (ABRA) takes as an input a serial MSC stream with one baud rate and outputs the same serial stream with another baud rate, asynchronous to the input one. The input signal bundle consists of serial clock, data and select signal, and the output signal bundle consists of the same signals.

Additionally, the output clock of an MSC without ABRA can be only set to 50% duty cycle, while with the ABRA block, other duty cycles are also possible.

22.2.1 Overview

The ABRA block is located between the MSC kernel and the output pins.

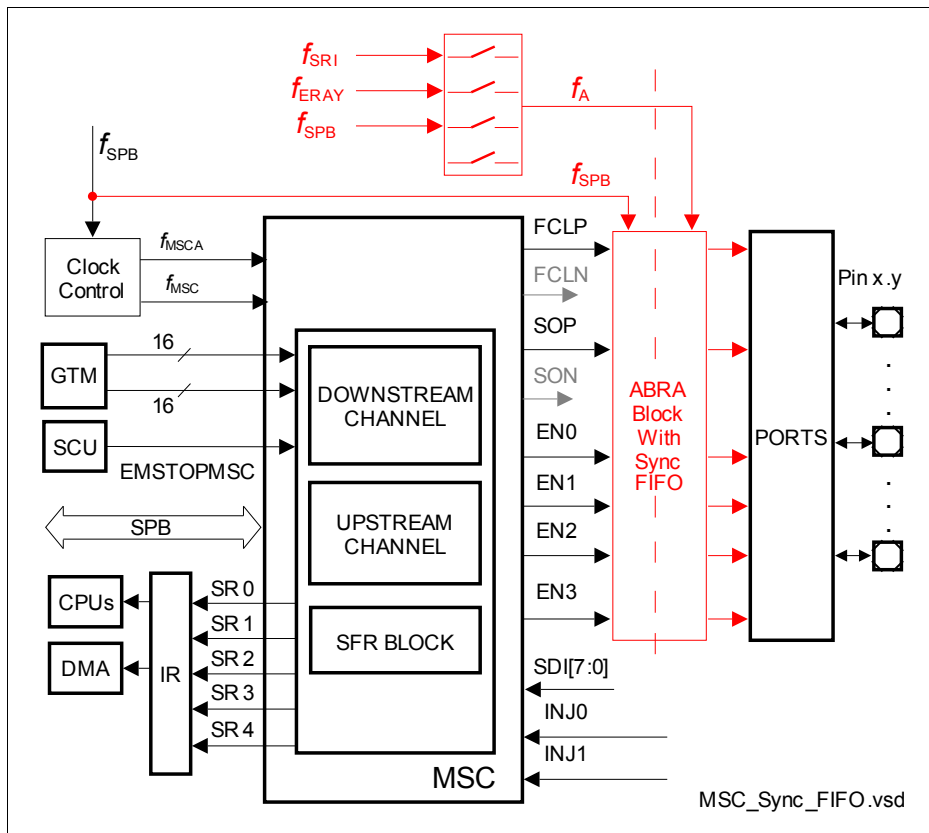


Figure 22-30 ABRA Overview

Micro Second Channel (MSC)

The user has two possibilities: either to use the ABRA block, or to bypass it. In case of using ABRA, additional delay in the signal path between the module and the pins of up to three f_A periods is to be taken into account. Additionally, some margin between the input frame length and the output frame length must be guaranteed by the customer, in order to avoid overflow of the ABRA block.

If the ABRA is not used, the behavior of the MSC module is identical with the previous implementations. The user can also choose between using the 64-bit extension or not, which makes 4 combinations in total, as shown in **Figure 22-31**

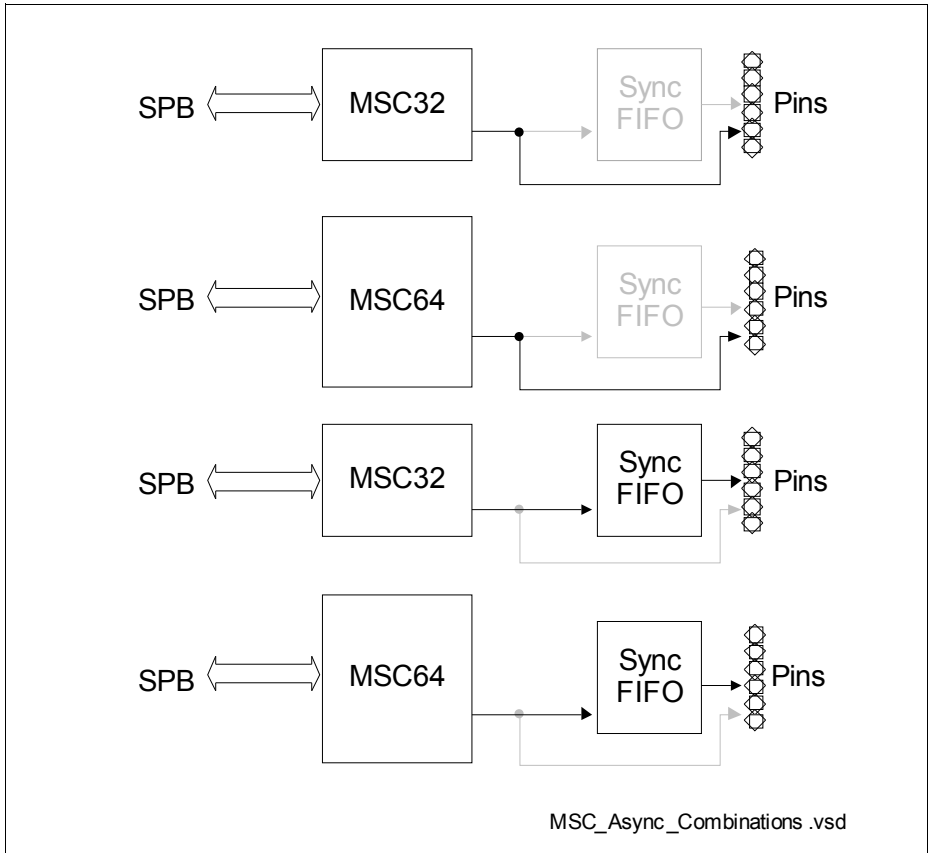


Figure 22-31 Use-Case Combinations

All configuration parameters related to the ABRA block are located in the ABC (Asynchronous Block Configuration) register.

22.2.2 Timings Issues

The ABRA block introduces a delay in the signal path from the MSC module to the pins. However it does not influence significantly the timings between the signals themselves. The output delays between the shift clock, data and enable signals remain approximately the same as if the ABRA block is not used, except for small differences that may appear due to the different routes that the signals take inside the chip, but these differences are included in the one common timing of the MSC module.

22.2.3 Adjusting the Passive Phase of a Frame

The main purpose of the Asynchronous Baud Rate Adjustment block is to transform a frame going in with a higher baud rate (for example 50Mbaud), to a frame going out with a lower baud rate (for example 40Mbaud). Therefore, the duration of the outgoing frame is longer then the duration of the incoming frame. In order not to overflow the ABRA block with too many incoming frames, there must be a pause between them, which is adjusted by configuring the passive phase of the frames. This pause plus the incoming frame duration must be longer or equal to the outgoing frame duration plus some margin.

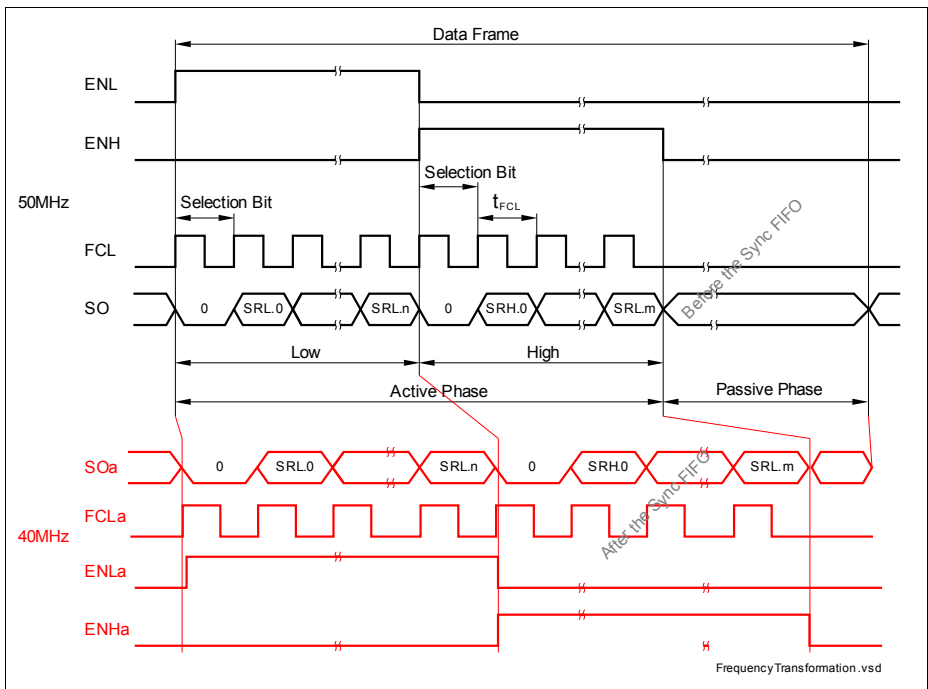


Figure 22-32 Passive Phase Adjustment

Micro Second Channel (MSC)

In order to allow for frame length adjustments:

- for command frames, the bit field **DSTE.PPCE** defines a programmable passive phase in a range of 2 to 65 input frequency cycles, and
- for data frames, the bit field **DSTE.PPDE** extends the passive phase from 0 to 32 to 0 to 128 bit times.

The length of the frame in the MSC domain (data and passive bits) must be longer than the length of the frame in the asynchronous domain, in order to prevent overload in case of back to back transmission:

$$(N_D + N_P) * T_{FCL} > (N_D + N_{PA}) * T_{FCLA}$$

which results to the the following inequality for setting the length of the passive phase in the MSC domain:

$$N_P > [N_D * (T_{FCLA} - T_{FCL}) + N_{PA} * T_{FCLA}] / T_{FCL}$$

Where:

T_{FCL} :

serial clock period in the MSC domain with activated ABRA: $T_{FCL} = 1 / \text{BaudRate}_{\text{MSCA}}$, $\text{BaudRate}_{\text{MSCA}} = f_{\text{SPB}} / (2^m)$, see **Intermedate Downstream Channel Baud Rate** with ABRA block.

T_{FCLA} :

serial clock period of the ABRA block $T_{FCLA} = 1 / \text{BaudRate}_{\text{ABRA}}$, $\text{BaudRate}_{\text{ABRA}} = f_A / (\text{ABC.NDA} * (\text{ABC.LOW}+1 + \text{ABC.HIGH}+1))$

N_P : number of the passive bits in the MSC domain (see **DSTE.PPDE** and **DSC.PPD**)

N_{PA} : number of the targeted passive bits in the asynchronous domain

N_D : number of data bits **DSC.NDBH+NDBL+SELH+SELL+DSCE.NDBLE+NDBHE** in both domains.

Note: If the passive phase settings in the MSC module are not correctly configured in such a way that two frames merge, that is, there is no passive phase on the pins at all, an overflow interrupt is triggered. If the passive phase is wrongly shorter, no error signal is generated.

Input/Output Baud Rate Ratio

The ABRA output baud rate going to the pins must be lower than the input baud rate coming from the MSC module and higher then roughly half of the input baud rate (more precisely higher than 57% of the input baud rate in worst-case, due to the fact that in worst-case MSC data frame can contain 64 data bits plus additionally two select bits).

22.2.4 Jitter of the Downstream Frames

This section describes the jitter effects affecting the starting point of the downstream frames, depending on the setting of the serial clock - serial clock only during the active phase of a frame, or continuous clock.

Note: If the asynchronous frequency f_A is a multiple of f_{SPB} , derived from the same PLL, there is no jitter of the downstream frames, only delay. This applies to the cases where f_A is f_{SPB} or f_{SRI} (for example 200MHz or 300MHz).

22.2.4.1 Jitter in Active Phase Clock Mode

The active phase clock mode is set when **OCR.CLKCTRL** = 0. In this mode the serial clock is generated only during the transmission of the data bits. The ABRA outgoing frame is delayed relative to the incoming frame for three to four f_A cycles. This implies passive phase duration jitter of +/- one f_A cycle. The example in the **Figure 22-33** assumes 50MBaud input and 40MBaud output baud rate, $f_A = 80\text{MHz}$, back to back frames.

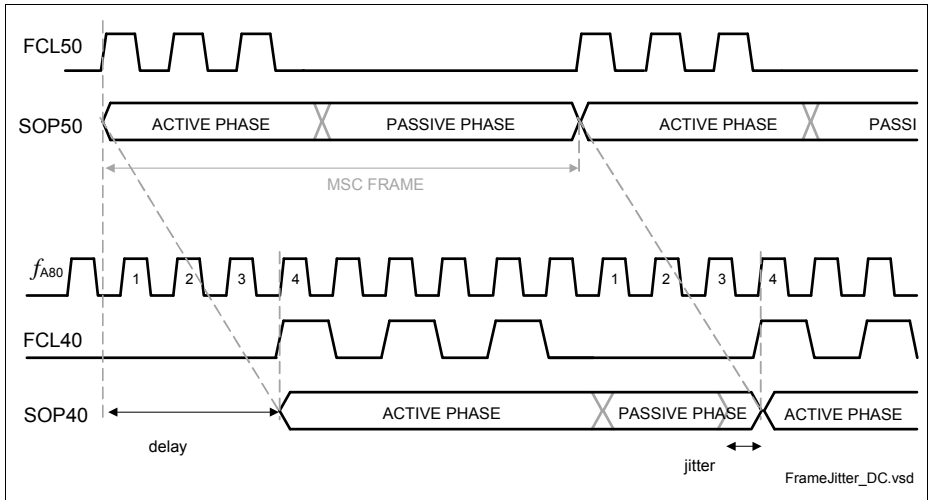


Figure 22-33 Frame Jitter in Active Phase Clock Mode

22.2.4.2 Jitter in Continuous Clock Mode

Continuous clock mode is set when **OCR.CLKCTRL** = 1. In this mode the serial clock is always on. The ABRA output frame is delayed relative to the input frame first for two to three f_A cycles, and then it is shifted out with the first available outgoing rising edge of the serial clock. This implies total delay of three to four f_A cycles plus zero to one half output bit time, and a jitter of one output bit time.

The average length of the output frames is equal to the length of the input frames. The ABRA block generates an averaging effect similar to that of a fractional divider. It introduces occasionally jitter of one bit time in the output frame length, in the passive phase of the frame, in order to adjust the average output time raster to the ideal input time raster. The granularity of the input time raster is one input bit time (for example 20ns at 50MBaud), the granularity of the output time raster is one output bit time (for example 25ns at 40MBaud).

The example in the **Figure 22-34** assumes 50MBaud input baud rate and 40MBaud output baud rate, $f_A = 80\text{MHz}$, back to back frames.

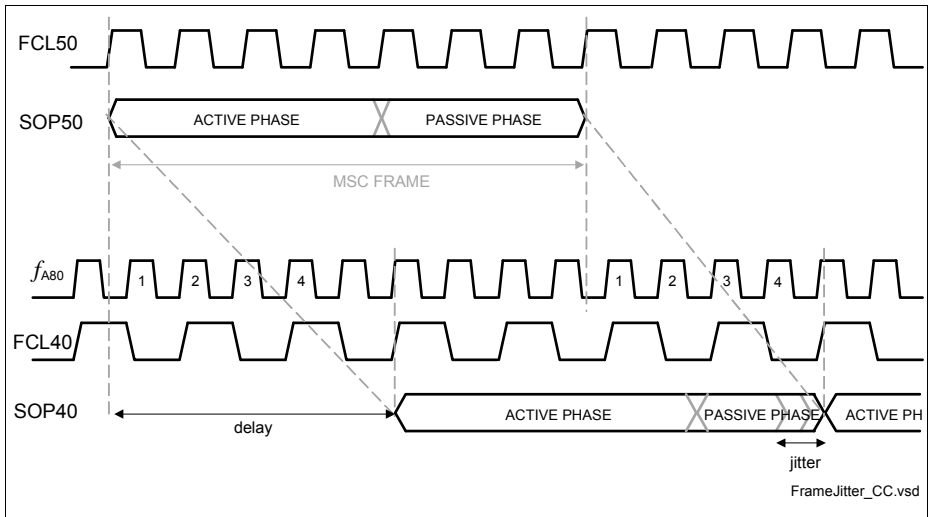


Figure 22-34 Frame Jitter in Continuous Clock Mode

22.2.5 Interrupt Position with the ABRA Block

The position of the MSC interrupts within a frame depends only on the internal time schedule of the MSC module in both use cases with and without ABRA block. Therefore the interrupt position in the internal time raster is constant and is the same as in the previous versions of the MSC module, and does not depend on the asynchronous baud rate which is seen on the pins. This behavior affects only the downstream interrupts, which are:

- Data Frame Interrupt
- Command Frame Interrupt
- Time Frame Finished Interrupt

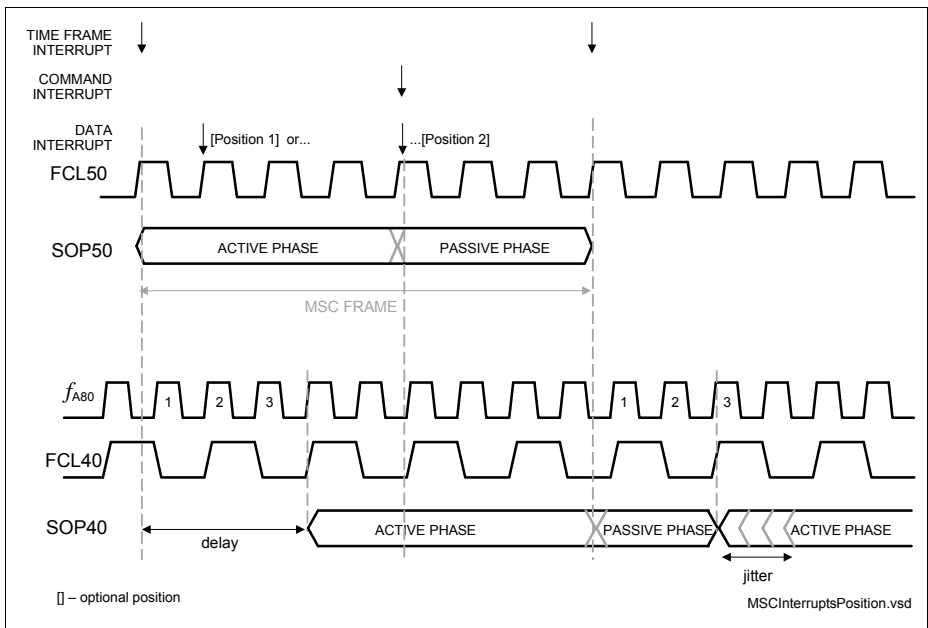


Figure 22-35 Interrupt Position with the ABRA Block

The downstream interrupts can be used for updates of the shadowed rw bit fields for the next frame transmission. Such bit fields are located in the Data Register **DD**, Command Register **DC**, Data Control **DSC** and Data Status **DSS**. Due to the next-frame-update effect, the small jitter and delay effects of the frames as seen on the pins is not relevant, as well as the position of the interrupt in the asynchronous frame.

22.2.6 Configuring the ABRA block

This section describes all the configurable parameters of the ABRA block. They are located in the **ABC** (Asynchronous Block Configuration) register.

Output Baud Rate

The ABRA block shifts out the data with a baud rate generated by dividing the clock f_A in the range of 2 to 32.

The high and the low time of the shift clock are generated by a separate 4-bit n-divider, each operating in the range of 1 to 16. Therefore the duty cycle can be finely configured to be 50% or any other ratio. This may provide some advantages when the communicating devices have asymmetrical timings (output delays or set-up/hold times). See the description of the bit-fields **ABC.HIGH** and **ABC.LOW**

The output baud rate of the ABRA block is defined with the formula:

$$\text{Baud Rate} = f_A / (\text{ABC.NDA} * (\text{ABC.LOW} + 1 + \text{ABC.HIGH} + 1))$$

Continuous Shift Clock

As defined in the bit field **ABC.CC**, the shift clock signal can be active either:

- only during the data transmission time
- continuously

Overflow and Underflow Interrupts

If the input and the output timings of the ABRA block are not adjusted to each other, an overflow of the ABRA block will occur and an error interrupt will be triggered on line X.

Overflow occurs either if the input data rate is too high or the output data rate is too low.

The corresponding bit fields are **ABC.OF**, **OFM**, **OIE**.

Underflow occurs either if the input data rate is too low or the output data rate is too high.

The corresponding bit fields are **ABC.UF**, **UFM**, **UIE**

See also **Interrupt Request Compressor**.

This feature can not guarantee protection against software errors like reconfiguring the baud rates in the middle of a frame.

The configuration of the ABRA block and the MSC kernel is static and the time behavior is deterministic and calculable. Therefore, an overflow/underflow event always signals some configuration error resulting in unadjusted input/output baud rate ratio (see **Input/Output Baud Rate Ratio**). In normal operation and with correctly configured baud rates (clocks and clock dividers) these events do not occur.

Micro Second Channel (MSC)

In case of underflow/overflow event the output frame will be corrupt. The simplest solution to a sporadic event caused by a transient (noise, alpha particle) is to reset the module, reconfigure and restart the communication.

Bypass Mode

If the ABRA block is not used, it is disabled and bypassed via the bit field **ABC.ABB**.

22.2.7 Implementation Issues

The ABRA block operates using synchronization FIFO. The MSC module writes its serial stream to the synchronization FIFO with its serial shift clock, and the same data is read out of the FIFO by using the ABRA serial shift clock, generating out of the f_A .

Three signals are being transformed: the serial shift clock, the serial data stream and the enable (chip select) signals.

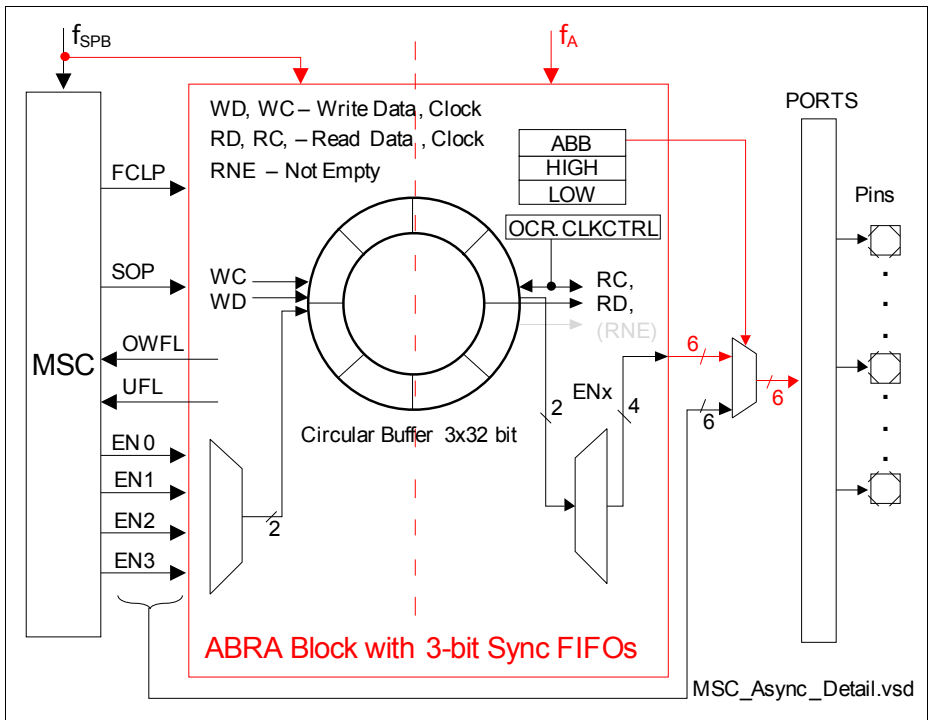


Figure 22-36 Implementation Details of the ABRA Block

22.2.8 ABRA Disable, Sleep and Suspend Behavior

This section describes the behavior of the MSC module in case of:

- switching the module on/off - disable behavior
- power saving - sleep mode
- debugging - OCDS suspend mode

The main concept is to provide as similar behavior as the behavior of the MSC module without the ABRA block, as much as possible.

22.2.8.1 Disable and Sleep Behavior

There is no difference in the behavior of the module in case of entering / leaving the disable and sleep state. Disable state is controlled by a bit set / cleared by software, sleep state is controlled by a hardware signal which can be disabled or enabled.

On disable/sleep state request, the module finishes the possibly ongoing frames (downstream or upstream), then acknowledges the request and the module clock is switched off.

On leaving the disable/sleep state, the module continues with the operation at the same point where it entered the corresponding state.

22.2.8.2 OCDS Suspend Behavior

In case of Hard Suspend, the clock is switched off immediately. A running frame will not be finished.

In case of Soft Suspend, the state is entered after the following two conditions are met:

- any ongoing upstream reception is completed, and
- any ongoing downstream frame transmission is completed, both from the MSC kernel and the ABRA block side

22.3 MSC Kernel Registers

This section describes the kernel registers of the MSC module. All MSC kernel register names described in this section will be referenced in other parts of the TC26x B-Step User's Manual by the module name prefix "MSCx_".

All registers in the MSC address spaces are reset with the application reset (definition see SCU section "Reset Operation").

MSC Kernel Register Overview

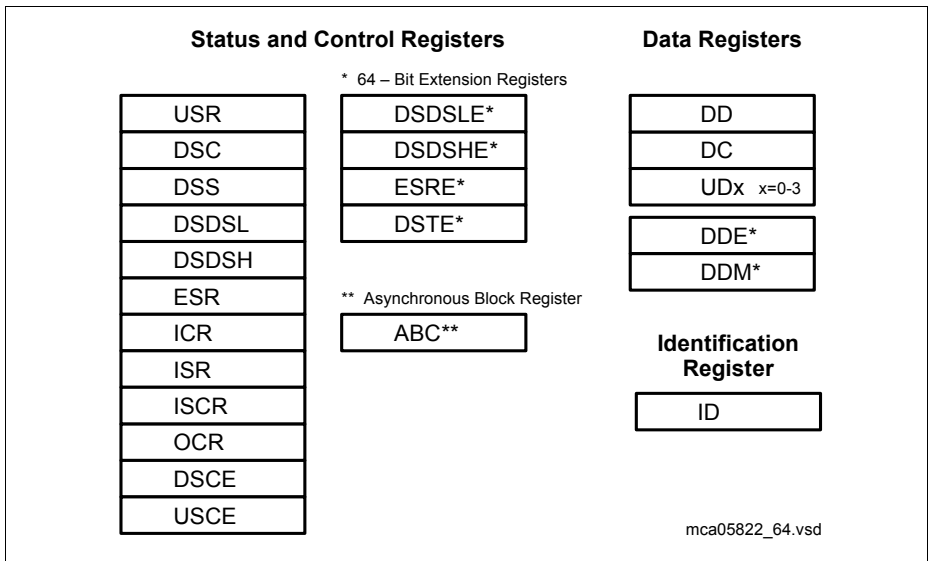


Figure 22-37 MSC Kernel Registers

Table 22-9 Registers Address Space

Module	Base Address	End Address	Note
MSC0	F0002600 _H	F00026FF _H	–
MSC1	F0002700 _H	F00027FF _H	–

Table 22-10 Registers Overview

Register Short Name	Register Long Name	Offset ¹⁾ Address	Write ²⁾ Access	Description see
ID	Module Identification Register	08 _H	BE	Page 22-53
FDR	Fractional Divider Register	0C _H	SV, E, P	Page 22-10 8
USR	Upstream Status Register	10 _H	U, SV, P	Page 22-54
DSC	Downstream Control Register	14 _H	U, SV, P	Page 22-56
DSS	Downstream Status Register	18 _H	U, SV, P	Page 22-63
DD	Downstream Data Register	1C _H	U, SV, P	Page 22-81
DC	Downstream Command Register	20 _H	U, SV, P	Page 22-81
DSDSL	Downstream Select Data Source Low Register	24 _H	U, SV, P	Page 22-65
DSDSH	Downstream Select Data Source High Register	28 _H	U, SV, P	Page 22-66
ESR	Emergency Stop Register	2C _H	U, SV, P	Page 22-67
UD0	Upstream Data Register 0	30 _H	U, SV, P	Page 22-83
UD1	Upstream Data Register 1	34 _H	U, SV, P	
UD2	Upstream Data Register 2	38 _H	U, SV, P	
UD3	Upstream Data Register 3	3C _H	U, SV, P	
ICR	Interrupt Control Register	40 _H	U, SV, P	
ISR	Interrupt Status Register	44 _H	U, SV, P	Page 22-71
ISC	Interrupt Set Clear Register	48 _H	U, SV, P	Page 22-73
OCR	Output Control Register	4C _H	U, SV, P	Page 22-75
DSCE	Downstream Control Enhanced Reg.	58 _H	U, SV, P	Page 22-59
USCE	Upstream Control Enhanced Register	5C _H	U, SV, P	Page 22-61
Extension Registers				
DSDSLE	Downstream Select Data Source Low Extension Reg.	60 _H	U, SV, P	Page 22-78
DSDSHE	Downstream Select Data Source High Extension Register	64 _H	U, SV, P	Page 22-79
ESRE	Emergency Stop Extension Register	68 _H	U, SV, P	Page 22-85
DDE	Downstream Data Extension Register	6C _H	U, SV, P	Page 22-87

Micro Second Channel (MSC)
Table 22-10 Registers Overview (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾ Address	Write ²⁾ Access	Description see
DDM	Downstream Data Mirror Register	70 _H	U, SV, P	Page 22-87
DSTE	Downstream Timing Extension Reg.	74 _H	U, SV, P	Page 22-80
Asynchronous Block Register				
ABC	Asynchronous Block Configuration Register	80 _H	U, SV, P	Page 22-88
BPI Registers				
CLC	Clock Control Register	00 _H	SV, E, P	Page 22-94
OCS	OCDS Control and Status Reg.	E8 _H	SV, P	Page 22-95
KRSTCLR	Reset Status Clear Register	EC _H	SV, E, P	Page 22-101
KRST1	Reset Control Register 1	F0 _H	SV, E, P	Page 22-100
KRST0	Reset Control Register 0	F4 _H	SV, E, P	Page 22-98
ACCEN1	Access Enable Register 1	F8 _H	SV, SE	Page 22-97
ACCEN0	Access Enable Register 0	FC _H	SV, SE	Page 22-96

1) The absolute register address is calculated as follows:

Module Base Address ([Table 22-9](#)) + Offset Address (shown in this column)

2) All registers have read access mode U, SV. Read access to reserved addresses delivers bus error (BE) except 50H and 54H.

All registers belong to reset class 3 except OCS, which belongs to reset class 1.

22.3.1 Module Identification Register

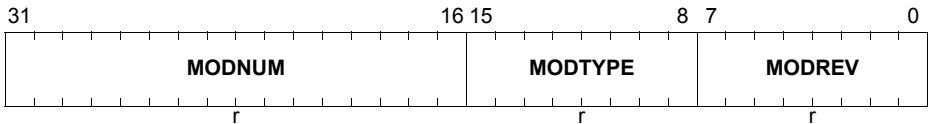
ID

The MSC Module Identification Register ID contains read-only information about the module version.

>> [Registers Overview](#)

ID

Module Identification Register (08_H) **Reset Value: 0028 C01X_H**



Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MODTYPE	[15:8]	r	Module Type This bit field defines the module as a 32-bit module: C0 _H
MODNUM	[31:16]	r	Module Number Value This bit field defines the module identification number for the MSC: 0028 _H

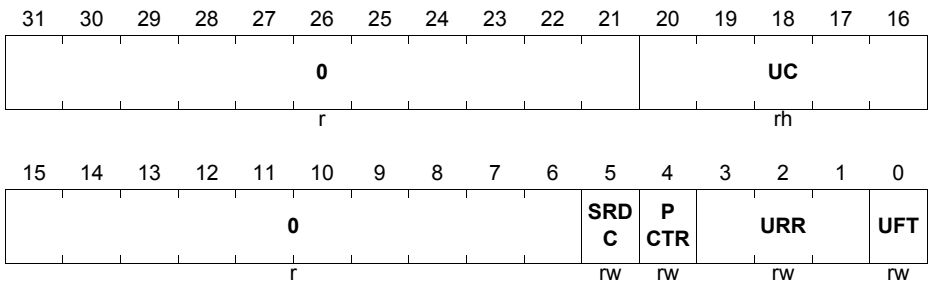
22.3.2 Status and Control Registers
USR

The Upstream Status Register is used to configure the upstream channel data format, baud rate, and parity type. It also provides the status information of the upstream counter (UC).

>> [Registers Overview](#)

USR

Upstream Status Register (10_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
UFT	0	rw	Upstream Channel Frame Type This bit determines the frame type used by the upstream channel for data reception. 0 _B 12-bit upstream frame selected 1 _B 16-bit upstream frame selected (with 4-bit address field)
URR	[3:1]	rw	Upstream Channel Receiving Rate This bit field determines the baud rate for the upstream channel. 000 _B Upstream channel disabled; no reception is possible 001 _B Baud rate = $f_{MSC}/4$ 010 _B Baud rate = $f_{MSC}/8$ 011 _B Baud rate = $f_{MSC}/16$ 100 _B Baud rate = $f_{MSC}/32$ 101 _B Baud rate = $f_{MSC}/64$ 110 _B Baud rate = $f_{MSC}/128$ 111 _B Baud rate = $f_{MSC}/256$

Micro Second Channel (MSC)

Field	Bits	Type	Description
PCTR	4	rw	Parity Control This bit determines the parity mode used by the upstream channel for data reception. 0_B Even parity mode is selected. A parity bit is set on an odd number of 1s in the serial address/data stream. 1_B Odd parity mode is selected. A parity bit is set on an even number of 1s in the serial address/data stream.
SRDC	5	rw	Service Request Delay Control This bit determines the additional delay inserted by the upstream channel when triggering the receive interrupt. Only the hardware generated interrupt can be delayed. The software generated interrupt by writing the ISC.SURDI bit is never delayed. 0_B No delay 1_B Delay of 1 bit time
UC	[20:16]	rh	Upstream Counter This bit field indicates the content of the upstream counter that counts the bits during upstream channel reception.
0	[15:6], [31:21]	r	Reserved Read as 0; shall be written with 0.

Micro Second Channel (MSC)
DSC

The Downstream Control Register is used to control the operation mode and frame layout of the downstream channel transmission. It also contains the two pending status bits.

>> [Registers Overview](#)

DSC

Downstream Control Register (14_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		PPD						0		NBC					
r		rw						r		rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DS DIS	EN SEL H	EN SEL L	NDBH				NDBL				DP	CP	TM		
rh	rw	rw	rw				rw				rh	rh	rw		

Field	Bits	Type	Description
TM	0	rw	Transmission Mode This bit selects the transmission mode of the downstream channel. 0 _B Triggered Mode selected 1 _B Data Repetition Mode selected
CP	1	rh	Command Pending This bit is set when the downstream command register DC is written. CP is cleared when the first bit of the related command frame is sent out.
DP	2	rh	Data Pending In Triggered Mode, this bit is set when the set data pending bit ISC.SDP is set by software. In Data Repetition Mode, this bit is set by hardware at the last passive time frame. At the start of the data frame, DP is cleared by hardware.

Micro Second Channel (MSC)

Field	Bits	Type	Description
NDBL	[7:3]	rw	<p>Number of SRL Bits Shifted at Data Frames</p> <p>NDBL determines the number of shift register low part (SRL) bits that are shifted out on SO during a data frame.</p> <p>00000_B No SRL bit shifted 00001_B SRL[0] shifted 00010_B SRL[1:0] shifted ..._B ... 01111_B SRL[14:0] shifted 10000_B SRL[15:0] shifted</p> <p>Other bit combinations are reserved; do not use these bit combinations.</p>
NDBH	[12:8]	rw	<p>Number of SRH Bits Shifted at Data Frames</p> <p>NDBH determines the number of shift register high part (SRH) bits that are shifted out on SO during a data frame.</p> <p>00000_B No SRH bit shifted; no selection bit is generated, the SRH active phase is completely skipped. 00001_B SRH[0] shifted 00010_B SRH[1:0] shifted ..._B ... 01111_B SRH[14:0] shifted 10000_B SRH[15:0] shifted</p> <p>Other bit combinations are reserved; do not use these bit combinations.</p>
ENSELL	13	rw	<p>Enable SRL Active Phase Selection Bit</p> <p>This bit determines whether a low level selection bit is inserted at the beginning of a data frame's SRL active phase.</p> <p>0_B No selection bit inserted. 1_B Low level selection bit inserted.</p>
ENSELH	14	rw	<p>Enable SRH Active Phase Selection Bit</p> <p>This bit determines whether a low level selection bit is inserted at the beginning of a data frame's SRH active phase.</p> <p>0_B No selection bit inserted. 1_B Low level selection bit inserted.</p>

Micro Second Channel (MSC)

Field	Bits	Type	Description
DSDIS	15	rh	Downstream Disable This bit indicates the state of the downstream channel operation. 0_B The downstream channel is enabled. A frame transmission can take place (Triggered Mode) or takes place (Data Repetition Mode). 1_B Downstream Counter becomes disabled. No new frame transmission is started. A running frame transmission is always completed.
NBC	[21:16]	rw	Number of Bits Shifted at Command Frames This bit field determines how many bits of the SRL/SRH shift registers are shifted out during transmission of a command frame. 000000_B No bit shifted 000001_B SRL[0] shifted 000010_B SRL[1:0] shifted 000011_B SRL[2:0] shifted \dots_B ... 010000_B SRL[15:0] shifted 010001_B SRL[15:0] and SRH[0] shifted 010010_B SRL[15:0] and SRH[1:0] shifted \dots_B ... 011111_B SRL[15:0] and SRH[14:0] shifted 100000_B SRL[15:0] and SRH[15:0] shifted Other bit combinations are reserved; do not use these bit combinations
PPD	[28:24]	rw	Passive Phase Length at Data Frames This bit field determines the length of the passive phase of a data frame. 00000_B Passive phase length is $2 \times t_{FCL}$ 00001_B Passive phase length is $2 \times t_{FCL}$ 00010_B Passive phase length is $2 \times t_{FCL}$ 00011_B Passive phase length is $3 \times t_{FCL}$ \dots_B ... 11111_B Passive phase length is $31 \times t_{FCL}$
0	[23:22], [31:29]	r	Reserved Read as 0; shall be written with 0.

Note: The "rw" bits in the DSC register are buffered in a shadow buffer at the start of a corresponding frame transmission.

Micro Second Channel (MSC)
DSCE

The Downstream Control Enhanced Register is used to control the enhanced operation mode and frame layout of the downstream channel transmission.

>> [Registers Overview](#)

DSCE
Downstream Control Enhanced Register 1(58_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
CDC	INJPOSP1						INJE	NP1	0	INJPOSP0						INJE	NP0
rw	rw						rw	r		rw						rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
CCF	EX												NDB	NDB			
rh	rw												rw	rw			
								0									
								r									

Field	Bits	Type	Description
NDBHE	0	rw	Number of SRH Bits Shifted at Data Frames Extension Additional MSB bit extension of the NDBH bit field.
NDBLE	1	rw	Number of SRH Bits Shifted at Data Frames Extension Additional MSB bit extension of the NDBL bit field.
EXEN	14	rw	Extension Enable Enables the extension bit fields.
CCF	15	rh	Command-Command Flag This bit flags that a second command frame has been written without data frame to be sent in between. It is active only if CTCM=1. Otherwise it is always low. 0 _B no second command 1 _B second command frame pending
INJENP0	16	rw	Injection Enable of the Pin 0 Signal This bit selects if an external signal is injected in a data frame. 0 _B Disabled 1 _B Enabled

Micro Second Channel (MSC)

Field	Bits	Type	Description
INJPOSP 0	[22:17]	rw	Injection Position of the Pin 0 Signal This bit selects the position of the injected external one bit signal, at the bit position in range of 0 to 63 in the data frame. If both PIN0POS and PIN1POS point to a same bit, PIN0POS has the higher priority, that is PIN0 level will be visible in the data frame.
INJENP1	24	rw	Injection Enable of the Pin 1 Signal This bit selects if an external signal is injected in a data frame. 0 _B Disabled 1 _B Enabled
INJPOSP 1	[30:25]	rw	Injection Position of the Pin 1 Signal This bit selects the position of the injected external one bit signal, at the bit position in range of 0 to 63 in the data frame.
CDCM	31	rw	Command-Data-Command in Data Repetition Mode This bit selects if a data frame is automatically inserted between two consecutive command frame requests. 0 _B Automatic data insertion disabled 1 _B Automatic data insertion enabled
0	23, [13:2]	r	Reserved Read as 0; shall be written with 0.

Micro Second Channel (MSC)

USCE

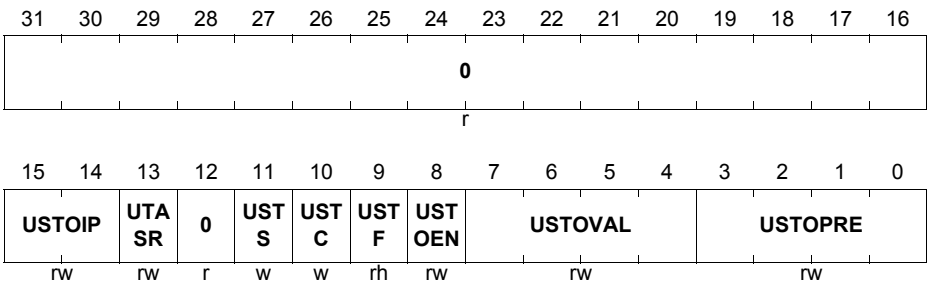
The Upstream Control Enhanced Register is used to control the upstream channel timeout feature.

>> [Registers Overview](#)

USCE

Upstream Control Enhanced Register 1(5C_H)

Reset Value: 0000 00FF_H



Field	Bits	Type	Description
USTOPRE	[3:0]	rw	Upstream Timeout Prescaler Prescaler for the upstream time-out limit. Expressed in upstream bit times. 2 ^N divider in the range of 1 to 32768. Write to this bit field triggers new start of the watchdog timer. 0000 _B 1 0001 _B 2 0010 _B 4 ... _B ... 1110 _B 32768 1111 _B reserved
USTOVAL	[7:4]	rw	Upstream Timeout Value Upstream timeout value for the N-divider in the range of 1 to 16, expressed in number of upstream bit time lengths (the upstream bit time is the reciprocal value of the upstream baud rate). TimeOut = BitTime * 2 ^(USTOPRE+1) * (USTOVAL+1). Write to this bit field triggers new start of the watchdog timer.

Micro Second Channel (MSC)

Field	Bits	Type	Description
USTOEN	8	rw	Upstream Timeout Interrupt Enable Enable bit for the upstream timeout interrupt. The timeout counter runs continuously independently of the USTOEN bit. The USTOEN enables only the interrupt generation. 0_B Disabled 1_B Enabled
USTF	9	rh	Upstream Timeout Flag Signals an upstream timer event. If set by software, the interrupt is also triggered, if enabled. The watchdog timer runs continuously and sets the USTF flag at overflow, independently from the enable bit. Therefore this bit must be cleared with the same write access which enables the interrupt. 0_B Disabled 1_B Enabled
USTC	10	w	Upstream Timeout Clear Clears the USTF flag. Write only bit, reads as 0. 0_B No action 1_B Clear
USTS	11	w	Upstream Timeout Set Sets the USTF flag. Write only bit, reads as 0. 0_B No action 1_B Set
UTASR	13	rw	Upstream Timeout Alternate Service Request Selects if the interrupt signal is routed to the alternate service request node. See Figure 22-29 . 0_B SR Multiplexer selected 1_B Alternate Service Request Node (SR4) selected
USTOIP	[15:14]	rw	Upstream Timeout Interrupt Node Pointer USTOIP selects the service request output line SR _n (n = 0-3) for the data frame interrupt. 00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected
0	[31:16], 12	r	Reserved Read as 0; shall be written with 0.

Micro Second Channel (MSC)

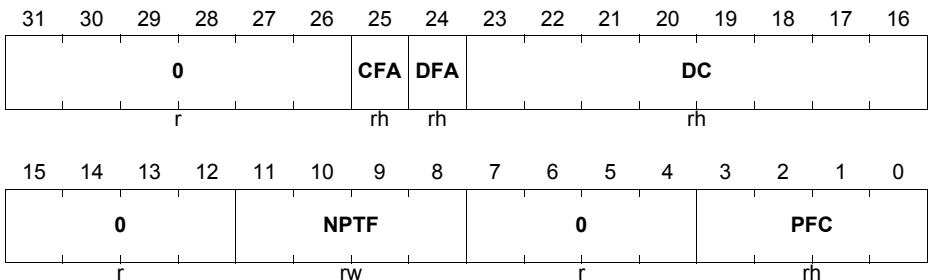
DSS

The Downstream Status Register DSS contains counter bit fields, status bits, and indicates the number of passive time frames.

>> [Registers Overview](#)

DSS
Downstream Status Register

 (18_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
PFC	[3:0]	rh	Passive Time Frame Counter In Data Repetition Mode, this bit field indicates the count of passive time frames that are currently transmitted. In Triggered Mode PFC remains at 0000 _B . 0000 _B Data frame is transmitted. 0001 _B First passive time frame is transmitted. 0010 _B Second passive time frame is transmitted. ... _B ... 1111 _B Fifteenth passive time frame is transmitted.
NPTF	[11:8]	rw	Number Of Passive Time Frames This bit field indicates the number of passive time frames that are inserted in Data Repetition Mode between two data frames. 0000 _B No passive time frame inserted. 0001 _B One passive time frame inserted. 0010 _B Two passive time frames inserted. ... _B ... 1111 _B Fifteen passive time frames inserted. <i>Note: NPTF is buffered in a shadow buffer at the start of each data frame.</i>

Micro Second Channel (MSC)

Field	Bits	Type	Description
DC	[23:16]	rh	Downstream Counter This bit field indicates the number of downstream shift clock periods that have been elapsed since the start of the current frame. 00 _H No shift clock elapsed (after counter reset). 01 _H 1 shift clock elapsed. ... _H ... 7F _H 127 shift clocks elapsed. DC is reset at the end of a downstream frame.
DFA	24	rh	Data Frame Active This bit indicates if a data frame is currently sent out. 0 _B No data frame is currently sent out. 1 _B A data frame is currently sent out.
CFA	25	rh	Command Frame Active This bit indicates if a command frame is currently sent out. 0 _B No command frame is currently sent out. 1 _B A command frame is currently sent out.
0	[7:4], [15:12], [31:26]	r	Reserved Read as 0; shall be written with 0.

Micro Second Channel (MSC)

DSDSL

The bit fields of the Downstream Select Data Low Register DSDSL determine the data source for each bit in shift register SRL.

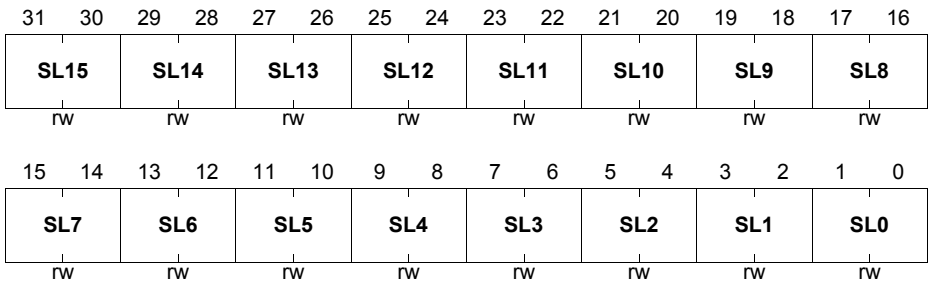
>> [Registers Overview](#)

DSDSL

Downstream Select Data Source Low Register

(24_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SLx (x = 0-15)	[2*x+1: 2*x]	rw	<p>Select Source for SRL</p> <p>SLx determines which data source is used for the shift register bit SRL[x] during data frame transmission.</p> <p>00_B SRL[x] is taken from data register DD.DDL[x].</p> <p>01_B Reserved.</p> <p>10_B SRL[x] is taken from the ALTINL input line x.</p> <p>11_B SRL[x] is taken from the ALTINL input line x in inverted state.</p>

Micro Second Channel (MSC)

DSDSH

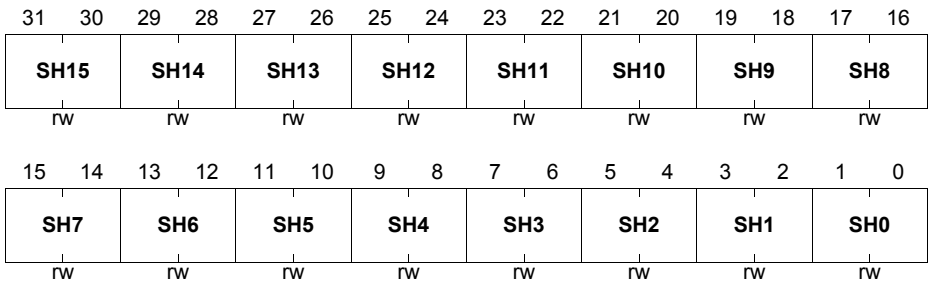
The bit fields of the Downstream Select Data Source High Register DSDSH determine the data source for each bit in shift register SRH.

>> [Registers Overview](#)

DSDSH

Downstream Select Data Source High Register
(28_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SHx (x = 0-15)	[2*x+1: 2*x]	rw	Select Source for SRH SHx determines which data source is used for the shift register bit SRH[x] during data frame transmission. 00 _B SRH[x] is taken from data register DD.DDH[x]. 01 _B Reserved. 10 _B SRH[x] is taken from the ALTINH input line x. 11 _B SRH[x] is taken from the ALTINH input line x in inverted state.

ESR

The Emergency Stop Register ESR determines which bits of SRL and SRH are enabled for emergency operation.

>> [Registers Overview](#)

ESR
Emergency Stop Register
(2C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
ENL_x (x = 0-15)	x	rW	Emergency Stop Enable for Bit x in SRL This bit enables the emergency stop feature selectively for each SRL bit. If the emergency stop condition is met and enabled (ENL _x = 1), the SRL[x] bit is of the data register DD.DDL[x] is used for the shift register load operation. 0 _B Emergency stop feature for bit SRL[x] is disabled. 1 _B The emergency stop feature for bit SRL[x] is enabled.
ENH_x (x = 0-15)	x+16	rW	Emergency Stop Enable for Bit x in SRH This bit enables the emergency stop feature selectively for each SRH bit. If the emergency stop condition is met and enabled (ENH _x = 1), the SRH[x] bit of the data register DD.DDH[x] is used for the shift register load operation. 0 _B Emergency stop feature for bit SRH[x] is disabled. 1 _B The emergency stop feature for bit SRH[x] is enabled.

Micro Second Channel (MSC)

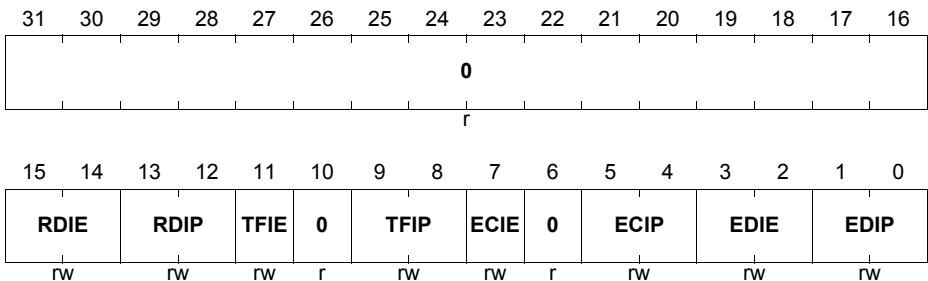
ICR

The Interrupt Control Register ICR holds the interrupt enable bits and interrupt pointers of four MSC interrupts.

>> [Registers Overview](#)

ICR

Interrupt Control Register (40_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
EDIP	[1:0]	rw	<p>Data Frame Interrupt Node Pointer EDIP selects the service request output line SR_n (n = 0-3) for the data frame interrupt.</p> <p>00_B Service request output SR0 selected 01_B Service request output SR1 selected 10_B Service request output SR2 selected 11_B Service request output SR3 selected</p>
EDIE	[3:2]	rw	<p>Data Frame Interrupt Enable This bit field determines the enable conditions for the data frame interrupt.</p> <p>00_B Interrupt generation disabled 01_B An interrupt is generated when the last data bit has been shifted out. 10_B An interrupt is generated when the first data bit has been shifted out, but only if DSC.NDBL is not equal 00000_B. This means, at least one SRL bit must be shifted out for the first data bit shifted interrupt to become active. 11_B Interrupt generation disabled</p>

Micro Second Channel (MSC)

Field	Bits	Type	Description
ECIP	[5:4]	rw	Command Frame Interrupt Node Pointer ECIP selects the service request output line SRn (n = 0-3) for the command frame interrupt. 00 _B Service request output SR0 selected 01 _B Service request output SR1 selected 10 _B Service request output SR2 selected 11 _B Service request output SR3 selected
ECIE	7	rw	Command Frame Interrupt Enable This bit enables the command frame interrupt. 0 _B Interrupt generation disabled. 1 _B Interrupt generation enabled.
TFIP	[9:8]	rw	Time Frame Interrupt Pointer TFIP selects the service request output line SRn (n = 3-0) for the time frame interrupt. 00 _B Service request output SR0 selected 01 _B Service request output SR1 selected 10 _B Service request output SR2 selected 11 _B Service request output SR3 selected
TFIE	11	rw	Time Frame Interrupt Enable This bit enables the time frame interrupt. 0 _B Interrupt generation disabled. 1 _B Interrupt generation enabled.
RDIP	[13:12]	rw	Receive Data Interrupt Pointer RDIP selects the service request output line SRn (n = 3-0) for the receive data interrupt. 00 _B Service request output SR0 selected 01 _B Service request output SR1 selected 10 _B Service request output SR2 selected 11 _B Service request output SR3 selected

Micro Second Channel (MSC)

Field	Bits	Type	Description
RDIE	[15:14]	rw	Receive Data Interrupt Enable This bit field determines the enable conditions for the receive data interrupt. 00 _B Interrupt generation disabled. 01 _B An interrupt is generated when data is received and written into the upstream data registers UD _x (x = 0-3). 10 _B An interrupt is generated as with RDIE = 01 _B but only if the received data is not equal to 00 _H . 11 _B An interrupt is generated when data is received and written into register UD3.
0	6, 10, [31:16]	r	Reserved Read as 0; shall be written with 0.

Micro Second Channel (MSC)

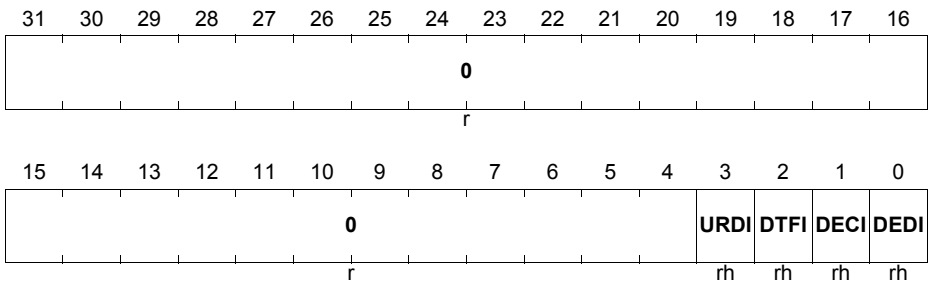
ISR

The Interrupt Status Register ISR holds the interrupt status flags that indicate an interrupt occurrence in downstream and upstream channels.

>> [Registers Overview](#)

ISR

Interrupt Status Register (44_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
DEDI	0	rh	Data Frame Interrupt Flag This flag is always set by hardware when a downstream channel data frame interrupt is generated. DEDI can be set or cleared by software when writing to register ISC with the appropriate bits ISC.SDEDI or ISC.CDEDI set.
DECI	1	rh	Command Frame Interrupt Flag This flag is always set by hardware when a downstream channel command frame interrupt is generated, whether or not it is enabled. DECI can be set or cleared by software when writing to register ISC with the appropriate bits SDECI or CDECI set.
DTFI	2	rh	Time Frame Interrupt Flag This flag is always set by hardware when a downstream channel time frame interrupt is generated, whether or not it is enabled. DTFI can be set or cleared by software when writing to register ISC with the appropriate bits SDTFI or CDTFI set.

Micro Second Channel (MSC)

Field	Bits	Type	Description
URDI	3	rh	Receive Data Interrupt Flag This flag is always set by hardware when an upstream channel receive data interrupt is generated, whether or not it is enabled. URDI can be set or cleared by software when writing to register ISC with the appropriate bits SURDI or CURDI set.
0	[31:4]	r	Reserved Read as 0; shall be written with 0.

Micro Second Channel (MSC)
ISC

The Interrupt Set Clear Register ISC is used to set or clear the MSC interrupt flags located in the Interrupt Status Register ISR. Reading ISC always returns 0000 0000_H.

>> [Registers Overview](#)

ISC
Interrupt Set Clear Register
(48_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									S	S	S	S	S	S	S
									DDIS	CP	DP	URDI	DTFI	DECI	DEDI
r									w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									C	C	C	C	C	C	C
									DDIS	CP	DP	URDI	DTFI	DECI	DEDI
r									w	w	w	w	w	w	w

Field	Bits	Type	Description
CDEDI	0	w	Clear DEDI Flag 0 _B No operation 1 _B Bit ISR.DEDI is cleared.
CDECI	1	w	Clear DECI Flag 0 _B No operation 1 _B Bit ISR.DECI is cleared.
CDTFI	2	w	Clear DTFI Flag 0 _B No operation 1 _B Bit ISR.DTFI is cleared.
CURDI	3	w	Clear URDI Flag 0 _B No operation 1 _B Bit ISR.URDI is cleared.
CDP	4	w	Clear DP Flag 0 _B No operation 1 _B Bit DSC.DP is cleared.
CCP	5	w	Clear CP Flag 0 _B No operation 1 _B Bit DSC.CP is cleared.

Micro Second Channel (MSC)

Field	Bits	Type	Description
CDDIS	6	w	Clear DSDIS Flag 0 _B No operation 1 _B Bit DSC.DSDIS is cleared.
SDEDI	16	w	Set DEDI Flag 0 _B No operation 1 _B Bit ISR.DEDI is set.
SDECI	17	w	Set DECI Flag 0 _B No operation 1 _B Bit ISR.DECI is set.
SDTFI	18	w	Set DTFI Flag 0 _B No operation 1 _B Bit ISR.DTFI is set.
SURDI	19	w	Set URDI Flag 0 _B No operation 1 _B Bit ISR.URDI is set.
SDP	20	w	Set DP Bit 0 _B No effect 1 _B Bit DSC.DP is set.
SCP	21	w	Set CP Flag 0 _B No operation 1 _B Bit DSC.CP is set.
SDDIS	22	w	Set DSDIS Flag 0 _B No operation 1 _B Bit DSC.DSDIS is set.
0	[15:7], [31:23]	r	Reserved Read as 0; shall be written with 0.

Note: When the ISC register is written with both bits (set and reset bit) for a specific interrupt flag, the clear operation takes place and the set operation is ignored.

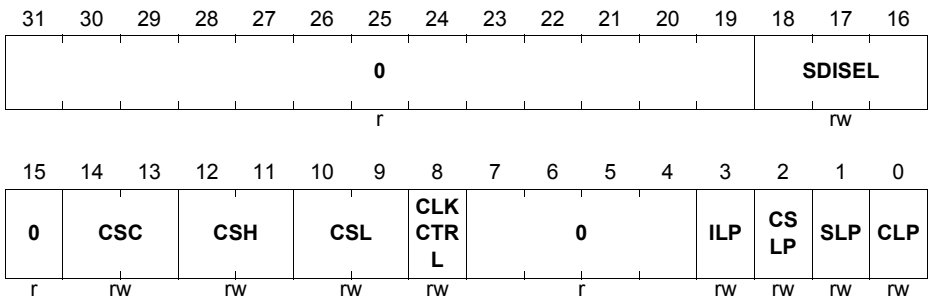
Micro Second Channel (MSC)
OCR

The Output Control Register OCR determines the MSC input/output signal polarities, the chip select output signal assignment, and the serial output clock generation.

>> [Registers Overview](#)

OCR

Output Control Register (4C_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
CLP	0	rw	FCLP Line Polarity 0 _B FCLP and FCL signal polarity is identical. FCLN signal has inverted FCL signal polarity. 1 _B FCLP signal has inverted FCL signal polarity. FCLN and FCL signal polarities are identical.
SLP	1	rw	SOP Line Polarity 0 _B SOP and SO signal polarity is identical. SON signal has inverted SO signal polarity. 1 _B SOP signal has inverted SO signal polarity. SON and SO signal polarities are identical.
CSLP	2	rw	Chip Selection Lines Polarity 0 _B EN[3:0] and ENL, ENH, ENC signal polarities are identical (high active). 1 _B EN[3:0] signal polarities are inverted (low active) to the ENL, ENH, ENC signal polarities. Bit CSLP is buffered during a frame transmission. This means that any change of CSLP becomes valid first with the start of the next frame transmission.

Micro Second Channel (MSC)

Field	Bits	Type	Description
ILP	3	rw	SDI Line Polarity 0 _B SDI and SI signal polarities are identical. 1 _B SDI and SI signal polarities are inverted.
CLKCTRL	8	rw	Clock Control This bit determines the activation of clock output FCL. 0 _B FCL is activated only during the active phases of data or command frames (not during passive time frames). 1 _B FCL is always active whether or not a downstream frame is currently transmitted.
CSL	[10:9]	rw	Chip Enable Selection for ENL This bit field selects the chip enable output ENx that becomes active during the SRL active phase (ENL = 1) of a data frame. The active level of ENx is defined by bit CSLP. 00 _B EN0 line is selected for ENL. 01 _B EN1 line is selected for ENL. 10 _B EN2 line is selected for ENL. 11 _B EN3 line is selected for ENL.
CSH	[12:11]	rw	Chip Enable Selection for ENH This bit field selects the chip enable output ENx that becomes active during the SRH active phase (ENH = 1) of a data frame. The active level of ENx is defined by bit CSLP. 00 _B EN0 line is selected for ENH. 01 _B EN1 line is selected for ENH. 10 _B EN2 line is selected for ENH. 11 _B EN3 line is selected for ENH.
CSC	[14:13]	rw	Chip Enable Selection for ENC This bit field selects the chip enable output ENx that becomes active during the active phase (ENC = 1) of a command frame. The active level of ENx is defined by bit CSLP. 00 _B EN0 line is selected for ENC. 01 _B EN1 line is selected for ENC. 10 _B EN2 line is selected for ENC. 11 _B EN3 line is selected for ENC.

Micro Second Channel (MSC)

Field	Bits	Type	Description
SDISEL	[18:16]	rw	Serial Data Input Selection This bit field selects the source for the serial data input SDI of the upstream channel. 000 _B SDI0 input is selected for SDI. 001 _B SDI1 input is selected for SDI. 010 _B SDI2 input is selected for SDI. 011 _B SDI3 input is selected for SDI. 100 _B SDI4 input is selected for SDI. 101 _B SDI5 input is selected for SDI. 110 _B SDI6 input is selected for SDI. 111 _B SDI7 input is selected for SDI.
0	[7:4], 15, [31:19]	r	Reserved Read as 0; shall be written with 0.

Micro Second Channel (MSC)

DSDSLE

The bit fields of the Downstream Select Data Low Register DSDSL determine the data source for the upper 16 bits in the shift register SRL.

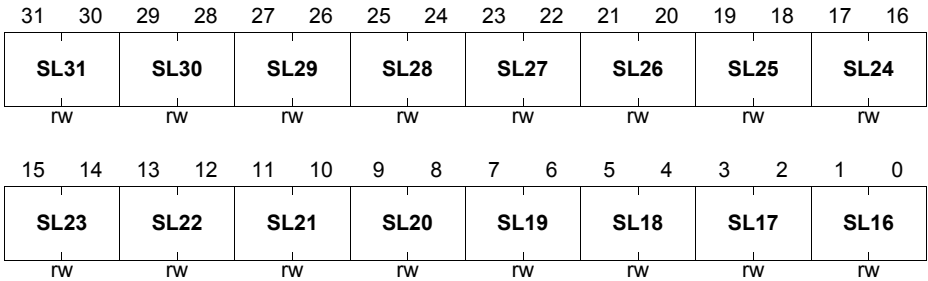
>> [Registers Overview](#)

DSDSLE

Downstream Select Data Source Low Extension Register

(60_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SL_x (x = 16-31)	[2*x-31 : 2*x-32]	rw	Select Source for SRL SL _x determines which data source is used for the shift register bit SRL[x] during data frame transmission. 00 _B SRL[x] is taken from data register DD.DDL[x]. 01 _B Reserved. 10 _B SRL[x] is taken from the ALTINL input line x. 11 _B SRL[x] is taken from the ALTINL input line x in inverted state.

Micro Second Channel (MSC)

DSDSHE

The bit fields of the Downstream Select Data Source High Register DSDSH determine the data source for the upper 16 bit in the shift register SRH.

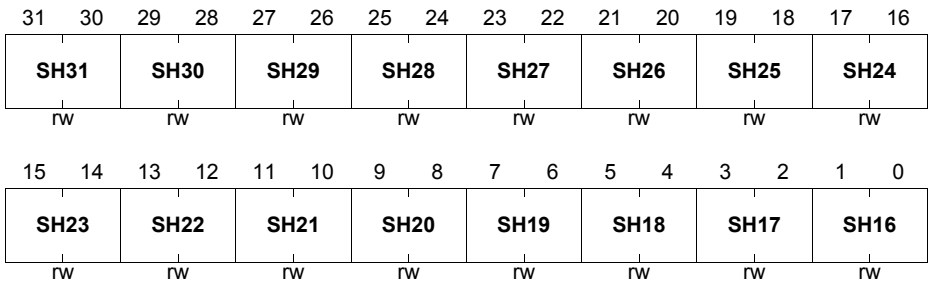
>> [Registers Overview](#)

DSDSHE

Downstream Select Data Source High Register

(64_H)

Reset Value: 0000 0000_H

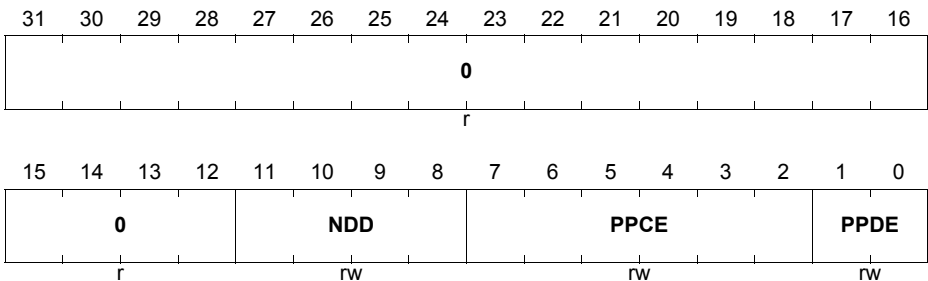


Field	Bits	Type	Description
SHx (x = 16-31)	[2*x-31 : 2*x-32]	rw	<p>Select Source for SRH</p> <p>SHx determines which data source is used for the shift register bit SRH[x] during data frame transmission.</p> <p>00_B SRH[x] is taken from data register DD.DDH[x].</p> <p>01_B Reserved.</p> <p>10_B SRH[x] is taken from the ALTINH input line x.</p> <p>11_B SRH[x] is taken from the ALTINH input line x in inverted state.</p>

Micro Second Channel (MSC)
DSTE

The Downstream Control Enhanced Register is used to control the enhanced operation mode and frame layout of the downstream channel transmission.

>> [Registers Overview](#)

DSTE
Downstream Timing Extension Register (74_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
PPDE	[1:0]	rw	Passive Phase Length at Data Frames Extension Additional MSB bits extension of the PPD bit field.
PPCE	[7:2]	rw	Passive Phase Length at Control Frames Extension Additional MSB bits extension of the fixed length of 2 of the command frames passive phase. The final length is 2 + PPCE resulting in a range of 2 to 65. 000000 _B 2 000001 _B 3 ... _B ... 111111 _B 65
NDD	[11:8]	rw	N Divider Downstream Defines the division ratio in the range of 1 to 16. 0000 _B 1 0001 _B 2 ... _B ... 1111 _B 16
0	[31:12]	r	Reserved Read as 0; shall be written with 0.

22.3.3 Data Registers

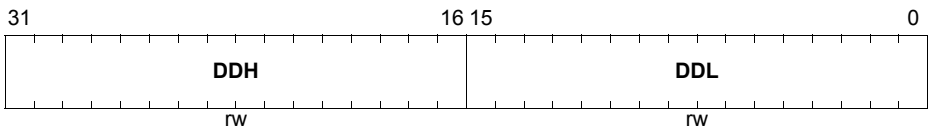
DD

The Downstream Data Register DD contains data to be transmitted during data frames.

>> [Registers Overview](#)

DD

Downstream Data Register (1C_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
DDL	[15:0]	rw	Downstream Data for SRL Shift Register Contains the data bits to be transmitted during the SRL active phase of a data frame.
DDH	[31:16]	rw	Downstream Data for SRH Shift Register Contains the data bits to be transmitted during the SRH active phase of a data frame.

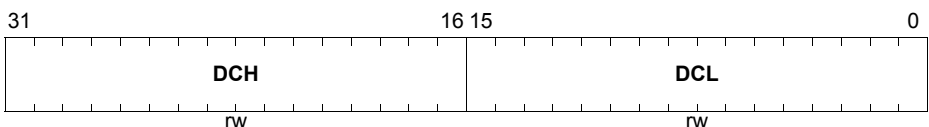
DC

The Downstream Command Register DC contains command information to be transmitted during command frames.

>> [Registers Overview](#)

DC

Downstream Command Register (20_H) **Reset Value: 0000 0000_H**



Micro Second Channel (MSC)

Field	Bits	Type	Description
DCL	[15:0]	rw	Downstream Command for SRL Shift Register Contains the data bits to be transmitted during the SRL active phase of a command frame.
DCH	[31:16]	rw	Downstream Command for SRH Shift Register Contains the data bits to be transmitted during the SRH active phase of a command frame.

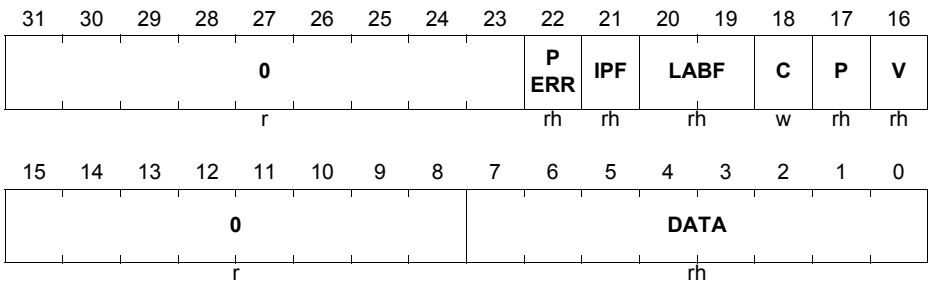
Micro Second Channel (MSC)
UDx

The four Upstream Data Registers UDx store the content (data, addresses, received and calculated parity bit, parity error bit) of a received upstream channel data frame.

>> [Registers Overview](#)

UDx (x = 0-3)

Upstream Data Register x **(30_H+x*4_H)** **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
DATA	[7:0]	rh	Received Data This bit field contains the 8-bit receive data.
V	16	rh	Valid Bit This bit is set by hardware when the received data is written to UDx. Writing bit C = 1 clears V. If hardware setting and software clearing of the valid bit occur simultaneously, bit V will be cleared.
P	17	rh	Parity Bit This flag contains the parity bit that has been received with the data frame.
C	18	w	Clear Bit 0 _B No operation. 1 _B Bit V is cleared. C is always read as 0.
LABF	[20:19]	rh	Lower Address Bit Field This bit field contains the two address bits A[1:0] of the 4-bit address field (16-bit data frame). If 12-bit data frame is selected, LABF is always set to 00 _B .

Micro Second Channel (MSC)

Field	Bits	Type	Description
IPF	21	rh	Internal Parity Flag This bit contains the parity bit that has been calculated in the MSC during data frame reception.
PERR	22	rh	Parity Error This bit indicates if a start bit error, parity error, or stop bit error occurred during frame reception. 0 _B No error detected. 1 _B Error detected.
0	[15:8], [31:23]	r	Reserved Read as 0; shall be written with 0.

22.3.4 Extension Registers

This sub-chapter describes the registers related to 64-bit extension of the MSC module.

ESRE

The Emergency Stop Extension Register ESRE determines which bits of SRL and SRH are enabled for emergency operation.

>> [Registers Overview](#)

ESRE

Emergency Stop Extension Register (68_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH	ENH
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL	ENL
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Field	Bits	Type	Description
ENLx (x = 16-31)	x-16	rw	Emergency Stop Enable for Bit x in SRL This bit enables the emergency stop feature selectively for each SRL bit. If the emergency stop condition is met and enabled (ENLx = 1), the SRL[x] bit of the data register DD.DDL[x] is used for the shift register load operation. 0 _B Emergency stop feature for bit SRL[x] is disabled. 1 _B The emergency stop feature for bit SRL[x] is enabled.
ENHx (x = 16-31)	x	rw	Emergency Stop Enable for Bit x in SRH This bit enables the emergency stop feature selectively for each SRH bit. If the emergency stop condition is met and enabled (ENHx = 1), the SRH[x] bit of the data register DD.DDH[x] is used for the shift register load operation. 0 _B Emergency stop feature for bit SRH[x] is disabled. 1 _B The emergency stop feature for bit SRH[x] is enabled.

Micro Second Channel (MSC)

DDE

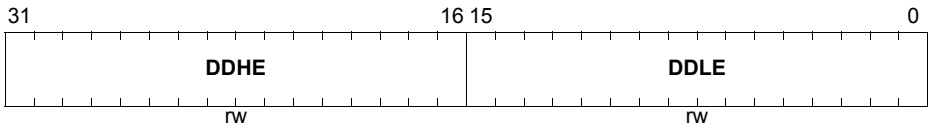
The Downstream Data Extension Register DDE contains data to be transmitted during data frames.

>> [Registers Overview](#)

DDE

Downstream Data Extension Register (6C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
DDLE	[15:0]	rw	Downstream Data Extension for SRL Shift Register Contains the data bits to be transmitted during the SRL active phase of a data frame.
DDHE	[31:16]	rw	Downstream Data Extension for SRH Shift Register Contains the data bits to be transmitted during the SRH active phase of a data frame.

Micro Second Channel (MSC)

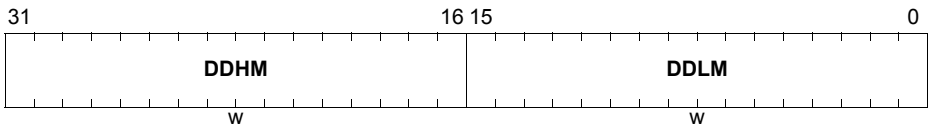
DDM

The Downstream Data Mirror Register DDM provides alternative address for the DD register. The purpose of this alternative address is to provide convenient way to write up to 64 bit data blocks to the MSC module by using DMA with automatic address increment and wrap around. The DMA writes first to the DDE register, and then to the DDM register, which triggers the data transfer in the same way as a write to the DD register.

>> [Registers Overview](#)

DDM

Downstream Data Mirror Register (70_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
DDL M	[15:0]	w	Downstream Data Mirror for SRL Shift Register Contains the data bits to be transmitted during the SRL active phase of a data frame.
DD H M	[31:16]	w	Downstream Data Mirror for SRH Shift Register Contains the data bits to be transmitted during the SRH active phase of a data frame.

22.3.5 Asynchronous Block Registers

ABC

The Asynchronous Block Configuration register ABC contains all the bit fields related to the ABRA block.

>> [Registers Overview](#)

>> [Configuring the ABRA block](#)

ABC

Asynchronous Block Configuration Register (80_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ABB	0	CLKSEL		UIE	UFM	UNF	0	UAS R	UIP		NDA				
rw	r	rw		rw	w	r	r	rw	rw		rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OIE	OFM	OVF	0	OAS R	OIP	HIGH			LOW						
rw	w	r	r	rw	rw	rw			rw						

Field	Bits	Type	Description
LOW	[3:0]	rw	Duration of the Low Phase of the Shift Clock Defines the duration of the low phase of the shift clock in periods of f_A in the range of 1 to 16. 0000 _B 1 0001 _B 2 ... _B ... 1111 _B 16
HIGH	[7:4]	rw	Duration of the High Phase of the Shift Clock Defines the duration of the high phase of the shift clock in periods of f_A in the range of 1 to 16. 0000 _B 1 0001 _B 2 ... _B ... 1111 _B 16

Micro Second Channel (MSC)

Field	Bits	Type	Description
OIP	[9:8]	rw	Overflow Interrupt Node Pointer OIP selects the service request output line SR _n (n = 0-3) for the overflow interrupt. 00 _B Service request output SR0 selected 01 _B Service request output SR1 selected 10 _B Service request output SR2 selected 11 _B Service request output SR3 selected
OASR	10	rw	Overflow Alternate Service Request Selects if the interrupt signal is routed to the alternate service request node. See Figure 22-29 . 0 _B SR Multiplexer selected 1 _B Alternate Service Request Node (SR4) selected
OVF	12	r	Overflow Flag Indicates if overflow error has occurred. 0 _B No overflow error 1 _B Overflow error
OFM	[14:13]	w	Overflow Flag Modify Sets or clears the overflow flag OF. 00 _B No action 01 _B Set (and triggers the overflow interrupt if enabled) 10 _B Clear (additionally flushes the ABRA FIFO and resets the bit counter) 11 _B No action
OIE	15	rw	Overflow Interrupt Enable Enables or disables the path of the interrupt signal towards the interrupt node. If enabled, an overflow event triggers an interrupt, and if disabled then not. The OF flag always indicates the occurrence of an overflow event, independently of OIE. 0 _B Overflow interrupt disabled 1 _B Overflow interrupt enabled
NDA	[18:16]	rw	N Divider ABRA Defines the division ratio in the range of 1 to 8. 000 _B 1 001 _B 2 ... _B ... 111 _B 8

Micro Second Channel (MSC)

Field	Bits	Type	Description
UIP	[20:19]	rw	Underflow Interrupt Node Pointer UIP selects the service request output line SRn (n = 0-3) for the underflow interrupt. 00 _B Service request output SR0 selected 01 _B Service request output SR1 selected 10 _B Service request output SR2 selected 11 _B Service request output SR3 selected
UASR	21	rw	Underflow Alternate Service Request Selects if the interrupt signal is routed to the alternate service request node. See Figure 22-29 . 0 _B SR Multiplexer selected 1 _B Alternate Service Request Node (SR4) selected
UNF	23	r	Underflow Flag Indicates if overflow error has occurred. 0 _B No overflow error 1 _B Overflow error
UFM	[25:24]	w	Underflow Flag Modify Sets or clears the overflow flag UF. 00 _B No action 01 _B Set (also triggers the underflow interrupt if enablede) 10 _B Clear (additionally flushes the ABRA FIFO and resets the bit counter) 11 _B No action
UIE	26	rw	Underflow Interrupt Enable Enables or disables the path of the interrupt signal towards the interrupt node. If enabled, an overflow event triggers an interrupt, and if disabled then not. The UF flag always indicates the occurrence of an overflow event, independently of UIE. 0 _B Overflow interrupt disabled 1 _B Overflow interrupt enabled
CLKSEL	[29:27]	rw	Clock Select Selects the clock source for the ABRA block. 000 _B no clock 001 _B f_{SPB} 010 _B f_{SRI} 100 _B f_{ERAY} others , reserved (no clock)

Micro Second Channel (MSC)

Field	Bits	Type	Description
ABB	31	rw	Asynchronous Block Bypass Defines if the asynchronous block and the n-divider of the MSC downstream path (located parallel to the fractional divider) are used or not. If bypassed, then also disabled. 0_B Bypassed and disabled 1_B Not bypassed and active
0	11, 22, 30	r	Reserved Read as 0; shall be written with 0.

*Note: The reconfiguration of the clock source has to be done by using two writes: first a write of zero to the CLKSEL bitfield, and then a second write defining the new clock source. Between the first and the second write a delay of minimum $4 * (1/f_A) + 2 * (1/f_{CLC})$ must be inserted by software, where f_A is the frequency being switched off with the first write. Exception: in case that the f_{CLC} is selected as f_A (CSR.CLKSEL = 1), no delay cycles between the writes are necessary. In both cases, simply using one write defining the new clock source is not allowed.*

22.4 MSC Module Implementation

This section describes the MSC module interface as implemented in the TC26x B-Step. It especially covers clock control, port and on-chip connections, interrupt control, and address decoding.

22.4.1 Module Identification Registers

The reset value of the ID module identification register is 0028 C01X_H.

22.4.2 BPI_FPI Module Registers (Single Kernel Configuration)

22.4.2.1 System Registers

Figure 22-38 shows all registers associated with the BPI_FPI module, configured for one kernel.

BPI_FPI Registers Overview

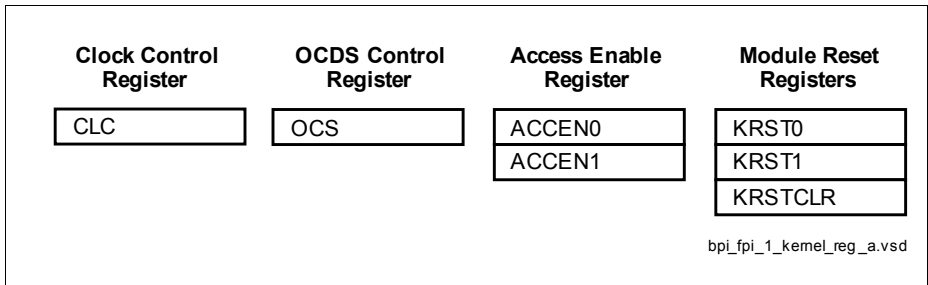


Figure 22-38 BPI_FPI Registers

The writes of the bus masters to the MSC module is controlled by Acces Protection registers ACCENx.

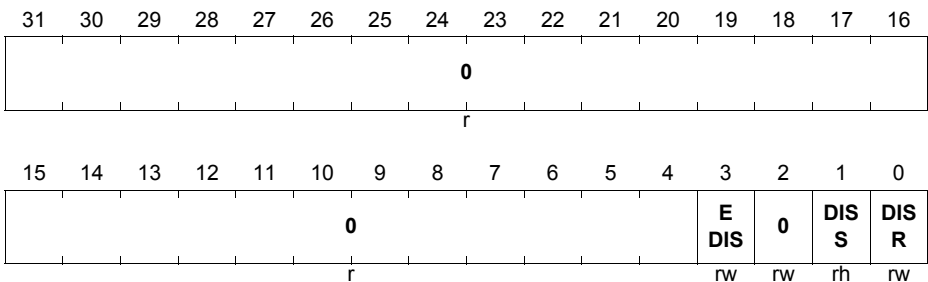
The MSC implements two ACCENx registers, ACCEN0 and ACCEN1.

The ACCENx registers are protected by Safety Endinit mechanism.

Micro Second Channel (MSC)
Clock Control Register (CLC)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI_FPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the f_{\leftarrow} module clock signal, sleep mode and disable mode for the module.

>> [Registers Overview](#)

CLC
Clock Control Register
(00_H)
Reset Value: 0000 0003_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
EDIS	3	rw	Sleep Mode Enable Control Used to control module's sleep mode.
0	[31:4]	r	Reserved Read as 0; shall be written with 0.
0	2	rw	Reserved Read as 0; shall be written with 0.

OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32 bit wide only and requires Supervisor Mode

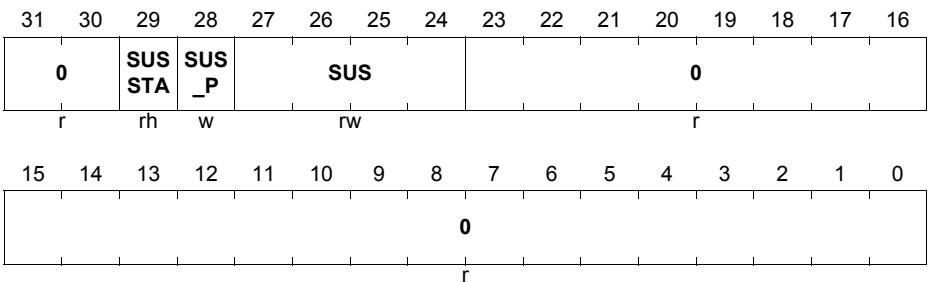
>> [Registers Overview](#)

OCS

OCDS Control and Status

(E8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately. 2 _H Soft suspend others , reserved
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:0], [31:30]	r	Reserved Read as 0; must be written with 0.

Access Enable Register (ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ... , EN31 -> TAG ID 011111_B.

>> [Registers Overview](#)

ACCEN0
Access Enable Register 0
(FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register (ACCEN1)

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

Micro Second Channel (MSC)

AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000_B, EN1 -> TAG ID 100001_B, ... , EN31 -> TAG ID 111111_B.

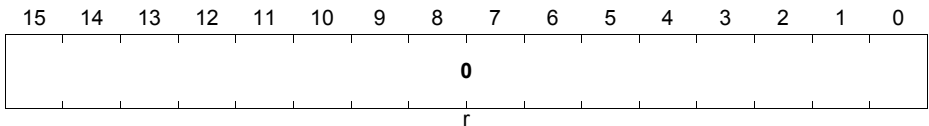
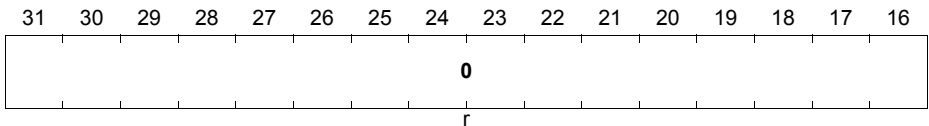
>> [Registers Overview](#)

ACCEN1

Access Enable Register 1

(F8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; shall be written with 0.

Micro Second Channel (MSC)

Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLE.CLR register bit.

Note: During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

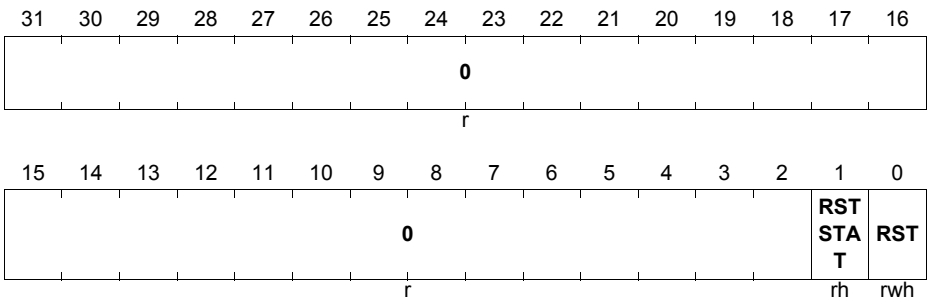
>> [Registers Overview](#)

KRST0

Kernel Reset Register 0

(F4_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set.</p> <p>0_B No kernel reset was requested 1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p>

Micro Second Channel (MSC)

Field	Bits	Type	Description
RSTSTAT	1	rh	<p>Kernel Reset Status</p> <p>This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits.</p> <p>0_B No kernel reset was executed 1_B Kernel reset was executed</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.</p>
0	[31:2]	r	<p>Reserved</p> <p>Read as 0; shall be written with 0.</p>

Micro Second Channel (MSC)

Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

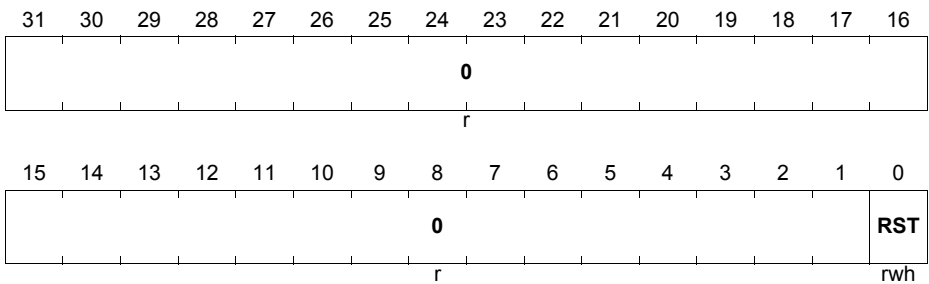
>> [Registers Overview](#)

KRST1

Kernel Reset Register 1

(F0_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; shall be written with 0.

Micro Second Channel (MSC)

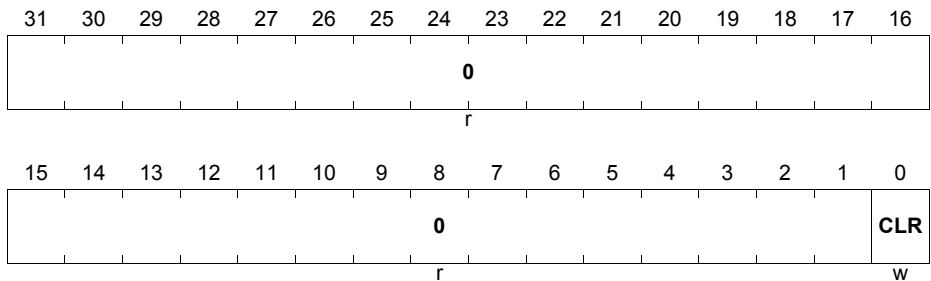
Kernel Reset Status Clear Register (KRSTCLR)

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

>> [Registers Overview](#)

KRSTCLR

Kernel Reset Status Clear Register (EC_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; shall be written with 0.

22.4.3 Interface Connections of the MSC Module

The MSC module is supplied with a separate clock control, address decoding, and interrupt control logic. The service request outputs are connected with the interrupt nodes in the Interrupt Router module. Outputs of the GTM module are connected to the alternate input buses ALTINL/ALTINH. The emergency stop output from the SCU controls the corresponding inputs of MSC0 module.

The serial data and clock outputs of the downstream channels of the MSC module are connected to combined GPIO/LVDS differential output drivers. Details about these four pins are defined in the ports chapter.

22.4.4 MSC Module-Related External Registers

Figure 22-39 summarizes the module-related external registers which are required for MSC programming (see also **Figure 22-37** for the module kernel specific registers). Generally, additionally to programming the MSC kernel and BPI registers, the Fractional Divider, the ports registers located in the Ports module and interrupt registers located in the Interrupt Router module.

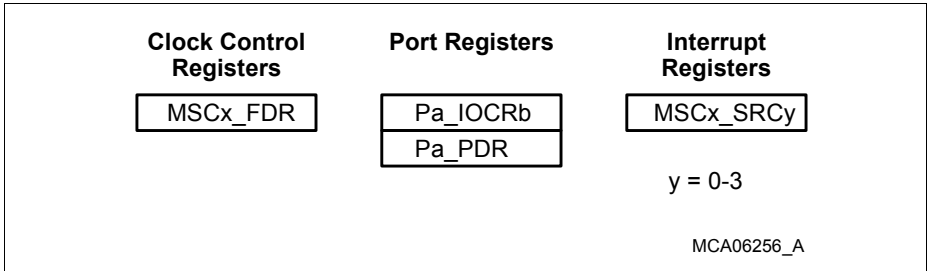


Figure 22-39 MSC Implementation-specific Special Function Registers

22.4.5 Clock Control

The following section describes the clock control and the baud rate generation of the MSC module.

The MSC module is provided with two independent clock signals (**Figure 22-40**):

- f_{CLC}
 This is the module clock that is used inside the MSC kernel for control purposes such as clocking of control logic and register operations. The frequency of f_{CLC} is always identical to the system clock frequency f_{SPB} . The clock control register MSC_CLC makes it possible to enable/disable f_{CLC} under certain conditions.
- f_{MSC}
 This clock is the module clock that is used inside the MSC for baud rate generation of the serial upstream and downstream channel. The fractional divider register MSC_FDR controls the frequency of f_{MSC} and makes it possible to enable/disable it independent of f_{CLC} .
- f_{MSCA}
 This clock is the downstream clock which is used together with the ABRA block. In this case, the upstream baud rate can be fine-tuned with the fractional divider, and the downstream channel frequency is configured with an n-divider.

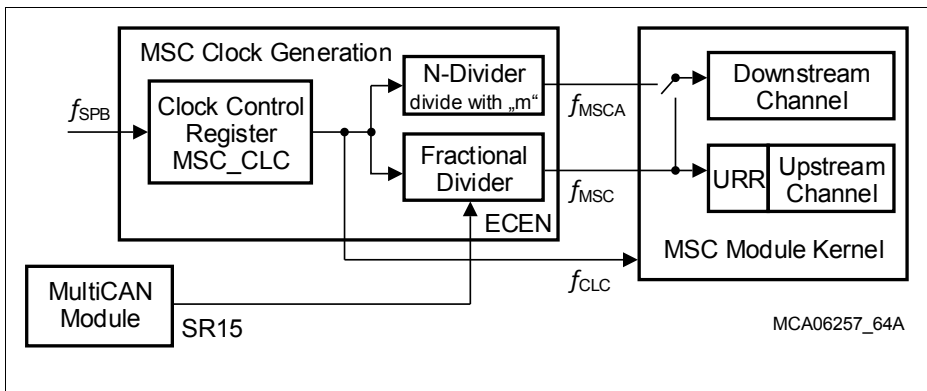
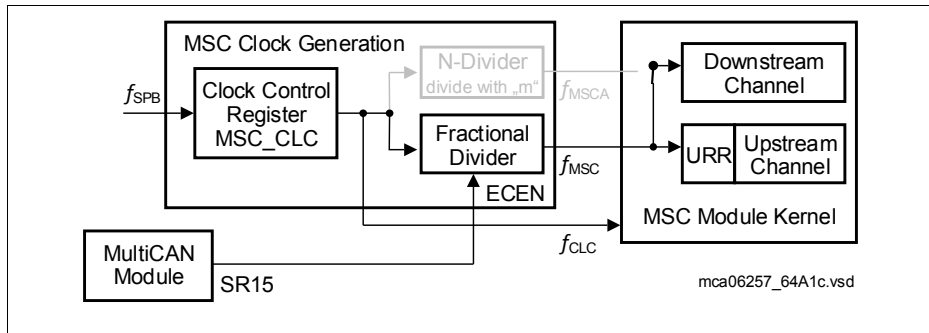


Figure 22-40 MSC Module Clock Generation

Micro Second Channel (MSC)

22.4.5.1 Clock Control Without the ABRA Block

Not using the ABRA block is the standard, compatibility use-case, which guarantees the full compatibility with the standard MSC module not containing the ABRA block. The clock generation is therefore identical with the standard MSC clock generation. The additional N-Divider (dividing with the division factor named “m”) is not used. Only the fractional divider is used (using the factor named “n” in the formulas below).


Figure 22-41 MSC Module Clock Generation

The following two formulas define the frequency of f_{MSC} :

$$f_{MSC} = f_{SPB} \times \frac{1}{n} \text{ with } n = 1024 - \text{MSC.FDR.STEP} \quad (22.4)$$

$$f_{MSC} = f_{SPB} \times \frac{n}{1024} \text{ with } n = 0-1023 \quad (22.5)$$

Micro Second Channel (MSC)
Downstream Channel Baud Rate

As the clock signal FCL of the synchronous downstream channel is always half the frequency of f_{MSC} , the resulting downstream channel baud rate is defined by:

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{1}{2 \times (1024 - \text{MSC.FDR.STEP})} \quad (22.6)$$

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{\text{MSC.FDR.STEP}}{2 \times 1024} \quad (22.7)$$

Upstream Channel Baud Rate

The baud rate of the asynchronous upstream channel is derived from the module clock f_{MSC} by a programmable clock divider selected by bit field USR.URR (see also [Equation \(22.3\)](#) on [Page 22-28](#)). The divide factor DF can be at minimum 4 and at maximum 256.

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{1}{DF \times (1024 - \text{MSC.FDR.STEP})} \quad (22.8)$$

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{\text{MSC.FDR.STEP}}{DF \times 1024} \quad (22.9)$$

[Equation \(22.4\)](#), [Equation \(22.6\)](#), and [Equation \(22.8\)](#) are valid for normal divider mode (FDR.DM = 01_B). [Equation \(22.5\)](#), [Equation \(22.7\)](#), and [Equation \(22.9\)](#) are valid for fractional divider mode (FDR.DM = 10_B).

Micro Second Channel (MSC)

22.4.5.2 Clock Control when using the ABRA Block

Using the ABRA block makes additional considerations regarding the baud-rate adjustments necessary.

An additional N-Divider **DSTE.NDD** is used for configuring the intermediate downstream baud rate the MSC module is using for writing to the ABRA block (dividing the f_{SPB} with the integer division factor named “m” in the formulas below). The final output downstream baud rate is configured in the ABRA block itself. The following restriction applies: the ABRA output downstream baud rate is between half and whole intermediate baud rate.

When ABRA block is active, the fractional divider **FDR.STEP** is used only for adjusting the upstream baud rate (using the factor named “n” in the formulas below).

Attention: The following restriction applies: the upstream baud rate must be less than one tenth of the f_{SPB} . See also [Upstream Channel](#).

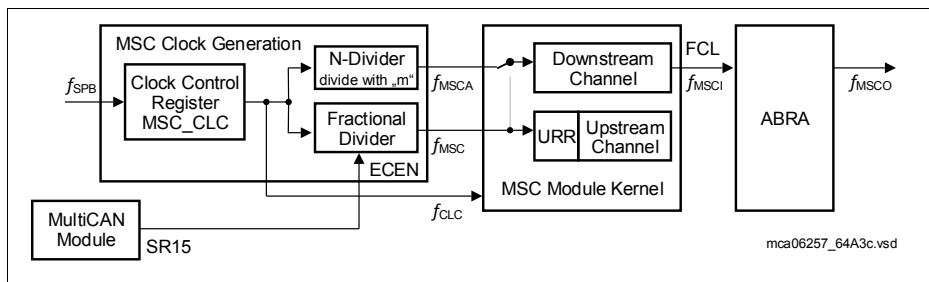


Figure 22-42 MSC Module Clock Generation

The following two formulas define the frequency of f_{MSC} :

$$f_{MSC} = f_{SPB} \times \frac{1}{n} \quad \text{with } n = 1024 - \text{MSC.FDR.STEP} \quad (22.10)$$

$$f_{MSC} = f_{SPB} \times \frac{n}{1024} \quad \text{with } n = 0-1023 \quad (22.11)$$

The following formula defines the frequency of f_{MSCA} :

$$f_{MSCA} = f_{SPB} \times \frac{1}{m} \quad \text{with } m = 1 - 16 \quad (22.12)$$

Micro Second Channel (MSC)
Intermediate Downstream Channel Baud Rate

When the ABRA block and the downstream N-divider are enabled with **ABC.ABB**, the intermediate clock signal FCL of the synchronous downstream channel going into the ABRA block is always half the frequency of f_{MSCA} :

$$\text{Baud rate}_{MSCA} = f_{SPB} \times \frac{1}{2 \times m} \quad (22.13)$$

The final output baud rate f_{MSCO} can be calculated as described in **Output Baud Rate**.

Upstream Channel Baud Rate

The baud rate of the asynchronous upstream channel is derived from the module clock f_{MSC} by a programmable clock divider selected by bit field **USR.URR** (see also **Equation (22.3)** on **Page 22-28**). The divide factor DF can be at minimum 4 and at maximum 256. Nevertheless, the minimum DF factor when using ABRA block should be 8. See the note on **Page 22-24**.

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{1}{DF \times (1024 - MSC.FDR.STEP)} \quad (22.14)$$

$$\text{Baud rate}_{MSC} = f_{SPB} \times \frac{MSC.FDR.STEP}{DF \times 1024} \quad (22.15)$$

Equation (22.10), **Equation (22.11)**, and **Equation (22.14)** are valid for normal divider mode (**FDR.DM** = 01_B). **Equation (22.11)** and **Equation (22.15)** are valid for fractional divider mode (**FDR.DM** = 10_B).

Example for Setting the Fractional Divider

This example shows how to receive upstream frames generated from 40MHz downstream clock, when the module baud-rate frequency is 100MHz/2=50MHz. The fractional divider is used to adjust the FCL clock of 50MHz to FCL clock of 40MHz, that is, it multiplies with 4/5. The best approximation of 4/5 with x/1024 is for x=819, giving an error of 0.025%. So in this case, the optimal setting for **FDR.STEP** is 819. In the next stage of the divider cascade, the DF power-of-two-divder divides the clock further to the final baud rate, matching the baud rate generated by the MSC slave device.

22.4.5.3 Fractional Divider Register

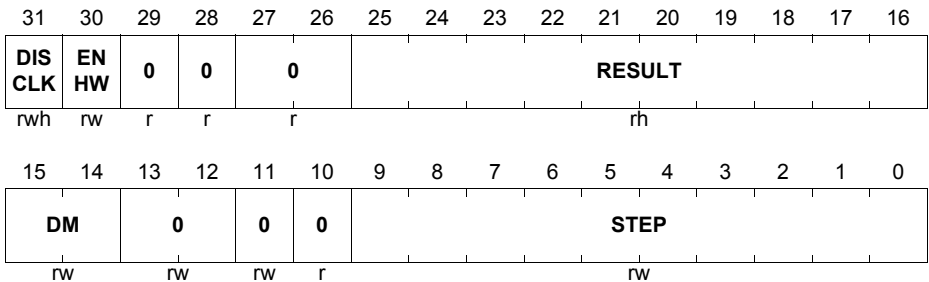
FDR

The Fractional Divider Register controls the clock rate of the shift clock f_{MSC} .

>> [Registers Overview](#)

FDR

Fractional Divider Register (0C_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
STEP	[9:0]	rw	Step Value Reload or addition value for RESULT.
DM	[15:14]	rw	Divider Mode DM selects normal or fractional divider mode.
RESULT	[25:16]	rh	Result Value Bit field for the addition result.
ENHW	30	rw	Enable Hardware Clock Control Controls operation of ECEN input and DISCLK bit.
DISCLK	31	rwh	Disable Clock Hardware controlled disable for f_{MSC} signal.
0	10, [27:26], 28, 29	r	Reserved Read as 0; shall be written with 0.
0	11, [13:12]	rw	Reserved rw bit with no function. Read as 0; shall be written with 0.

22.4.6 Port Control

MSC clock and data output lines are connected to dedicated LVDS differential output drivers. Some of the MSC module I/O lines are connected to I/O ports and therefore controlled in the port logic.

The following port control operations selections must be executed for these I/O lines:

- Input/output function selection (IOCR registers)
- Pad driver characteristics selection for the outputs (PDR registers)

22.4.7 On-Chip Connections

This section describes the on-chip connections of the MSCx modules.

22.4.7.1 EMGSTOPMSC Signal (from SCU)

The emergency stop input signal EMGSTOPMSC of the MSCx modules is connected to the output signal of the emergency stop input control logic. This logic is located in the SCU. Its functionality is controlled by the SCU emergency stop register.

Note: If the emergency stop signal is used by the MSC module, setting the SCU.EMSR.MODE=0 is mandatory. MODE=1 is not allowed to be used with the MSC module.

22.4.7.2 Interrupt Service Requests

All five service request outputs SR[4:0] are connected to the Interrupt Router Module. There they are routed further to a CPU or a DMA. The MSCx_SRCy registers are also located in the IR (Interrupt Router) Module.

22.4.7.3 Connections to Ports / Pins

These connections are described in the chapters regarding the ports and the pinning. Unused SDI signal are tied high.

Micro Second Channel (MSC)

22.4.7.4 GTM Connections

Each MSC module is connected to up to 64 TOM or ATOM outputs of the GTM module. For more detailed description of the connections for each derivative, see the section “MSC Connections” in the GTM chapter.

Table 22-11 GTM to MSC Connections

MSC Input	GTM Output
MSC0_ALTINL[15:0]	GTM_MSC0ALTINL[15:0]
MSC0_ALTINLE[15:0]	GTM_MSC0ALTINH[15:0]
MSC0_ALTINH[15:0]	GTM_MSC0ALTINH[15:0]
MSC0_ALTINHE[15:0]	GTM_MSC0ALTINL[15:0]
MSC1_ALTINL[15:0]	GTM_MSC1ALTINL[15:0]
MSC1_ALTINLE[15:0]	GTM_MSC1ALTINH[15:0]
MSC1_ALTINH[15:0]	GTM_MSC1ALTINH[15:0]
MSC1_ALTINHE[15:0]	GTM_MSC1ALTINL[15:0]

22.5 Use Case Example Micro Second Channel (MSC)

This section provides a code example for the MSC module to give an overview about the core functionality. It describes how to send data frames continuously without any software interaction, called data repetition mode. The MSC module contains two MSC serial interfaces, MSC 0 and MSC 1, each is able to connect up to four external power devices. This examples uses interface MSC 0.

Each interface can send command and data frames (see also 25.1.2.1 Frame Formats and Definitions(LINK)) via downstream channel. Between each data frame one or more passive frames can be inserted (see also Figure 25-9, Data Repetitions Mode Frame Examples (LINK)). This example realizes sending out only data frames in repetition mode with one passive frame in between. The module has more features than the example shows. For further details please see user's manual.

The MSC module can also receive frames with the upstream channel. These frames are usually status frames from the external devices. The example keeps the focus on sending out via downstream channel so the upstream channel is not part of this chapter. The initialization process follows the following order:

1. Enable MSC 0
2. Initialize MSC 0 for down streaming
3. Start sending

Step description to initialize the MSC module:

(Line 1) reset ENDINIT to get access to ENDINIT protected registers. (see also ENDINIT protection chapter 8.9.3(LINK)).

(Line 2) enable control of the module, in clock control register CLC (LINK).

(Line 3) read back (dummy variable has to be defined). The reading process ensures that the write process from line 2 is done.

(Line 4) This line loads the fractional divider register FDR(LINK). DM=01 sets the module to normal divider mode while STEP defines the Baudrate.

Note: Assuming $f_{clk}=100\text{MHz}$ and $STEP=3FE \rightarrow \text{Baudrate} = 25 \text{ Mbaud}$

(Line 5) set ENDINIT to lock the protected register again.

(Line 6) This line loads the downstream control register DSC (LINK). TM=1 sets the module to data repetition mode, NDBL and NDBH are defining the length of the SRL and SRH active phase, here set to 15 bit(see also Figure 25-6(LINK)). PPD=2 sets the length of the passive frame to $2 \times t_{FCL}$ and DIDIS=1 disables sending. Sending will be enabled after initialization.

(Line 7) This line set the number of passive time frames in the downstream status register DSS (LINK). NPTF=1 \rightarrow one passive time frame gets inserted.

Micro Second Channel (MSC)

(Line 8) The data gets loaded in the downstream data register DD (LINK). The data itself is here just an example.

(Line 9) This line resets DIDIS, so the module starts sending out data and passive time frames.

Initialization of the MSC module:

```
// enable MSC 0
(1) SCU_vResetENDINIT (0); // enable ENDINIT reg
(2) MSC0_CLC = 0x000; // disable module control
(3) dummy = MSC0_CLC; // read to check the made changes
(4) MSC0_FDR = (1 << 14) | 0x3FE; // DM=01, STEP=3FE
(5) SCU_vSetENDINIT (0); // lock ENDINIT reg
// Initialize MSC 0 for down streaming
(6) MSC0_DSC = 0x02009081; //PPD=2,DSDIS=1,NDBH=16,NDBL=16,TM=1
(7) MSC0_DSS = (1 << 8); // NPTF=1
(8) MSC0_DD = 0x5555AAAA; // downstream data (example)
// Start sending
(9) MSC0_DSC |= (1 << 15); // DIDIS=0
```


23 Controller Area Network Controller (MultiCAN+)

This chapter describes the MultiCAN+ controller of the TC26x B-Step. It contains the following sections:

- CAN basics (see [Page 23-2](#))
- Overview of the CAN Module in the TC26x B-Step (see [Page 23-12](#))
- CAN Flexible Data Rate in TC26x B-Step (see [Page 23-15](#))
- Functional description of the MultiCAN+ Kernel (see [Page 23-17](#))
- MultiCAN+ Kernel register description (see [Page 23-71](#))
- TC26x B-Step implementation-specific details (port connections and control, interrupt control, address decoding, clock control, see [Page 23-149](#)).

Table 23-1 Fixed Module Constants

Constant	Description
n_objects	Number of Message Objects available.
n_interrupts	Number of Interrupt Output Lines available.
n_pendings n_pendingregs	Number of Message Pending Bits available. There are n_pendings/32 message pending registers.
n_lists	Number of Lists available for allocation of Message Objects.
n_nodes	Number of CAN Nodes available As each CAN node has its own list in addition to the list of un-allocated elements, the relation n_nodes < n_lists is true.

Controller Area Network Controller (MultiCAN+)**23.1 CAN Basics**

CAN is an asynchronous serial bus system with one logical bus line. It has an open, linear bus structure with equal bus participants called nodes. A CAN bus consists of two or more nodes.

The bus logic corresponds to a “wired-AND” mechanism. Recessive bits (equivalent to the logic 1 level) are overwritten by dominant bits (logic 0 level). As long as no bus node is sending a dominant bit, the bus is in the recessive state. In this state, a dominant bit from any bus node generates a dominant bus state. The maximum CAN bus speed is, by definition, 1 Mbit/s. This speed limits the CAN bus to a length of up to 40 m. For bus lengths longer than 40 m, the bus speed must be reduced.

The binary data of a CAN frame is coded in NRZ code (Non-Return-to-Zero). To ensure re-synchronization of all bus nodes, bit stuffing is used. This means that during the transmission of a message, a maximum of five consecutive bits can have the same polarity. Whenever five consecutive bits of the same polarity have been transmitted, the transmitter will insert one additional bit (stuff bit) of the opposite polarity into the bit stream before transmitting further bits. The receiver also checks the number of bits with the same polarity and removes the stuff bits from the bit stream (= destuffing).

In CAN FD format frames, the CAN bit stuffing method is changed for the CRC Sequence. The stuff bits will be inserted at fixed positions.

23.1.1 Addressing and Bus Arbitration

In the CAN protocol, address information is defined in the identifier field of a message. The identifier indicates the contents of the message and its priority. The lower the binary value of the identifier, the higher is the priority of the message.

For bus arbitration, CSMA/CD with NDA (Carrier Sense Multiple Access/Collision Detection with Non-Destructive Arbitration) is used. If bus node A attempts to transmit a message across the network, it first checks that the bus is in the idle state (“Carrier Sense”) i.e. no node is currently transmitting. If this is the case (and no other node wishes to start a transmission at the same moment), node A becomes the bus master and sends its message. All other nodes switch to receive mode during the first transmitted bit (Start-Of-Frame bit). After correct reception of the message (acknowledged by each node), each bus node checks the message identifier and stores the message, if required. Otherwise, the message is discarded.

If two or more bus nodes start their transmission at the same time (“Multiple Access”), bus collision of the messages is avoided by bit-wise arbitration (“Collision Detection / Non-Destructive Arbitration” together with the “Wired-AND” mechanism, dominant bits override recessive bits). Each node that sends also reads back the bus level. When a recessive bit is sent but a dominant one is read back, bus arbitration is lost and the transmitting node switches to receive mode. This condition occurs for example when the message identifier of a competing node has a lower binary value and therefore sends a

Controller Area Network Controller (MultiCAN+)

message with a higher priority. In this way, the bus node with the highest priority message wins arbitration without losing time by having to repeat the message. Other nodes that lost arbitration will automatically try to repeat their transmission once the bus returns to idle state. Therefore, the same identifier can be sent in a Data Frame only by one node in the system. There must not be more than one node programmed to send Data Frames with the same identifier.

Standard message identifier has a length of 11 bits. CAN specification 2.0B extends the message identifier lengths to 29 bits, i.e. the extended identifier.

23.1.2 CAN Frame Formats

Four different data frame formats are supported which differ in the length of the Arbitration Field and Control Field:

- Classical CAN Base format: 11-bit long identifier, constant bit rate
- Classical CAN Extended format: 29-bit long identifier, constant bit rate
- CAN FD Base format: 11-bit long identifier, dual bit rate
- CAN FD Extended format: 29-bit long identifier, dual bit rate

In addition for Classical CAN remote frames exist, for 11-bit and 29bit identifiers.

23.1.3 CAN Frame Types

There are three types of CAN frames:

- Data Frames
- Remote Frames
- Error Frames

A Data Frame contains a Data Field of 0 to 8 bytes in length. A Remote Frame contains no Data Field and is typically generated as a request for data (e.g. from a sensor). Data and Remote Frames can use an 11-bit “Standard” identifier or a 29-bit “Extended” identifier. An Error Frame can be generated by any node that detects a CAN bus error.

23.1.3.1 Data Frames

There are four types of Data Frames defined (see [Figure 23-1](#)):

- Standard Data Frame Classical CAN Format
- Extended Data Frame Classical CAN Format
- Standard Data Frame CAN FD Format
- Extended Data Frame CAN FD Format

11-bit Data Frame (CAN Format)

A Data Frame begins with the Start-Of-Frame bit (SOF = dominant level) for hard synchronization of all nodes. The SOF is followed by the Arbitration Field consisting of 12 bits, the 11-bit identifier (reflecting the contents and priority of the message), and the

Controller Area Network Controller (MultiCAN+)

RTR (Remote Transmission Request for Classical CAN) bit. With RTR at dominant level, the frame is marked as Data Frame. With RTR at recessive level, the frame is defined as a Remote Frame.

The next field is the Control Field consisting of 6 bits. The first bit of this field is the IDE (Identifier Extension) bit and is at dominant level for the Standard Data Frame. The following bit is reserved and defined as a dominant bit. The remaining 4 bits of the Control Field are the Data Length Code (DLC) that specifies the number of bytes in the Data Field. The Data Field can be 0 to 8 bytes wide. The Cyclic Redundancy (CRC) Field that follows the data bytes is used to detect possible transmission errors. It consists of a 15-bit CRC sequence completed by a recessive CRC delimiter bit.

The final field is the Acknowledge Field. During the ACK Slot, the transmitting node sends out a recessive bit. Any node that has received an error free frame acknowledges the correct reception of the frame by sending back a dominant bit, regardless of whether or not the node is configured to accept that specific message. This behavior assigns the CAN protocol to the “in-bit-response” group of protocols. The recessive ACK delimiter bit, which must not be overwritten by a dominant bit, completes the Acknowledge Field.

Seven recessive End-of-Frame (EOF) bits finish the Data Frame. Between any two consecutive frames, the bus must remain in the recessive state for at least 3 bit times (called Inter Frame Space). If after the Inter Frame Space, no other nodes attempt to transmit the bus remains in idle state with a recessive level.

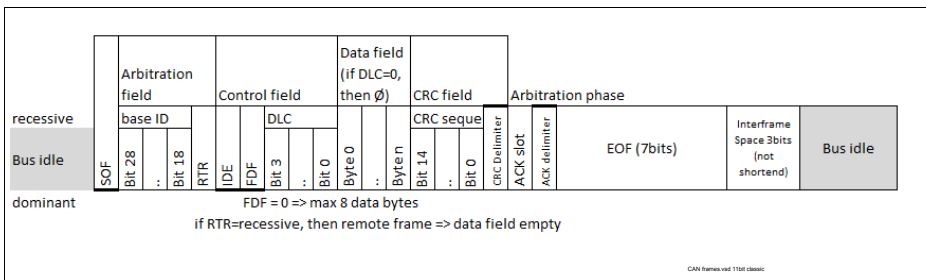


Figure 23-1 Classical 11bit ID CAN Data Frame

Extended Data Frame (Classical CAN Format)

In the Extended CAN Data Frame, the message identifier of the standard frame has been extended to 29-bit. A split of the extended identifier into two parts, an 11-bit least significant section (as in classical CAN frame) and an 18-bit most significant section, ensures that the Identifier Extension bit (IDE) can remain at the same bit position in both standard and extended frames.

In the Extended CAN Data Frame, the SOF bit is followed by the 32-bit Arbitration Field. The first 11 bits are the least significant bits of the 29-bit Identifier (“Base-ID”). These 11 bits are followed by the recessive Substitute Remote Request (SRR) bit. The SRR is

Controller Area Network Controller (MultiCAN+)

further followed by the recessive IDE bit, which indicates the frame to be an Extended CAN frame. If arbitration remains unresolved after transmission of the first 11 bits of the identifier, and if one of the nodes involved in arbitration is sending a classical CAN frame, then the CAN frame will win arbitration due to the assertion of its dominant IDE bit. Therefore, the SRR bit in an Extended CAN frame is recessive to allow the assertion of a dominant RTR bit by a node that is sending a CAN Remote Frame. The SRR and IDE bits are followed by the remaining 18 bits of the extended identifier and the RTR bit. Control field and frame termination is identical to the Classical Data Frame.

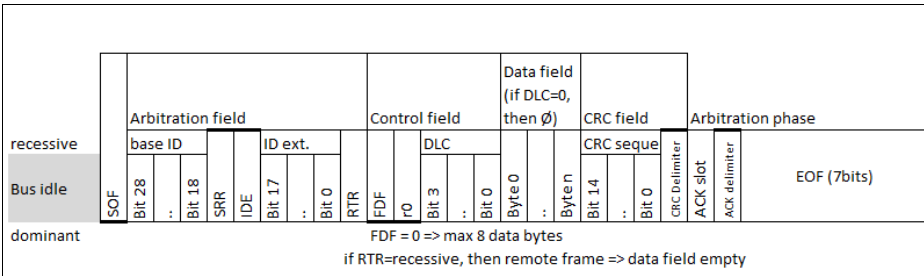


Figure 23-2 Classical 29 bit ID CAN Data Frame

Standard and Extended Data Frame (CAN FD Format)

Data frames for CAN FD with 11bit identifier are shown in [Figure 23-3](#).

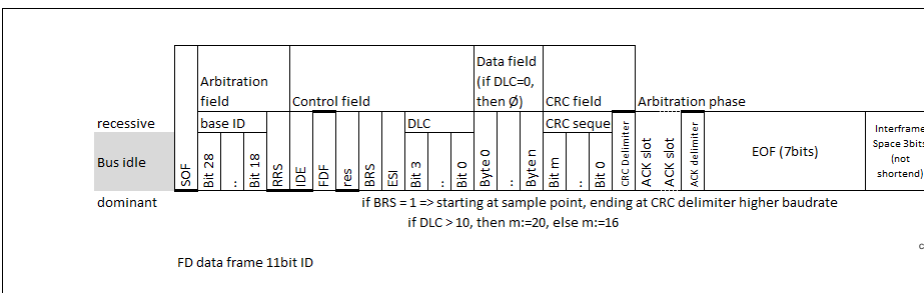
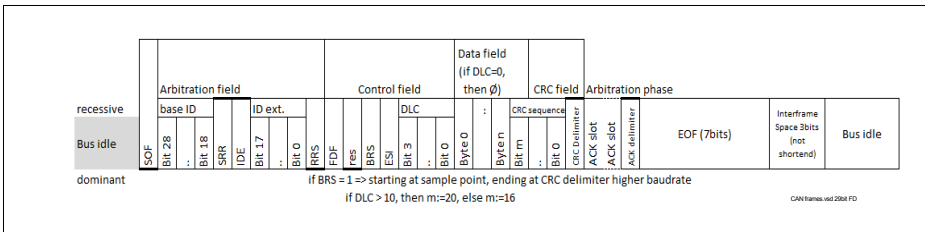


Figure 23-3 CAN FD 11bit ID Data Frames

Extended data frames for CAN FD are shown in [Figure 23-4](#).

Controller Area Network Controller (MultiCAN+)

Figure 23-4 CAN FD 29bit ID Data Frames

The difference between Standard / Extended Data Frames in CAN and CAN FD format are highlighted below:

1. In the Arbitration Field:
 - a) CAN Format Frames contain the RTR bit, where in CAN FD Format it is replaced with the dominant r1(reserved) bit.
 - b) The reserved r0, r1 (bits) are sent dominant. Receivers accept dominant and recessive bits.
2. In the Control Field:
 - a) CAN FD Format Frames consists of the additional (FDF) CAN FD format bit, (BRS) bit rate switch bit and (ESI) error state indicator bit.
 - b) CAN FD format (FDF) bit comes after the IDE bit for frames with 11-bit identifier and it comes after the r1 (reserved) bit with 29-bit identifier. FDF is the new name for the previous r0 bit.
 - c) Bit Rate Switch (BRS) bit switches the bit rate from standard bit rate of Arbitration Phase to preconfigured bit rate of the Data Phase when the bit is transmitted recessive. The bit rate is not switched when BRS bit is transmitted dominant.
 - d) Error State Indicator (ESI) bit is transmitted dominant by error active nodes and recessive by error passive nodes.
 - e) Data Length Code (DLC) bits indicates the number of bytes in the Data Field.
3. In the CRC Field:
 - a) CRC sequence bits for CAN FD uses CRC_17 for data field up to sixteen bytes long and CRC_21 for data field longer than sixteen bytes. CRC calculation consists of the SOF, Arbitration Field, Control Field and (if present) Data field, supplemented with nCRC bits of '0'.
 Stuff-bits are included in CRC calculation. All CRC sequences (CRC_15, 17, 21) are calculated for all nodes, where the node that wins the arbitration send the CRC sequence selected by values of the FDF bit and DLC. Receivers only consider the selected CRC polynomial to check for CRC error
 - b) CRC Delimiter for CAN FD consists of one or two recessive bits that has the function of switching the Data phase to Arbitration phase when the sample point reaches the first bit of the CRC Delimiter.
 Transmitter sends only one recessive bit as CRC delimiter, but accepts two

Controller Area Network Controller (MultiCAN+)

recessive bits before the edge from recessive to dominant of the Acknowledge slot. Receiver sends Acknowledge bit after the first CRC Delimiter.

- c) Due to a weakness of the CRC in a certain situation, and additional checksum is needed. For example a software checksum inside the last data byte. This is a weakness in the standard and not of the module. Therefore with DIS2015 the CRC field of CAN FD will be changed.
4. In the ACK Field:
 - a) For ACK slot, CAN FD nodes accept a two bit long dominant phase of overlapping ACK bits as a valid ACK, to compensate for phase shifts between Receivers.

23.1.3.2 Remote Frames

Normally, data transmission is performed on an autonomous basis with the data source node (e.g. a sensor) sending out a Data Frame. It is also possible, however, for a destination node (or nodes) to request the data from the source. For this purpose, the destination node sends a Remote Frame with an identifier that matches the identifier of the required Data Frame. The appropriate data source node will then send a Data Frame as a response to this remote request.

There are 2 differences between a Remote Frame and a Data Frame.

- The RTR bit is in the recessive state in a Remote Frame.
- There is no Data Field in a Remote Frame.

If a Data Frame and a Remote Frame with the same identifier are transmitted at the same time, the Data Frame wins arbitration due to the dominant RTR bit following the identifier. In this way, the node that transmitted the Remote Frame receives the requested data immediately.

Remote frames are only defined in Classical CAN format. Remote frames and the corresponding RTR bit do not exist for CAN FD format. Remote frame requests do not change the format of the preconfigured reply data frames (i.e. FDF and BRS bits of preconfigured data frames are not changed on remote frame requests, reply data frames may be in CAN base/extended or CAN FD base/extended).

23.1.3.3 Error Frames

An Error Frame is generated by any node that detects a bus error. An Error Frame consists of two fields, an Error Flag field followed by an Error Delimiter field. The Error Delimiter Field consists of 8 recessive bits and allows the bus nodes to restart bus communications after an error. There are, however, two forms of Error Flag fields. The form of the Error Flag field depends on the error status of the node that detects the error.

When an error-active node detects a bus error, the node generates an Error Frame with an active-error flag. The error-active flag is composed of six consecutive dominant bits that actively violate the bit-stuffing rule. All other stations recognize a bit-stuffing error and generate Error Frames themselves. The resulting Error Flag field on the CAN bus

Controller Area Network Controller (MultiCAN+)

therefore consists of six to twelve consecutive dominant bits (generated by one or more nodes). The Error Delimiter field completes the Error Frame. After completion of the Error Frame, bus activity returns to normal and the interrupted node attempts to re-send the aborted message.

If an error-passive node detects a bus error, the node transmits an error-passive flag followed, again, by the Error Delimiter field. The error-passive flag consists of six consecutive recessive bits, and therefore the Error Frame (for an error-passive node) consists of 14 recessive bits (i.e. no dominant bits). Therefore, the transmission of an Error Frame by an error-passive node will not affect any other node on the network, unless the bus error is detected by the node that is actually transmitting (i.e. the bus master). If the bus master node generates an error-passive flag, this may cause other nodes to generate Error Frames due to the resulting bit-stuffing violation. After transmission of an Error Frame an error-passive node must wait for 6 consecutive recessive bits on the bus before attempting to rejoin bus communications.

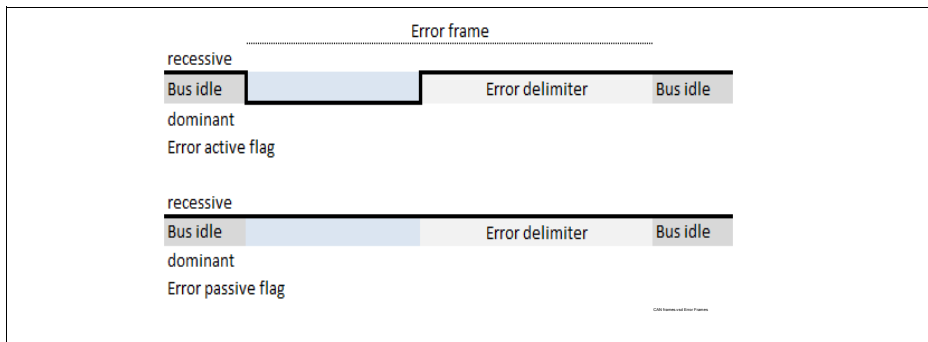


Figure 23-5 CAN Error Frames

A CAN FD node operating in the Data-Phase will switch back to the Arbitration Phase when starting an Error Flag.

23.1.3.4 Overload Frame

The Overload Frame consists of two bit fields which are Overload Flag and Overload Delimiter.

Overload conditions

If a CAN FD node samples a dominant bit at the eight bit (last bit) of an Error Delimiter or Overload Delimiter, or if a CAN FD Receiver samples a dominant bit at the last bit of End of Frame, it will start transmitting an Overload Frame (not an Error Frame). The Error Counters will not be incremented.

Controller Area Network Controller (MultiCAN+)
23.1.4 The Nominal Bit Time

One bit cell (this means one high or low pulse of the NRZ code) is composed by four segments. Each segment is an integer multiple of Time Quanta t_Q . The Time Quanta is the smallest discrete timing resolution used by a CAN node. The nominal bit time definition with its segments is shown in [Figure 23-6](#).

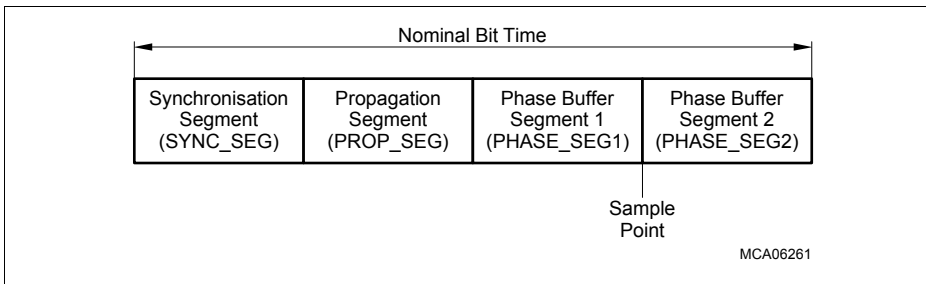


Figure 23-6 Partition of Nominal Bit Time

The Synchronization Segment (SYNC_SEG) is used to synchronize the various bus nodes. If there is a bit state change between the previous bit and the current bit, then the bus state change is expected to occur within this segment. The length of this segment is always $1 t_Q$.

The Propagation Segment (PROP_SEG) is used to compensate for signal delays across the network. These delays are caused by signal propagation delay on the bus line and through the electronic interface circuits of the bus nodes.

The Phase Segments 1 and 2 (PHASE_SEG1, PHASE_SEG2) are used to compensate for edge phase errors. These segments can be lengthened or shortened by re-synchronization. PHASE_SEG2 is reserved for calculation of the subsequent bit level, and is $\geq 2 t_Q$. At the sample point, the bus level is read and interpreted as the value of the bit cell. It occurs at the end of PHASE_SEG1.

The total number of t_Q in a bit time is between 8 and 25.

As a result of re-synchronization, PHASE_SEG1 can be lengthened or PHASE_SEG2 can be shortened. The amount of lengthening or shortening the phase buffer segments has an upper limit given by the re-synchronization jump width. The re-synchronization jump width may be between 1 and $4 t_Q$, but it may not be longer than PHASE_SEG1.

23.1.4.1 CAN FD bit timing

The first part of a CAN FD frame until the BRS bit is transmitted with the Nominal Bit Rate. The bit rate is switched if the BRS bit is recessive, until the CRC Delimiter is reached or until the CAN FD controller sees an error condition that results in the starting

Controller Area Network Controller (MultiCAN+)

of an Error Frame. CAN FD Error Frames, as well as ACK Field, End of Frame, Overload Frames and all in CAN Format are transmitted with the Nominal Bit Rate

Synchronization Rules

Hard synchronization and Resynchronization are the two forms of synchronization. They obey the following rules,

- Hard synchronization is performed whenever there is a recessive to dominant edge during Bus Idle, Suspend Transmission, and second or third bits of Intermission. Hard synchronization is also performed at the recessive to the dominant edge from FDF to r0 in CAN FD format frames
- A transmitter shall not resynchronize while it transmits in the CAN FD data phase

23.1.5 Error Detection and Error Handling

The CAN protocol has sophisticated error detection mechanisms. The following errors can be detected:

- **Cyclic Redundancy Check (CRC) Error**

With the CRC, the transmitter calculates special check bits for the bit sequence from the start of a frame until the end of the Data Field. This CRC sequence is transmitted in the CRC Field. The receiving node also calculates the CRC sequence using the same formula, and performs a comparison to the received sequence. If a mismatch is detected, a CRC error has occurred and an Error Frame is generated. The message is repeated.

- **Acknowledge Error**

In the Acknowledge Field of a message, the transmitter checks whether a dominant bit is read during the Acknowledge Slot (that is sent out as a recessive bit). If not, no other node has received the frame correctly, an Acknowledge Error has occurred, and the message must be repeated. No Error Frame is generated.

- **Form Error**

If a transmitter detects a dominant bit in one of the four segments End of Frame, Interframe Space, Acknowledge Delimiter, or CRC Delimiter, a Form Error has occurred, and an Error Frame is generated. The message is repeated.

- **Bit Error**

A Bit Error occurs if a) a transmitter sends a dominant bit and detects a recessive bit or b) if the transmitter sends a recessive bit and detects a dominant bit when monitoring the actual bus level and comparing it to the just transmitted bit. In case b), no error occurs during the Arbitration Field (ID, RTR, IDE) and the Acknowledge Slot.

- **Stuff Error**

If between Start of Frame and CRC Delimiter, six consecutive bits with the same polarity are detected, the bit-stuffing rule has been violated. A stuff error occurs and an Error Frame is generated. The message is repeated.

Controller Area Network Controller (MultiCAN+)

Detected errors are made public to all other nodes via Error Frames (except Acknowledge Errors). The transmission of the erroneous message is aborted and the frame is repeated as soon as possible. Furthermore, each CAN node is in one of the three error states (error-active, error-passive or bus-off) according to the value of the internal error counters. The error-active state is the usual state where the bus node can transmit messages and active-error frames (made of dominant bits) without any restrictions. In the error-passive state, messages and passive-error frames (made of recessive bits) may be transmitted. The bus-off state makes it temporarily impossible for the node to participate in the bus communication. During this state, messages can be neither received nor transmitted.

Basic CAN, Full CAN

There is one more CAN characteristic that is related to the interface of a CAN module (controller) and the host CPU: Basic-CAN and Full-CAN functionality.

In Basic-CAN devices, only basic functions of the protocol are implemented in hardware, such as the generation and the check of the bit stream. The decision, whether a received message has to be stored or not (acceptance filtering), and the complete message management must be done by software.

Full-CAN devices (this is the case for the MultiCAN+ controller as implemented in TC26x B-Step) manage the whole bus protocol in hardware, including the acceptance filtering and message management. Full-CAN devices contain message objects that handle autonomously the identifier, the data, the direction (receive or transmit) and the information of CAN operation. During the initialization of the device, the host CPU determines which messages are to be sent and which are to be received. The host CPU is informed by interrupt if the identifier of a received message matches with one of the programmed (receive-) message objects. The CPU load of Full-CAN devices is greatly reduced. When using Full-CAN devices, high baud rates and high bus loads with many messages can be handled.

Controller Area Network Controller (MultiCAN+)**23.2 Overview**

The MultiCAN+ module provides a communication interface which is fully compliant with CAN specification V2.0B (active) and to CAN FD ISO11898-1 DIS version 2014, providing communications at up to 1 Mbit/s in Classical CAN (ISO 11898-1:2003(E) mode and/or CAN FD until 5 MBaud dataspeed (dependent on frequency and nodes)).

The MultiCAN+ module for the TC26x B-Step consists of module (i.e MultiCAN with 5 CAN nodes), representing serial communication interfaces. Each CAN node communicates over two pins (TXD and RXD). The device ports which are used for TXD and RXD may be individually configured within the PORTS block. Several port configuration options are available to provide application-specific flexibility.

The MultiCAN+ module contains 5 independently operating CAN nodes with Full-CAN functionality that are able to exchange Data and Remote Frames via a gateway function. Each CAN node can receive and transmit standard frames with 11-bit identifiers as well as extended frames with 29-bit identifiers.

All CAN nodes share a common set of 256 message objects. Each message object can be individually allocated to one of the CAN nodes. Besides serving as a storage container for incoming and outgoing frames, message objects can be combined to build gateways between the CAN nodes or to setup a FIFO buffer.

The message objects are organized in double-chained linked lists, where each CAN node has its own list of message objects. A CAN node stores frames only into message objects that are allocated to the message object list of the CAN node, and it transmits only messages belonging to this message object list. A powerful, command-driven list controller performs all message object list operations.

The bit timings for the CAN nodes are derived from the module timer clock (f_{CAN}) and are programmable up to a data rate of 1 Mbit/s in Classical CAN (ISO 11898-1:2003(E) mode or up to 5 MBaud in CAN FD mode. External bus transceivers are connected to a CAN node via a pair of receive and transmit pins.

23.2.1 Features List

The MultiCAN+ module provides the following features:

- Compliant with ISO 11898 and SAE J 1939
- CAN functionality according to CAN specification V2.0 B active
- Supports CAN with Flexible Data-Rate Specification CAN FD DIS version ISO11898-1 2014 with max. 64 data bytes¹⁾
- Dedicated control registers for each CAN node
- Data transfer rates up to 1 Mbit/s when operating in Classical CAN mode per ISO 11898-1:2003(E)

1) At time of release of this document ISO 11898-1 (Classical CAN and CAN FD) is in working draft mode, an errata sheet will document differences when ISO 11898-1 has been finalized.

Controller Area Network Controller (MultiCAN+)

- Supports up to 5MBaud, when operating in CAN FD mode.
- Support for asynchronous clock sources for baud rate generation by providing separate frequency domain and input:
 - System frequency clock f_{CLC}
 - Low-jitter E-Ray PLL clock
 - Direct oscillator clock (e.g. from ceramic resonator)
- Frequency jitter calibration based on external CAN messages during runtime
- Flexible and powerful message transfer control and error handling capabilities
- Advanced CAN bus bit timing analysis and baud rate detection for each CAN node via a frame counter
- Full-CAN functionality: A set of 256 message objects can be individually
 - Allocated (assigned) to any CAN node
 - Configured as transmit or receive object
 - Setup to handle frames with 11-bit or 29-bit identifier
 - Identified by a timestamp via a frame counter
 - Configured to remote monitoring mode
- Advanced Acceptance Filtering
 - Each message object provides an individual acceptance mask to filter incoming frames
 - A message object can be configured to accept standard or extended frames or to accept both standard and extended frames
 - Message objects can be grouped into different priority classes for transmission and reception
 - The selection of the message to be transmitted first can be based on frame identifier, IDE bit and RTR bit according to CAN arbitration rules, or on its order in the list
- Advanced CAN node features
 - Analyzer Mode supports monitoring of bus traffic without actively participating on the bus
 - Internal Loop-Back Mode is available for test purposes
 - An internal timer is provided that can detect a receive or remote frame time-out; the message objects to be supervised are selectable
 - Data transmission from a node can be stopped without affecting reception
 - Programmable minimum delay between two consecutive messages
- Advanced message object functionality
 - Message objects can be combined to build FIFO message buffers of arbitrary size, limited only by the total number of message objects
 - Message objects can be linked to form a gateway that automatically transfers frames between 2 different CAN buses. A single gateway can link any two CAN nodes. An arbitrary number of gateways can be defined
- Advanced data management
 - The message objects are organized in double-chained lists

Controller Area Network Controller (MultiCAN+)

- List reorganizations can be performed at any time, even during full operation of the CAN nodes
- A powerful, command-driven list controller manages the organization of the list structure and ensures consistency of the list
- Message FIFOs are based on the list structure and can easily be scaled in size during CAN operation
- Advanced interrupt handling
 - Message interrupts, node interrupts can be generated
 - Interrupt requests can be routed individually to one of the 16 interrupt output lines
 - Message post-processing notifications can be combined flexibly into a dedicated register field of 256 notification bits
- Module internal SRAM with ECC protection
- Pretended Networking Operation for energy saving purposes
 - If direct oscillator clock is selected for baud rate generation, CPU clock can be slowed down or CPU Idle Mode can be entered
 - Three selectable message object can be transmitted periodically without CPU involvement, triggered by a CAN node timer or by STM or GTM modules.
 - Message objects are received without CPU involvement and can wake up the system via receive interrupt

AUTOSAR optimized and backward compatible with existing MultiCAN software

Controller Area Network Controller (MultiCAN+)**23.3 CAN Flexible Data-Rate (CAN FD)**

CAN Flexible Data-Rate (CAN FD) builds on existing CAN (ISO 11898-1) specifications allowing higher data rates and larger payloads. This is achieved with a new CAN FD frame format different from existing Classical CAN format, both frame formats can coexist within the same network. Classical CAN nodes and CAN FD nodes can communicate with each other as long as CAN FD frame format is not being used.

CAN FD functionality is available on all nodes of MultiCAN+ module.

Feature Lists

- Supports CAN Flexible Data-Rate Specification V1.0

23.3.1 Transmitter Delay Compensation

The CAN protocol requires that transmitted data be compared with the received data from its local CAN transceiver to determine if there are any transmit errors. In the case of CAN FD data phase when a faster bit rate is used, resulting in shorter bit timing, the delay caused by the local transmitting loop delay will be greater than TSEG1 (time segment before sample point) causing a bit error to be detected. The transmitter loop delay limits the bit rate in the data phase of a CAN FD frame.

Thus to overcome this limitation, a Transmitter Delay Compensation feature is introduced where a new sample point (Secondary Sample Point) shall be used by transmitters in the data phase of a CAN FD frame, the sample point which does not account for the transceiver loop delay is ignored.

The Secondary Sample Point consists of the transmitter loop delay and a configurable Transmitter Delay Compensation offset (NTDCRx.TDCO). (i.e The secondary sample point is reached by counting the total delay consisting of the compensation offset and measured transceiver loop delay.)

The Transmitter loop delay is measured in each transmitted frame at the edge from the FDF bit to the following bit r0, between the edge of the transmitted bit and the edge of the received bit. (see [Figure 23-7](#)) The count down is started with the begin of the bit time (transmit point).

Controller Area Network Controller (MultiCAN+)

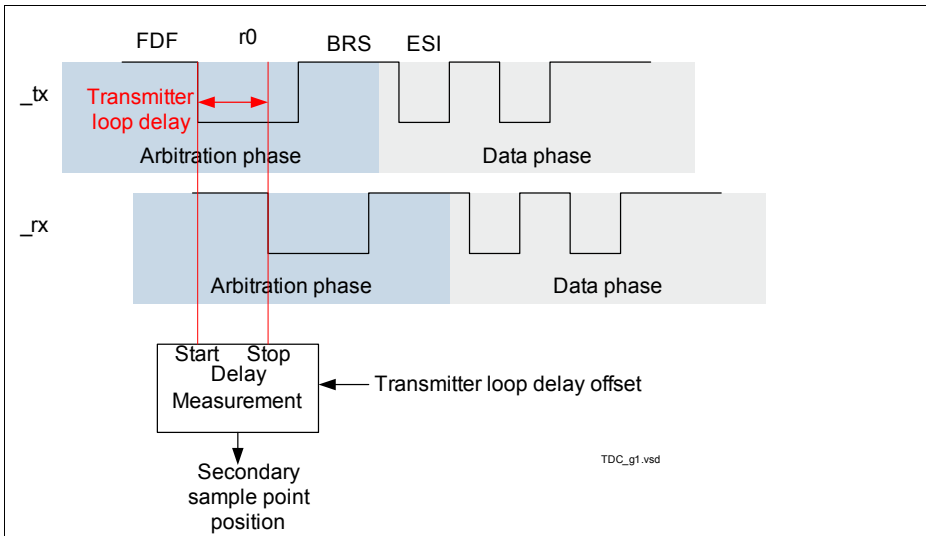


Figure 23-7 Transmitter Delay Loop measurement

Transmitter Delay compensation offset (NTDCRx.TDCO) is used to adjust the secondary sample point inside the bit time (e.g half of the bit time in the data phase). Finally the resulting Secondary Sample Point is rounded down to the next integer number of time quanta t_q and placed after the end of the transmitted bit. If a bit error is detected at the Secondary Sample Point, the transmitter will react to this bit error at the next sample point.

(i.e When a bit error is detected by the Transmitter Delay compensation unit, the error is reported at the next sample point and becomes visible (in error active case) at the transmit point that follows the regular sample point.)

Secondary Sample Point is used in the Data Phase of CAN FD and Sample Point is used in the Arbitration Phase of CAN FD. The Transmitter Delay compensation which determines the secondary sample point is able to handle a total delay (measured transceiver loop delay + compensation offset) which goes beyond the current bit time.

MultiCAN+ module allows the secondary sample point to be placed anywhere within the current and next 3 bit times (i.e covers up to 4 data phase bit rate). The maximum delay which can be compensated by MultiCAN+ delay compensation during the data phase is 4 bit times.

Note:

9. CAN receive input line contributes to the measured loop delay.
10. Measurement granularity of the Transmitter Delay compensation is the fast time quantum given by the fast baud rate prescaler.

Controller Area Network Controller (MultiCAN+)

23.4 MultiCAN+ Kernel Functional Description

This section describes the functionality of the MultiCAN+ module.

23.4.1 Module Structure

Figure 23-8 shows the general structure of the MultiCAN+ module.

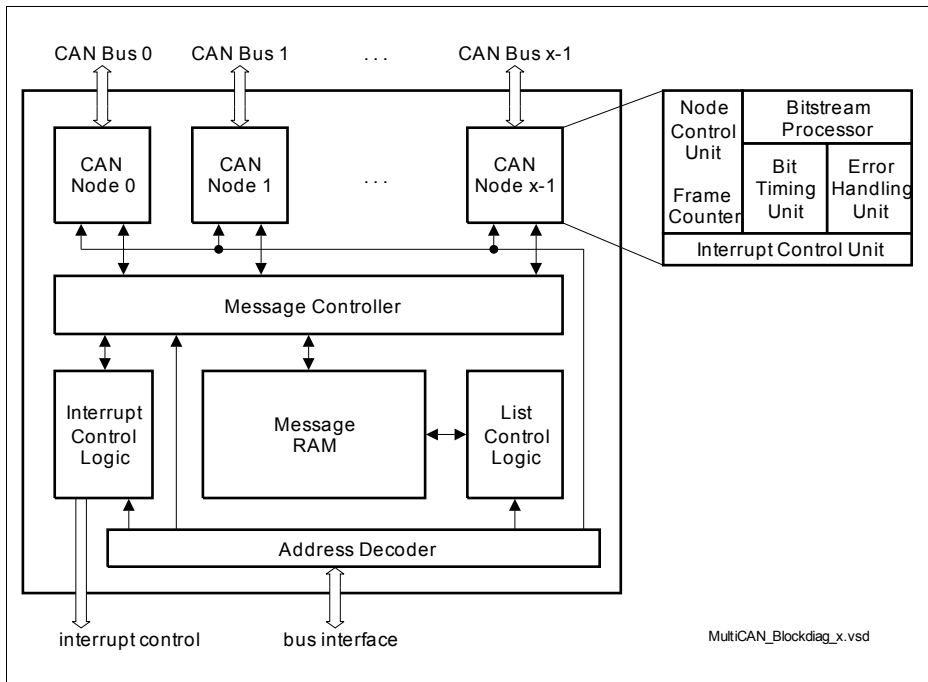


Figure 23-8 MultiCAN+ Block Diagram

CAN Nodes

Each CAN node consists of several sub-units.

- Bitstream Processor**
 The Bitstream Processor performs data, remote, error and overload frame processing according to the ISO 11898 standard. This includes conversion between the serial data stream and the input/output registers.
- Bit Timing Unit**
 The Bit Timing Unit determines the length of a bit time and the location of the sample point according to the user settings, taking into account propagation delays and phase shift errors. The Bit Timing Unit also performs resynchronization.

Controller Area Network Controller (MultiCAN+)

- **Error Handling Unit**

The Error Handling Unit manages the receive and transmit error counter. Depending on the contents of both counters, the CAN node is set into an error-active, error passive or bus-off state.

- **Node Control Unit**

The Node Control Unit coordinates the operation of the CAN node:

- Enable/disable CAN transfer of the node
- Enable/disable and generate node-specific events that lead to an interrupt request (CAN bus errors, successful frame transfers etc.)
- Administration of the frame counter and of the node timers

- **Interrupt Control Unit**

The Interrupt Control Unit in the CAN node controls the interrupt generation for the different conditions that can occur in the CAN node.

Message Controller

The Message Controller handles the exchange of CAN frames between the CAN nodes and the message objects that are stored in the Message RAM. The Message Controller performs several functions:

- Receive acceptance filtering to determine the correct message object for storing of a received CAN frame
- Transmit acceptance filtering to determine the message object to be transmitted first, individually for each CAN node
- Transfer contents between message objects and the CAN nodes, taking into account the status/control bits of the message objects
- Handling of the FIFO buffering and gateway functionality
- Aggregation of message-pending notification bits

List Controller

The List Controller performs all operations that lead to a modification of the double-chained message object lists. Only the list controller is allowed to modify the list structure. The allocation/deallocation or reallocation of a message object can be requested via a user command interface (command panel). The list controller state machine then performs the requested command autonomously.

Interrupt Control

The general interrupt structure is shown in [Figure 23-9](#). The interrupt event can trigger the interrupt generation. The interrupt pulse is generated independently of the interrupt flag in the interrupt status register. The interrupt flag can be reset by software by writing a 0 to it.

Controller Area Network Controller (MultiCAN+)

If enabled by the related interrupt enable bit in the interrupt enable register, an interrupt pulse can be generated at one of the 16 interrupt output lines INT_0m of the MultiCAN+ module. If more than one interrupt source is connected to the same interrupt node pointer (in the interrupt node pointer register), the requests are combined to one common line.

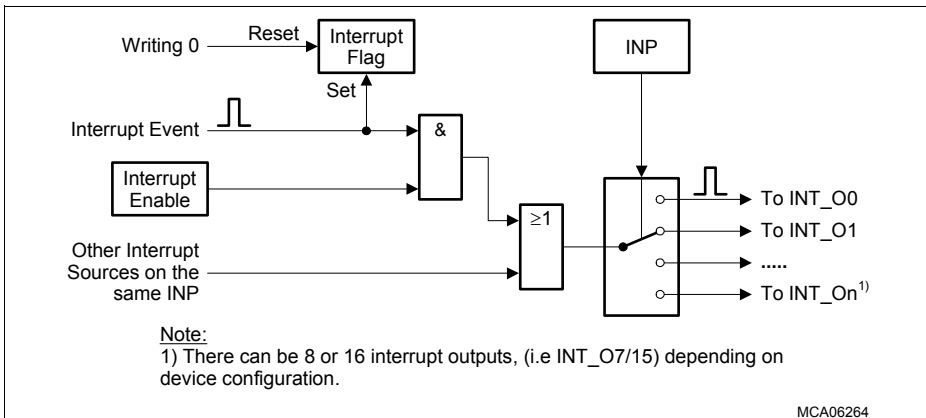


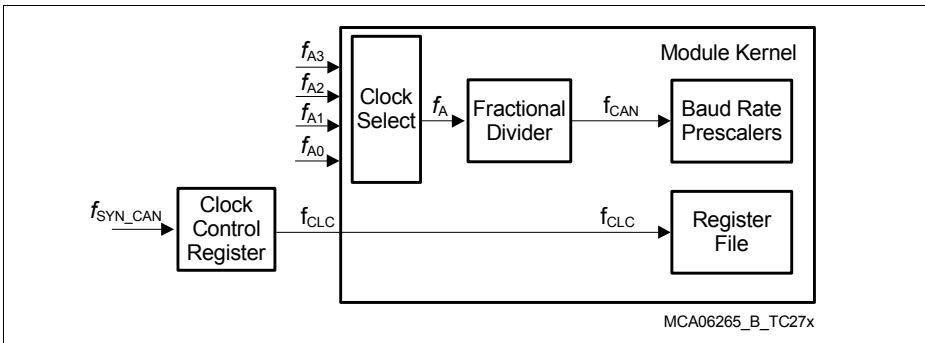
Figure 23-9 General Interrupt Structure

23.4.2 Clock Control

The CAN module timer clock f_{CAN} of the functional blocks of the MultiCAN+ module is derived from the asynchronous, higher precision clock f_A or from the synchronous clock source. The Fractional Divider is used to generate f_{CAN} used for bit timing calculation, The frequency of f_{CAN} is identical for all CAN nodes. The register file operate with the module control clock f_{CLC} . See also **“MultiCAN+ Clock Generation” on Page 23-20**.

The output clock f_{CAN} of the Fractional Divider is based on the clock f_A , but only every nth clock pulse is taken. The suspend signal (coming as acknowledge from the MultiCAN+ module in response to a OCDS suspend request) freezes or resets the Fractional Divider. The clock select and fractional divider are reset together with the CAN module kernel when the module kernel reset is triggered.

Controller Area Network Controller (MultiCAN+)


Figure 23-10 MultiCAN+ Clock Generation

The $f_{\text{SYN_CAN}}$ is identical to f_{SPB} . f_{Ai} is the asynchronous clock input.

Table 23-2 indicates the minimum operating frequencies in MHz for f_{CLC} that are required for a baud rate of 1 Mbit/s for the active CAN nodes. If a lower baud rate is desired, the values can be scaled linearly (e.g. for a maximum of 500 kbit/s, 50% of the indicated value are required).

For CAN FD operations, please refer to **Table 23-3** for the minimum operating frequency in MHz for f_{CLC} that are required.

The values imply that the CPU (or DMA) executes maximum accesses to the MultiCAN+ module. The values may contain rounding effects.

Controller Area Network Controller (MultiCAN+)
Table 23-2 Minimum Operating Frequencies¹⁾²⁾ [MHz]

Number of allocated message objects MO ³⁾ ,	Number of Active CAN Nodes				
	1	2	3	4	5
16 MO	12	19	26	33	40
32 MO	15	23	30	37	44
64 MO	21	28	37	46	53
128 MO	40	45	50	55	61
256 MO	72	77	82	88	93

1) In the case of 15 time quanta, the minimum operating frequency required is 15MHz.

2) To guarantee the minimum operating frequencies when low power mode (pretended networking mode) is enabled (i.e when frequency of f_{SPB} is reduced), f_{SYN_CAN} and f_{CLC} are switched from f_{SPB} to f_{CAN} when SCU register CCUCON.LPDIV>0.

3) Only those message objects have to be taken into account that are allocated to a CAN node. The unallocated message objects have no influence on the minimum operating frequency.

Controller Area Network Controller (MultiCAN+)
Table 23-3 Minimum Operating Frequencies for CAN FD [MHz]

No. of allocated MO ¹⁾	No. of Active CAN Nodes	Acceleration Factor ²⁾³⁾				
		1	2	3	4	5
16 MO	1	20	40	60	80	100
	2	39	48	72	96	-
	3	58	58	84	-	-
	4	77	77	96	-	-
	5	96	96	-	-	-
32 MO	1	20	40	60	80	100
	2	39	48	72	96	-
	3	58	58	84	-	-
	4	77	77	96	-	-
	5	96	96	-	-	-
64 MO	1	27	40	60	80	100
	2	39	51	72	96	-
	3	58	65	84	-	-
	4	77	79	96	-	-
	5	96	96	-	-	-
128 MO	1	47	65	75	81	100
	2	57	79	91	99	-
	3	67	93	-	-	-
	4	77	-	-	-	-
	5	96	-	-	-	-

1) Only those message objects have to be taken into account that are allocated to a CAN node. The unallocated message objects have no influence on the minimum operating frequency.

When message objects are configured as transmit or receive FIFO structures i.e MOFCRn.MMC = 0001/0010, only the base object of 1 is counted towards the minimum frequency requirement, all other slave objects on the FIFO do not count towards the minimum frequency requirement. See [Section 23.4.12.5](#)

2) Acceleration Factor is the ratio of the Data Bit Rate to Nominal Bit Rate. The Nominal Bit Rate is taken as 1Mbit/s in table above. As an example, an acceleration factor(A.F) of 4 refers to a Data Bit Rate of 4Mbit/s and Nominal Bit Rate is 1Mbit/s.

Only the node operating with the highest baudrate need to be considered when several nodes operate at different acceleration factor. e.g When 1node operate at A.F of 3 and 3 nodes operate at A.F of 2, the minimum operating frequency required is determined by the node operate at A.F of 3.

3) Please note that other combinations of acceleration factor are possible, values here are for illustrative purposes only. Values with dash '-' indicates required f_{CLC} exceeding the maximum frequency capability of product SCU typically at 100MHz.

Controller Area Network Controller (MultiCAN+)

The baud rate generation of the MultiCAN+ being based on f_A , this frequency has to be chosen carefully to allow correct CAN bit timing. The required value of f_A is given by an integer multiple (n) of the CAN baud rate multiplied by the number of time quanta per CAN bit time. For example, to reach 1 Mbit/s with 20 tq per bit time, possible values of f_A are given by formula $[n \times 20]$ MHz, with n being an integer value, starting at 1.

In order to minimize jitter, it is not recommended to use the fractional divider mode for high baud rates.

Additionally, for correct operation of the MultiCAN, the following conditions have to be fulfilled,

$$\text{Baudrate}_{\max} = [(8 \times T_{\text{CAN}}) + (8 \times T_{\text{CLC}}) + (4 \times \text{No. of active CAN nodes} \times T_{\text{CLC}})] \quad (23.1)$$

also

$$\text{NBTR.SJW} < \text{NBTR.TSEG1}$$

As an example, when $f_{\text{CLC}} = 10\text{MHz}$, $f_{\text{CAN}} = 20\text{MHz}$, No of active CAN nodes =2,

$$\text{Baudrate}_{\max} = [(8 \times 50\text{ns}) + (8 \times 100\text{ns}) + (4 \times 2 \times 100\text{ns})] = 2000\text{ns} = 500 \text{KBaud}$$

Table 23-4 below illustrates the minimum CAN module timer clock f_{CAN} and Module Control Clock f_{CLC} that's required to support a baudrate generation of 500KBaud. If a higher baudrate is desired, the values need to be calculated as per **Equation (23.1)**.

Table 23-4 Minimum Operating Frequencies [MHz] required for 500KBaud

No. of active CAN nodes	$f_{\text{CAN}} = f_{\text{CLC}}$ (MHz)	$f_{\text{CAN}} \neq f_{\text{CLC}}$ (MHz) ¹⁾	
		f_{CAN}	f_{CLC}
1	10	16	8
		20	8
		24	8
		80	7
2	12	16	11
		20	10
		24	10
		80	9
3	14	16	14
		20	13
		24	12
		80	11

Controller Area Network Controller (MultiCAN+)
Table 23-4 Minimum Operating Frequencies [MHz] required for 500KBaud

4	16	16	16
		20	15
		24	15
		80	13

1) To guarantee the minimum operating frequencies when low power mode (pretended networking mode) is enabled (i.e. when frequency of f_{SPB} is reduced), f_{SYN_CAN} and f_{CLC} are switched from f_{SPB} to f_{CAN} when SCU register CCUCON.LPDIV>0.

23.4.3 Port Input Control

It is possible to select the input lines for the RXDCANx inputs for the CAN nodes. The selected input is connected to the CAN node and is also available to wake-up the system. More details are defined in [Section 23.7.5.2](#) on [Page 23-166](#).

23.4.4 OCDS Suspend

The OCDS suspend of MultiCAN+ is controlled with the **OCS** register. MultiCAN+ module provides the following Suspend Modes:

Hard Suspend Mode (VERSION: RW ACCESS NEEDS KERNEL CLOCKING) (All actions are immediately stopped)

The Hard Suspend mode is not to be used in normal CAN applications, this mode is meant for debugging the peripheral IP within a controlled environment.

In Hard Suspend Mode, the MultiCAN+ module clocks f_{CLC} and f_{CAN} are switched off. Reading and writing of registers is possible but will enable the kernel clock for a few cycles. In this mode, there is a very high probability that the communication with other CAN devices is made impossible and that the CAN bus is blocked (e.g. if the suspended CAN module just sends a dominant level). A reset operation must be executed to leave Hard Suspend Mode.

Attention: Register accesses with clocking in Hard Suspend Mode can have unintended side effects like signals becoming and staying active. This can affect also other modules, so a MultiCAN+ kernel reset might not be sufficient to bring the system into a defined state.

Soft Suspend Mode (The current action is finished)

In Soft Suspend Mode, the MultiCAN+ module clock f_{CLC} is kept running. Module functions are stopped automatically after internal actions have been finished before it enters the suspended state with **OCS.SUSSTA** set (for example, after a CAN frame has been sent out). Due to this behavior, the communication network is not blocked. All registers are accessible for read and write actions. As a result, the debugger can stop

Controller Area Network Controller (MultiCAN+)

the module actions and modify registers. These modifications are taken into account after the Suspend Mode is left. Soft Suspend mode does not influence the kernel clock. The Hard Suspend Mode can be enabled/disabled only for the complete MultiCAN+ module. The Soft Suspend Mode can be individually enabled for each CAN node.

The user has to be aware that the Soft Suspend Mode can corrupt the TTCAN timing values, because the counter clock can be disabled. In order to guarantee correct TTCAN behavior, a TTCAN node should not enter any Suspend Mode if the consistency of the timing data must be ensured.

Note: During suspend mode, kernel registers may be accessed only when the kernel clock is running.

23.4.5 OCDS Trigger Bus (OTGB) Interface

The MultiCAN Trigger Set is shown in [Table 23-5](#). Its output is on OTGB0 or OTGB1 controlled by the **OCS** register.

Table 23-5 TS16_CAN Trigger Set MultiCAN

Bits	Name	Description
[7:0]	MON	Message Object Number for transmit and receive direction. Only valid when TT or RT is active and FMI is inactive.
[11:8]	NN	Node Number for transmit and receive direction. Only valid when TT or RT is active and FMI is inactive.
12	TT	Transmit Trigger. Active for one clock cycle when the transmit message is setup.
13	RT	Receive Trigger. Active for one clock cycle after End-of-Frame (EOF) processing for a receive message.
14	FMI	Foreign Message Indicator. The received message will not be copied to any message object. Only valid when RT is active.
15		Reserved

MCDS Trigger Setup

[Table 23-6](#) lists the qualification methods for TC16_CAN with MCDS. The methods are not exclusive and can be combined.

Controller Area Network Controller (MultiCAN+)

Table 23-6 TS16_CAN MCDS Trigger Setup

Qualification	Method
Transmit and/or receive messages	Positive edge sensitivity on TS16_CAN.TT and/or RT. Configured in the OTGM MCDS I/F.
No foreign messages.	MCDS data value condition $TS16_CAN < 4000_H$
Specific message object	MCDS masked data value comparison. $TS16_CAN \& 00FF_H ==$ message object number
Specific node	MCDS masked data value comparison. $TS16_CAN \& 0F00_H ==$ (node number) $<< 8$

Controller Area Network Controller (MultiCAN+)**23.4.6 CAN Node Control**

Each CAN node may be configured and run independently of the other CAN node. Each CAN node is equipped with its own node control logic to configure the global behavior and to provide status information.

Note: In the following descriptions, index “x” stands for the node number and index “n” represents the message object number.

Configuration Mode is activated when bit NCRx.CCE is set to 1. This mode allows CAN bit timing parameters and the error counter registers to be modified.

CAN Analyzer Mode

CAN Analyzer Mode is activated when bit NCRx.CALM is set to 1. In this operation mode, Data And Remote Frames are monitored without active participation in any CAN transfer (CAN transmit pin is held on recessive level). Incoming Remote Frames are stored in a corresponding transmit message object, while arriving data frames are saved in a matching receive message object.

In CAN Analyzer Mode, the entire configuration information of the received frame is stored in the corresponding message object, and can be evaluated by the CPU to determine their identifier, IDE bit information and data length code (ID and DLC optionally if the Remote Monitoring Mode is active, bit MOFCRn.RMM = 1). Incoming frames are not acknowledged, and no Error Frames are generated. If CAN Analyzer Mode is enabled, Remote Frames are not responded to by the corresponding Data Frame, and Data Frames cannot be transmitted by setting the transmit request bit MOSTATn.TXRQ. Receive interrupts are generated in CAN Analyzer Mode (if enabled) for all error free received frames.

The node-specific interrupt configuration is also defined by the Node Control Logic via the NCRx register bits TRIE, ALIE and LECIE:

- If control bit TRIE is set to 1, a transfer interrupt is generated when the NSRx register has been updated (after each successfully completed message transfer).
- If control bit ALIE is set to 1, an alert interrupt is generated when a “bus-off” condition has been recognized or the Error Warning Level has been exceeded or under-run. Additionally, list or object errors lead to this type of interrupt.
- If control bit LECIE is set to 1, a last error code interrupt is generated when an error code > 0 is written into bit field NSRx.LEC by hardware.

Setting bit TXDIS in register NCRx stops the transmit activity of this node without affecting reception; bit CANDIS disables the node completely.

The Node x Status Register NSRx provides an overview about the current state of the respective CAN node x, comprising information about CAN transfers, CAN node status, and error conditions.

Controller Area Network Controller (MultiCAN+)

The CAN frame counter can be used to check the transfer sequence of message objects or to obtain information about the instant a frame has been transmitted or received from the associated CAN bus. CAN frame counting is performed by a 16-bit counter, controlled by register NFCRx. Bit fields NFCRx.CFMOD and NFCRx.CFSEL determine the operation mode and the trigger event incrementing the frame counter.

23.4.6.1 Bit Timing Unit

According to the ISO 11898 standard, a CAN bit time is subdivided into different segments (**Figure 23-11**). Each segment consists of multiples of a time quantum t_q . The magnitude of t_q is adjusted by Node x Bit Timing Register bit fields NBTRx.BRP and NBTRx.DIV8, both controlling the baud rate prescaler (register NBTRx is described on **Page 23-105**). The baud rate prescaler is driven by the module timer clock f_{CAN} (generation and control of f_{CAN} is described on **Page 23-162**).

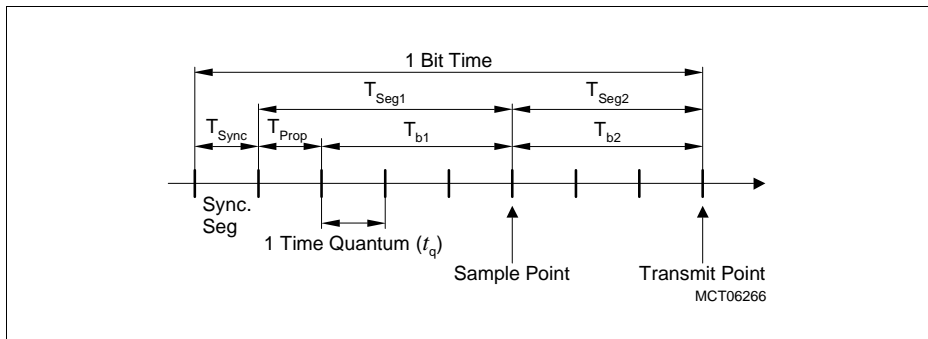


Figure 23-11 CAN Bus Bit Timing Standard

The Synchronization Segment (T_{Sync}) allows a phase synchronization between transmitter and receiver time base. The Synchronization Segment length is always one t_q . The Propagation Time Segment (T_{Prop}) takes into account the physical propagation delay in the transmitter output driver on the CAN bus line and in the transceiver circuit. For a working collision detection mechanism, T_{Prop} must be two times the sum of all propagation delay quantities rounded up to a multiple of t_q . The phase buffer segments 1 and 2 (T_{b1} , T_{b2}) before and after the signal sample point are used to compensate for a mismatch between transmitter and receiver clock phases detected in the synchronization segment.

The maximum number of time quanta allowed for re-synchronization is defined by bit field NBTRx.SJW. The Propagation Time Segment and the Phase Buffer Segment 1 are combined to parameter T_{Seg1} , which is defined by the value NBTRx.TSEG1. A minimum of 3 time quanta is demanded by the ISO standard. Parameter T_{Seg2} , which is defined by the value of NBTRx.TSEG2, covers the Phase Buffer Segment 2. A minimum of 2 time

Controller Area Network Controller (MultiCAN+)

quanta is demanded by the ISO standard. According to ISO standard, a CAN bit time, calculated as the sum of T_{Sync} , T_{Seg1} and T_{Seg2} , must not fall below 8 time quanta.

Calculation of the bit time:

$$\begin{aligned}
 t_q &= (\text{BRP} + 1) / f_{\text{CAN}} && \text{if DIV8} = 0 \\
 &= 8 \times (\text{BRP} + 1) / f_{\text{CAN}} && \text{if DIV8} = 1 \\
 T_{\text{Sync}} &= 1 \times t_q \\
 T_{\text{Seg1}} &= (\text{TSEG1} + 1) \times t_q && (\text{min. } 3 t_q) \\
 T_{\text{Seg2}} &= (\text{TSEG2} + 1) \times t_q && (\text{min. } 2 t_q) \\
 \text{bit time} &= T_{\text{Sync}} + T_{\text{Seg1}} + T_{\text{Seg2}} && (\text{min. } 8 t_q)
 \end{aligned}$$

To compensate phase shifts between clocks of different CAN controllers, the CAN controller must synchronize on any edge from the recessive to the dominant bus level. The hard synchronization is enabled (at the start of frame), the bit time is restarted at the synchronization segment. Otherwise, the re-synchronization jump width T_{SJW} defines the maximum number of time quanta, a bit time may be shortened or lengthened by one re-synchronization. The value of SJW is defined by bit field NBTRx.SJW.

$$\begin{aligned}
 T_{\text{SJW}} &= (\text{SJW} + 1) \times t_q \\
 T_{\text{Seg1}} &\geq T_{\text{SJW}} + T_{\text{prop}} \\
 T_{\text{Seg2}} &\geq T_{\text{SJW}}
 \end{aligned}$$

The maximum relative tolerance for f_{CAN} in classical CAN format and CAN FD format depends on the Phase Buffer Segments, re-synchronization jump width and the bit time.

Classical CAN Format

$$\begin{aligned}
 df_{\text{CAN}} &\leq \min(T_{b1}, T_{b2}) / [2 \times (13 \times \text{bit time} - T_{b2})] && \text{AND} \\
 df_{\text{CAN}} &\leq T_{\text{SJW}} / 20 \times \text{bit time}
 \end{aligned}$$

CAN FD Format

$$\begin{aligned}
 df_{\text{CAN}} &\leq \min(T_{b1(N)}, T_{b2(N)}) / [2 \times (13 \times \text{bit time}_{(N)} - T_{b2(N)})] && \text{AND} \\
 df_{\text{CAN}} &\leq T_{\text{SJW}(N)} / 20 \times \text{bit time}_{(N)} && \text{AND} \\
 df_{\text{CAN}} &\leq T_{\text{SJW}(D)} / 20 \times \text{bit time}_{(D)} && \text{AND}
 \end{aligned}$$

Controller Area Network Controller (MultiCAN+)

$$df_{CAN} \leq \min(T_{b1(D)}, T_{b2(D)}) / (2 \times [(6 \times \text{bit time}_{(D)} - T_{b2(D)}) \times BRP_{(D)} / BRP_{(N)} + (7 \times \text{bit time}_{(N)})]) \quad \text{AND}$$

$$df_{CAN} \leq [T_{S,JW(D)} - (BRP_{(N)} / BRP_{(D)} - 1)] / (2 \times [(2 \times \text{bit time}_{(N)} - T_{b2(N)}) \times BRP_{(N)} / BRP_{(D)} + T_{b2(D)} + 4 \times \text{bit time}_{(D)}])$$

A valid CAN bit timing must be written to the CAN Node Bit Timing Register NBTR and Fast Node Bit Timing Register before resetting the INIT bit in the Node Control Register, i.e. before enabling the operation of the CAN node.

The Node Bit Timing Register may be written only if bit CCE (Configuration Change Enable) is set in the corresponding Node Control Register.

23.4.6.2 Bitstream Processor

Based on the message objects in the message buffer, the Bitstream Processor generates the remote and Data Frames to be transmitted via the CAN bus. It controls the CRC generator and adds the checksum information to the new remote or Data Frame. After including the SOF bit and the EOF field, the Bitstream Processor starts the CAN bus arbitration procedure and continues with the frame transmission when the bus was found in idle state. While the data transmission is running, the Bitstream Processor continuously monitors the I/O line. If (outside the CAN bus arbitration phase or the acknowledge slot) a mismatch is detected between the voltage level on the I/O line and the logic state of the bit currently sent out by the transmit shift register, a CAN LEC error interrupt request is generated, and the error code is indicated by the Node x Status Register bit field NSRx.LEC.

The data consistency of an incoming frame is verified by checking the associated CRC field. When an error has been detected, a CAN LEC error interrupt request is generated and the associated error code is presented in the Node x Status Register NSRx. Furthermore, an Error Frame is generated and transmitted on the CAN bus. After decomposing a faultless frame into identifier and data portion, the received information is transferred to the message buffer executing remote and Data Frame handling, interrupt generation and status processing.

23.4.6.3 Error Handling Unit

The Error Handling Unit of a CAN node x is responsible for the fault confinement of the CAN device. Its two counters, the Receive Error Counter REC and the Transmit Error Counter TEC (bit fields of the Node x Error Counter Register NECNTx, see [Page 23-112](#)) are incremented and decremented by commands from the Bitstream Processor. If the Bitstream Processor itself detects an error while a transmit operation is running, the Transmit Error Counter is incremented by 8. An increment of 1 is used when the error condition was reported by an external CAN node via an Error Frame generation. For error analysis, the transfer direction of the disturbed message and the

Controller Area Network Controller (MultiCAN+)

node that recognizes the transfer error are indicated for the respective CAN node x in register NECNT x . Depending on the values of the error counters, the CAN node is set into error- active, error-passive, or bus-off state.

The CAN node is in error-active state if both error counters are below the error-passive limit of 128. The CAN node is in error-passive state, if at least one of the error counters is equal to or greater than 128.

The bus-off state is activated if the Transmit Error Counter is equal to or greater than the bus-off limit of 256. This state is reported for CAN node x by the Node x Status Register flag NSRx.BOFF. The device remains in this state, until the “bus-off” recovery sequence is finished. Additionally, Node x Status Register flag NSRx.EWRN is set when at least one of the error counters is equal to or greater than the error warning limit defined by the Node x Error Count Register bit field NECNT x .EWRNLVL. Bit NSRx.EWRN is reset if both error counters fall below the error warning limit again (see [Page 23-95](#)).

23.4.6.4 CAN Frame Counter

Each CAN node is equipped with a frame counter that counts transmitted/received CAN frames or obtains information about the time when a frame has been started to transmit or be received by the CAN node. CAN frame counting/bit time counting is performed by a 16-bit counter that is controlled by Node x Frame Counter Register NFCRx (see [Page 23-113](#)). Bit field NFCRx.CFSEL determines the operation mode of the frame counter:

- **Frame Count Mode:**
After the successful transmission and/or reception of a CAN frame, the frame counter is copied into the CFCVAL bit field of the MOIPR n register of the message object involved in the transfer. Afterwards, the frame counter is incremented.
- **Time Stamp Mode:**
The frame counter is incremented with the beginning of a new bit time. When the transmission/reception of a frame starts, the value of the frame counter is captured and stored to the CFC bit field of the NFCRx register. After the successful transfer of the frame the captured value is copied to the CFCVAL bit field of the MOIPR n register of the message object involved in the transfer.
- **Bit Timing Mode:**
Used for baud rate detection and analysis of the bit timing ([Chapter 23.4.8.3](#)).
- **Error Count Mode:**
The frame counter is incremented when an error frame is received or an error is detected by the node (010_B to 110_B) (see [Table 23-12](#) for [Encoding of the LEC Bit field](#)). If the NFCRx.CFCIE interrupt bit is enabled, the NFCRx.CFCOV overflow flag will be set when the frame counter overflows.

23.4.6.5 Node Timing Functions

A CAN node offers the following timing functions:

- A receive time-out mode that can detect reception of message objects; the message objects to be supervised are selectable.
- Without CPU involvement, a selectable message object can be transmitted periodically, triggered by a timer, a System Timer (STM) or General Timer Module (GTM).

The clocking options for the node timer is controlled by Node Timer Clock Control Register, **CAN_NTCCR_x (x = 0-4)**, the node timing functions for Receive Timeout Mode is controlled by Node x Timer Receive Timeout Register, **CAN_NTRTR_x (x = 0-4)** and for Transmit Trigger Mode is controlled by Node x Timer A/B/C Transmit Trigger Registers, **CAN_NTATTR_x (x = 0-4)**, **CAN NTBTR_x (x = 0-4)**, **CAN_NTCTTR_x (x = 0-4)**.

Modes with Timer Usage

A CAN node timer is driven by the CAN bit time clock, divided by a prescaler selected via bit field TPSC in the corresponding timer control register. The timers are enabled by writing to the RELOAD bits in the relevant node timer registers; then it decrements from its initial value. The further behavior depends on the selected timer mode:

- **Receive Timeout Mode:**
By setting bit MOFCR_n.RXTOE, a receive time-out check is enabled for message object n, which may be a receive or remote data object. If any of the participating message objects is received before the timer is 0, the timer will be reloaded. When the timer reaches 0, it will stop. Bit TE in NTRTR_x register will be set. With bit TEIE = 1, a node interrupt will be generated.
- **Transmit Trigger Mode:**
When the timer reaches 0, Bit MOSTAT_n.TXRQ of the message object n selected by bit field TXMO in the timer control register will be set.

Transmit Trigger by System Timer or General Timer Module

For Node x Timer the trigger for transmission of a message can also be set by a System Timer (STM) trigger event or General Timer Module (GTM) trigger event. See **Figure 23-12**.

Bit NTCCR_x.TRIGSRC in the timer clock control register enables this feature and the timer is started once values are written to the RELOAD bits of the relevant NTATT_x, NTBTT_x, or NTCTT_x registers. In transmit trigger mode, when a trigger event occurs (STM or GTM), the node timer will be decremented per trigger event timing prescaled by (TPSC+1) till it reaches zero where Bit MOSTAT_n.TXRQ of the message object n selected by bit field TXMO in the timer control register will be set.

Controller Area Network Controller (MultiCAN+)

Note: The transmit request bits (i.e NTATTx.TXMO / NTBTTx.TXMO / NTCTTx.TXMO) in the timer control register of a node is able to trigger transmit request (MOSTATn.TXRQ) of message object in a list belonging to any other CAN nodes.

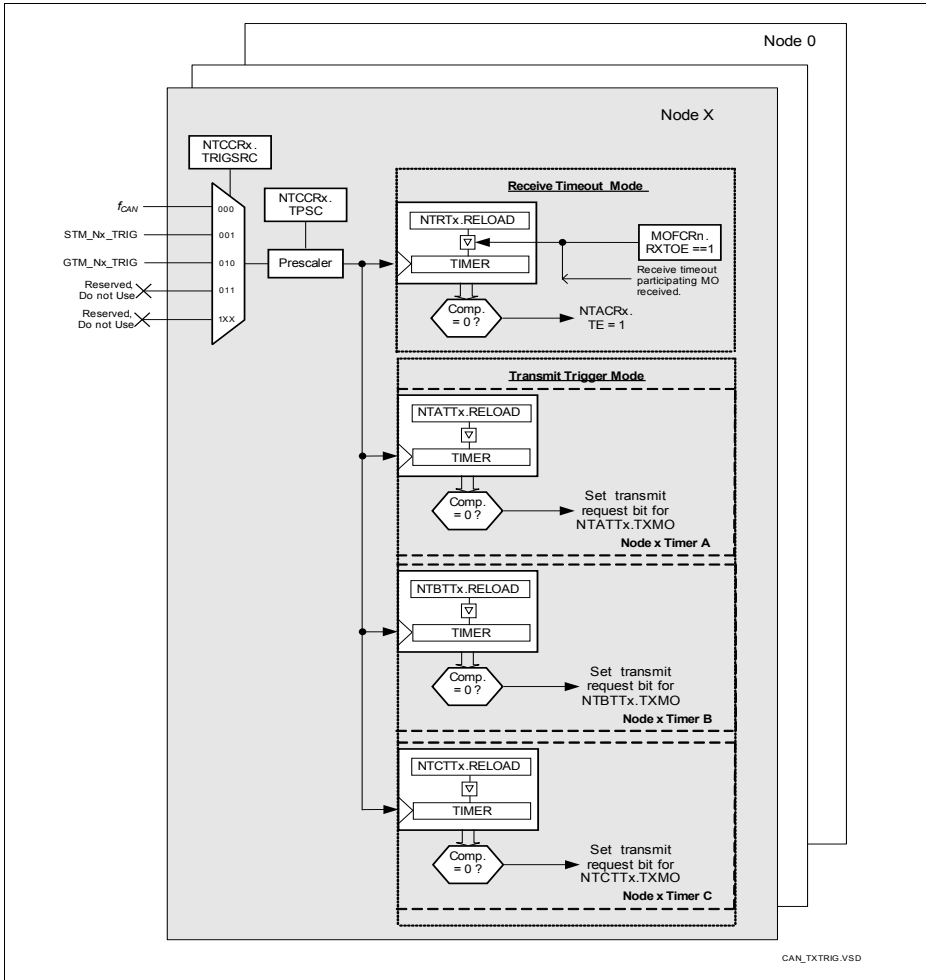


Figure 23-12 CAN Node Timing Modes

23.4.6.6 CAN Node Interrupts

Each CAN node has five hardware triggered interrupt request types that are able to generate an interrupt request upon:

- The successful transmission or reception of a frame
- A CAN protocol error with a last error code
- An alert condition: Transmit/receive error counters reach the warning limit, bus-off state changes, a List Length Error occurs, or a List Object Error occurs
- An overflow of the frame counter
- A node timer A, C or D event

Besides the hardware generated interrupts, software initiated interrupts can be generated using the Module Interrupt Trigger Register MITR. Writing a 1 to bit n of bit field MITR.IT generates an interrupt request signal on the corresponding interrupt output line INT_On. When writing MITR.IT more than one bit can be set resulting in activation of multiple INT_On interrupt output lines at the same time. See also **“Interrupt Control” on Page 23-170** for further processing of the CAN node interrupts.

Controller Area Network Controller (MultiCAN+)

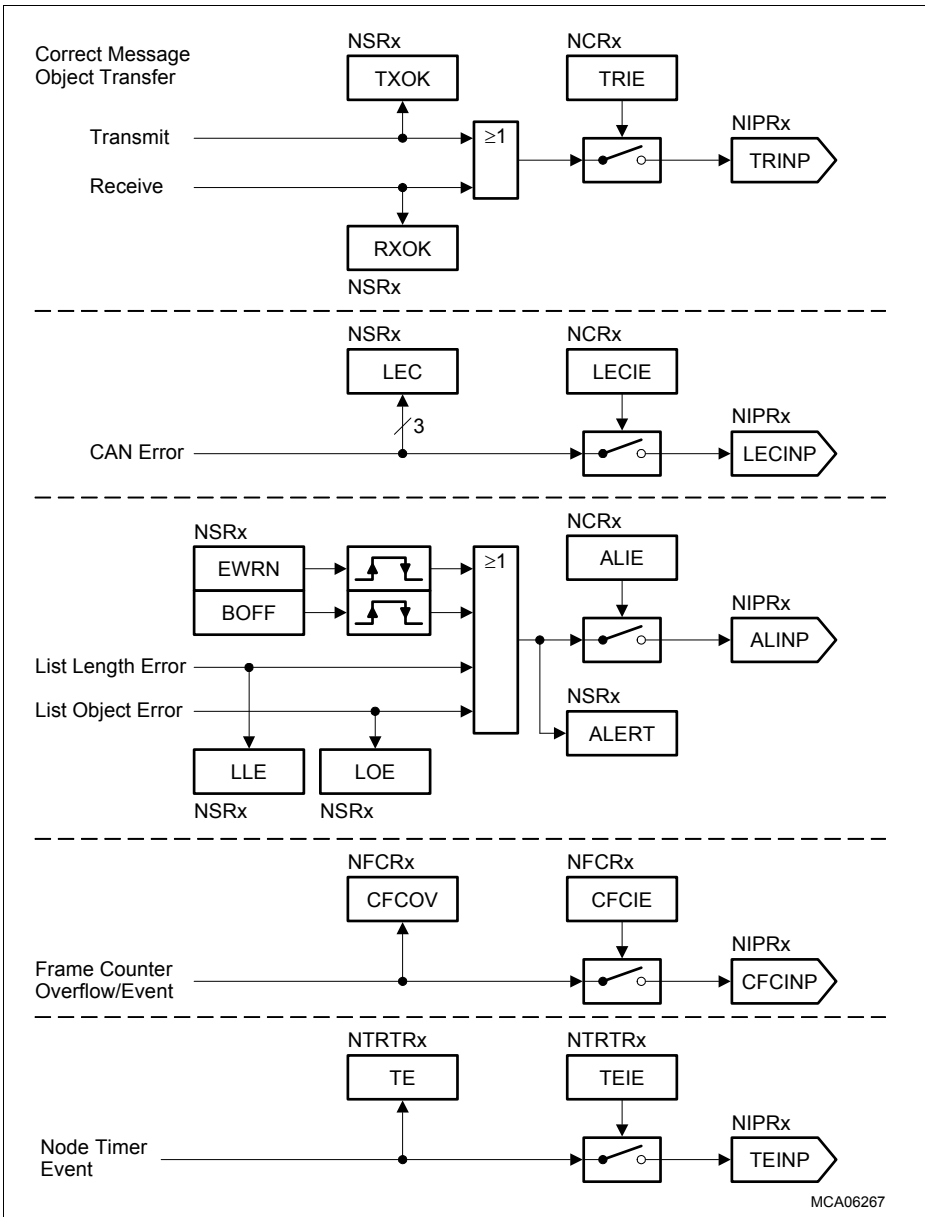


Figure 23-13 CAN Node Interrupts

Controller Area Network Controller (MultiCAN+)

23.4.7 Message Object List Structure

This section describes the structure of the message object lists in the MultiCAN+ module.

23.4.7.1 Basics

The message objects of the MultiCAN+ module are organized in double-chained lists, where each message object has a pointer to the previous message object in the list as well as a pointer to the next message object in the list. The MultiCAN+ module provides 16 lists. Each message object is allocated to one of these lists. In the example in [Figure 23-14](#), the three message objects (3, 5, and 16) are allocated to the list with index 2 (List Register LIST2).

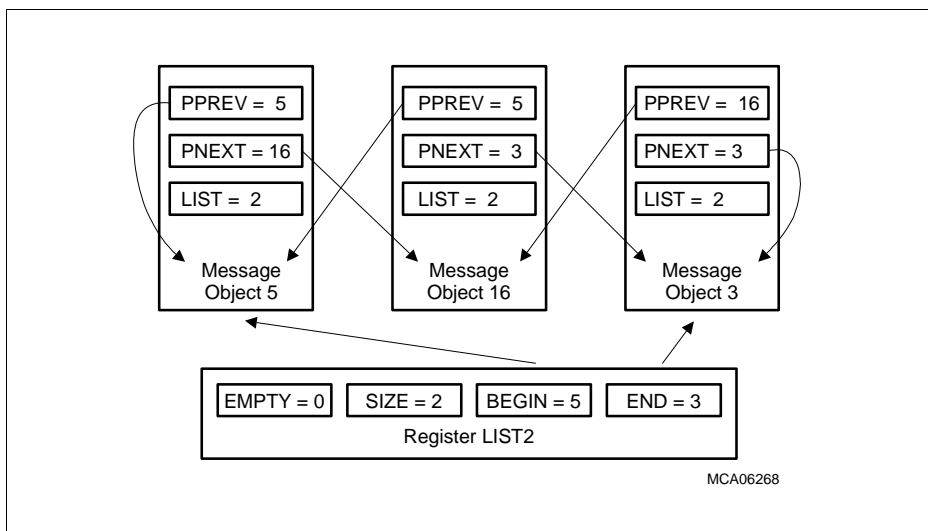


Figure 23-14 Example Allocation of Message Objects to a List

Bit field BEGIN in the List Register (for definition, see [Page 23-86](#)) points to the first element in the list (object 5 in the example), and bit field END points to the last element in the list (object 3 in the example). The number of elements in the list is indicated by bit field SIZE of the List Register (SIZE = number of list elements - 1, thus SIZE = 2 for the 3 elements in the example). The EMPTY bit of the List Register indicates whether or not a list is empty (EMPTY = 0 in the example, because list 2 is not empty).

Each message object n has a pointer PNEXT in its Message Object n Status Register MOSTAT n (see [Page 23-126](#)) that points to the next message object in the list, and a pointer PPREV that points to the previous message object in the list. PPREV of the first message object points to the message object itself because the first message object has

Controller Area Network Controller (MultiCAN+)

no predecessor (in the example message object 5 is the first message object in the list, indicated by PPREV = 5). PNEXT of the last message object also points to the message object itself because the last message object has no successor (in the example, object 3 is the last message object in the list, indicated by PNEXT = 3).

Bit field MOSTATn.LIST indicates the list index number to which the message object is currently allocated. The message object of the example are allocated to list 2. Therefore, all LIST bit fields for the message objects assigned to list 2 are set to LIST = 2.

23.4.7.2 List of Unallocated Elements

The list with list index 0 has a special meaning: it is the list of all unallocated elements. An element is called unallocated if it belongs to list 0 (MOSTATn.LIST = 0). It is called allocated if it belongs to a list with an index not equal to 0 (MOSTATn.LIST > 0).

After reset, all message objects are unallocated. This means that they are assigned to the list of unallocated elements with MOSTATn.LIST = 0. After this initial allocation of the message objects caused by reset, the list of all unallocated message objects is ordered by message number (predecessor of message object n is object n-1, successor of object n is object n+1).

23.4.7.3 Connection to the CAN Nodes

Each CAN node is linked to one unique list of message objects. A CAN node performs message transfer only with the message objects that are allocated to the list of the CAN node. This is illustrated in [Figure 23-15](#). Frames that are received on a CAN node may only be stored in one of the message objects that belongs to the CAN node; frames to be transmitted on a CAN node are selected only from the message objects that are allocated to that node, as indicated by the vertical arrows.

There are more lists (16) than CAN nodes (5). This means that some lists are not linked to one of the CAN nodes. A message object that is allocated to one of these unlinked lists cannot receive messages directly from a CAN node and it may not transmit messages.

FIFO and gateway mechanisms refer to message numbers and not directly to a specific list. The user must take care that the message objects targeted by FIFO/gateway belong to the desired list. The mechanisms make it possible to work with lists that do not belong to the CAN node.

Controller Area Network Controller (MultiCAN+)

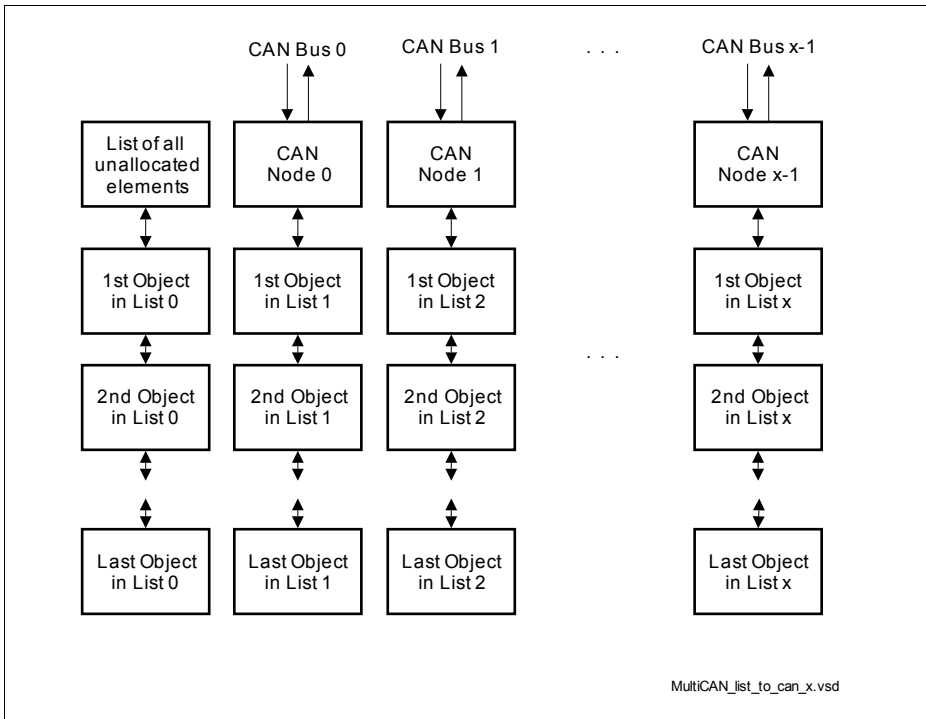


Figure 23-15 Message Objects Linked to CAN Nodes

23.4.7.4 List Command Panel

The list structure cannot be modified directly by write accesses to the LIST registers and the PPREV, PNEXT and LIST bit fields in the Message Object Status Registers, as they are read only. The list structure is managed by and limited to the list controller inside the MultiCAN+ module. The list controller is controlled via a command panel allowing the user to issue list allocation commands to the list controller. The list controller has two main purposes:

1. Ensure that all operations that modify the list structure result in a consistent list structure.
2. Present maximum ease of use and flexibility to the user.

The list controller and the associated command panel allows the programmer to concentrate on the final properties of the list, which are characterized by the allocation of message objects to a CAN node, and the ordering relation between objects that are allocated to the same list. The process of list (re-)building is done in the list controller.

Controller Area Network Controller (MultiCAN+)

Table 23-7 gives an overview on the available panel commands while **Table 23-11** on **Page 23-80** describes the panel commands in more detail.

Table 23-7 Panel Commands Overview

Command Name	Description
No Operation	No new command is started.
Initialize Lists	Run the initialization sequence to reset the CTRL and LIST field of all message objects.
Static Allocate	Allocate message object to a list.
Dynamic Allocate	Allocate the first message object of the list of unallocated objects to the selected list.
Static Insert Before	Remove a message object (source object) from the list that it currently belongs to, and insert it before a given destination object into the list structure of the destination object.
Dynamic Insert Before	Insert a new message object before a given destination object.
Static Insert Behind	Remove a message object (source object) from the list that it currently belongs to, and insert it behind a given destination object into the list structure of the destination object.
Dynamic Insert Behind	Insert a new message object behind a given destination object.

A panel command is started by writing the respective command code into the Panel Control Register bit field PANCTR.PANCMD (see **Page 23-79**). The corresponding command arguments must be written into bit fields PANCTR.PANAR1 and PANCTR.PANAR2 before writing the command code, or latest along with the command code in a single 32-bit write access to the Panel Control Register.

With the write operation of a valid command code, the PANCTR.BUSY flag is set and further write accesses to the Panel Control Register are ignored. The BUSY flag remains active and the control panel remains locked until the execution of the requested command has been completed. After a reset and resetting the CLC.DISR (see **Page 23-164**) register, the list controller builds up list 0. Afterwards the BUSY bit is set, dependent on core speed this might be visible. During list controller initialization, BUSY is set and other accesses to the CAN RAM are forbidden. The CAN RAM can be accessed again when BUSY becomes inactive.

Note: The CAN RAM is automatically initialized after reset by the list controller in order to ensure correct list pointers in each message object. The end of this CAN RAM initialization is indicated by bit PANCTR.BUSY becoming inactive.

Controller Area Network Controller (MultiCAN+)

In case of a dynamic allocation command that takes an element from the list of unallocated objects, the PANCTR.RBUSY bit is also set along with the BUSY bit (RBUSY = BUSY = 1). This indicates that bit fields PANAR1 and PANAR2 are going to be updated by the list controller in the following way:

1. The message number of the message object taken from the list of unallocated elements is written to PANAR1.
2. If ERR (bit 7 of PANAR2) is set to 1, the list of unallocated elements was empty and the command is aborted. If ERR is 0, the list was not empty and the command will be performed successfully.

The results of a dynamic allocation command are written before the list controller starts the actual allocation process. As soon as the results are available, RBUSY becomes inactive (RBUSY = 0) again, while BUSY still remains active until completion of the command. This allows the user to set up the new message object while it is still in the process of list allocation. The access to message objects is not limited during ongoing list operations. However, any access to a register resource located inside the RAM delays the ongoing allocation process by one access cycle.

As soon as the command is finished, the BUSY flag becomes inactive (BUSY = 0) and write accesses to the Panel Control Register are enabled again. Also, the “No Operation” command code is automatically written to the PANCTR.PANCMD field. A new command may be started any time when BUSY = 0.

All fields of the Panel Control Register PANCTR except BUSY and RBUSY may be written by the user. This makes it possible to save and restore the Panel Control Register if the Command Panel is used within independent (mutually interruptible) interrupt service routines. If this is the case, any task that uses the Command Panel and that may interrupt another task that also uses the Command Panel should poll the BUSY flag until it becomes inactive and save the whole PANCTR register to a memory location before issuing a command. At the end of the interrupt service routine, the task should restore PANCTR from the memory location.

Before a message object that is allocated to the list of an active CAN node shall be moved to another list or to another position within the same list, bit MOSTATn.MSGVAL (“Message Valid”) of message object n must be cleared.

23.4.8 CAN Node Analyzer Mode

The chapter describes the CAN node analyzer capabilities of the MultiCAN+ module.

23.4.8.1 Analyzer Mode

The CAN Analyzer Mode makes it possible to monitor the CAN traffic for each CAN node individually without affecting the logical state of the CAN bus. The CAN Analyzer Mode for CAN node x is selected by setting Node x Control Register bit NCRx.CALM.

Controller Area Network Controller (MultiCAN+)

In CAN Analyzer Mode, the transmit pin of a CAN node is held at a recessive level permanently. The CAN node may receive frames (Data, Remote, and Error Frames) but is not allowed to transmit. Received Data/Remote Frames are not acknowledged (i.e. acknowledge slot is sent recessive) but will be received and stored in matching message objects as long as there is any other node that acknowledges the frame. The complete message object functionality is available, but no transmit request will be executed. CAN Analyzer mode works for both Classical CAN format and CAN FD format.

23.4.8.2 Loop-Back Mode

The MultiCAN+ module provides a Loop-Back Mode to enable an in-system test of the MultiCAN+ module as well as the development of CAN driver software without access to an external CAN bus.

The loop-back feature consists of an internal CAN bus (inside the MultiCAN+ module) and a bus select switch for each CAN node (see [Figure 23-16](#)). With the switch, each CAN node can be connected either to the internal CAN bus (Loop-Back Mode activated) or the external CAN bus, respectively to transmit and receive pins (normal operation). The CAN bus that is not currently selected is driven recessive; this means the transmit pin is held at 1, and the receive pin is ignored by the CAN nodes that are in Loop-Back Mode.

The Loop-Back Mode is selected for CAN node *x* by setting the Node *x* Port Control Register bit NPCRx.LBM. All CAN nodes that are in Loop-Back Mode may communicate together via the internal CAN bus without affecting the normal operation of the other CAN nodes that are not in Loop-Back Mode.

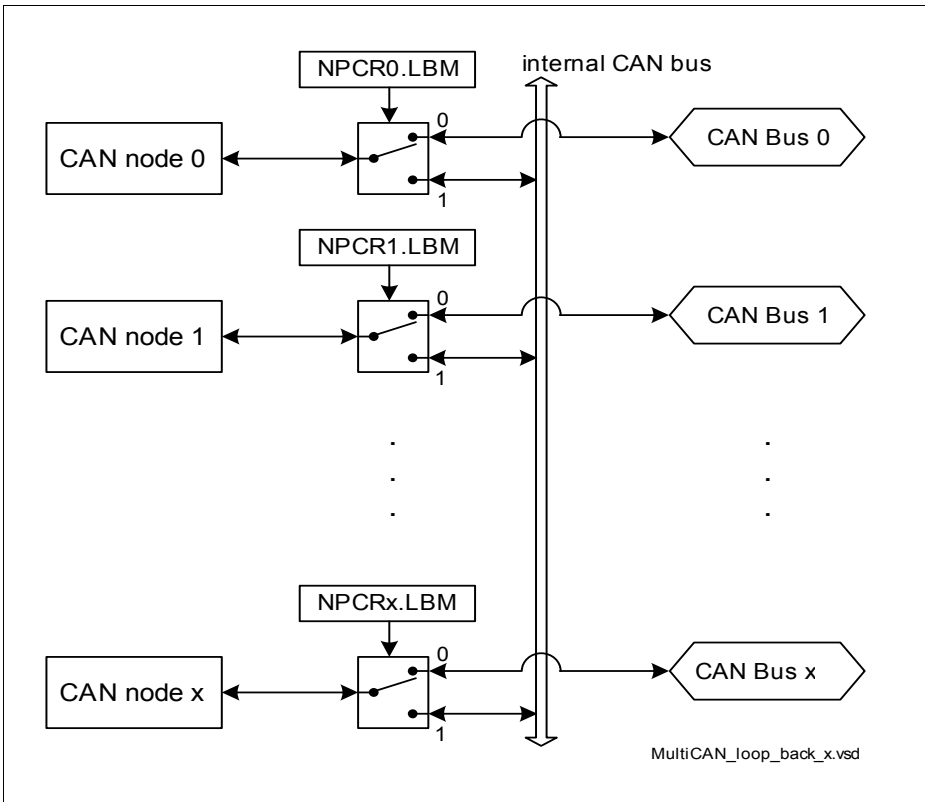


Figure 23-16 Loop-Back Mode

23.4.8.3 Bit Timing Analysis

Detailed analysis of the bit timing can be performed for each CAN node using the analysis modes of the CAN frame counter. The bit timing analysis functionality of the frame counter may be used for automatic detection of the CAN baud rate, as well as to analyze the timing of the CAN network.

Bit timing analysis for CAN node x is selected when bit field $NFCRx.CFMODE = 10_B$. Bit timing analysis does not affect the operation of the CAN node.

The bit timing measurement results are written into the $NFCRx.CFC$ bit field. Whenever $NFCRx.CFC$ is updated in bit timing analysis mode, bit $NFCRx.CFCOV$ is also set to indicate the CFC update event. The value of $NFCRx.CFC$ is valid one module cycle later when $NFCRx.CFCOV$ is set. If $NFCRx.CFCIE$ is set, an interrupt request can be generated (see [Figure 23-13](#)).

Controller Area Network Controller (MultiCAN+)

Automatic Baud Rate Detection

For automatic baud rate detection, the time between the observation of subsequent dominant edges on the CAN bus must be measured. This measurement is automatically performed if bit field `NFCRx.CFSEL = 000B`. With each dominant edge monitored on the CAN receive input line, the time (measured in f_{CAN} clock cycles) between this edge and the most recent dominant edge is stored in the `NFCRx.CFC` bit field.

Synchronization Analysis

The bit time synchronization is monitored if `NFCRx.CFSEL = 010B`. The time between the first dominant edge and the sample point is measured and stored in the `NFCRx.CFC` bit field. The bit timing synchronization offset may be derived from this time as the first edge after the sample point triggers synchronization and there is only one synchronization between consecutive sample points.

Synchronization analysis can be used, for example, for fine tuning of the baud rate during reception of the first CAN frame with the measured baud rate.

Driver Delay Measurement

The delay between a transmitted edge and the corresponding received edge is measured when `NFCRx.CFSEL = 011B` (dominant to dominant) and `NFCRx.CFSEL = 100B` (recessive to recessive). These delays indicate the time needed to represent a new bit value on the physical implementation of the CAN bus.

23.4.9 Message Acceptance Filtering

The chapter describes the Message Acceptance Filtering capabilities of the MultiCAN+ module.

23.4.9.1 Receive Acceptance Filtering

When a CAN frame is received by a CAN node, a unique message object is determined in which the received frame is stored after successful frame reception. A message object is qualified for reception of a frame if the following six conditions are met.

- The message object is allocated to the message object list of the CAN node by which the frame is received.
- Bit `MOSTATn.MSGVAL` in the Message Object Status Register (see [Page 23-126](#)) is set.
- Bit `MOSTATn.RXEN` is set.
- Bit `MOSTATn.DIR` is equal to bit `RTR` of the received frame.
If bit `MOSTATn.DIR = 1` (transmit object), the message object accepts only Remote Frames. If bit `MOSTATn.DIR = 0` (receive object), the message object accepts only Data Frames.

Controller Area Network Controller (MultiCAN+)

- If bit MOAMRn.MIDE = 1, the IDE bit of the received frame becomes evaluated in the following way: If MOARn.IDE = 1, the IDE bit of the received frame must be set (indicates extended identifier). If MOARn.IDE = 0, the IDE bit of the received frame must be cleared (indicates standard identifier).
If bit MOAMRn.MIDE = 0, the IDE bit of the received frame is “don’t care”. In this case, message objects with standard and extended frames are accepted.
- The identifier of the received frame matches the identifier stored in the Arbitration Register of the message object as qualified by the acceptance mask in the MOAMRn register. This means that each bit of the received message object identifier is equal to the bit field MOARn.ID, except those bits for which the corresponding acceptance mask bits in bit field MOAMRn.AM are cleared. These identifier bits are “don’t care” for reception. **Figure 23-17** illustrates this receive message identifier check.

Among all messages that fulfill all six qualifying criteria the message object with the highest receive priority wins receive acceptance filtering and becomes selected to store the received frame. All other message objects lose receive acceptance filtering.

The following priority scheme is defined for the message objects:

A message object a (MOa) has higher receive priority than a message object b (MOb) if the following two conditions are fulfilled (see **Page 23-143**):

1. MOa has a higher priority class than MOb. This means, the 2-bit priority bit field MOARa.PRI must be equal or less than bit field MOARb.PRI.
2. If both message objects have the same priority class (MOARa.PRI = MOARb.PRI), MOb is a list successor of MOa. This means that MOb can be reached by means of successively stepping forward in the list, starting from a.

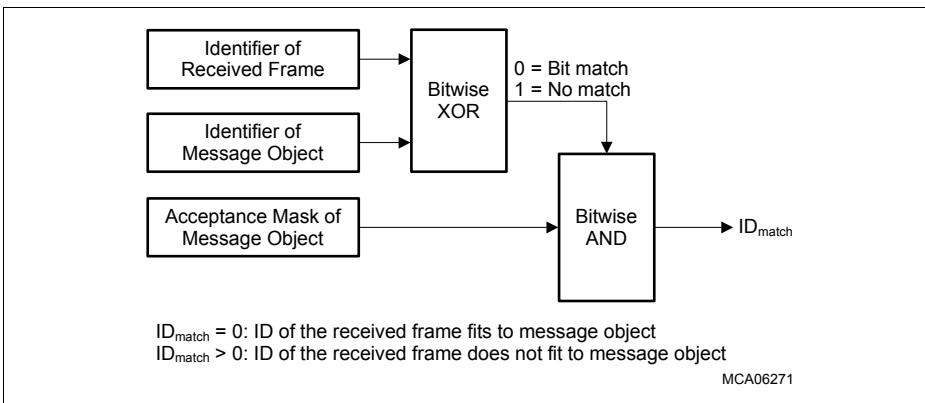


Figure 23-17 Received Message Identifier Acceptance Check

23.4.9.2 Transmit Acceptance Filtering

A message is requested for transmission by setting a transmit request in the message object that holds the message. If more than one message object have a valid transmit request for the same CAN node, one of these message objects is chosen for transmission, because only a single message object can be transmitted at one time on a CAN bus.

A message object is qualified for transmission on a CAN node if the following four conditions are met (see also [Figure 23-18](#)).

1. The message object is allocated to the message object list of the CAN node.
2. Bit MOSTATn.MSGVAL is set.
3. Bit MOSTATn.TXRQ is set.
4. Bit MOSTATn.TXEN0 and MOSTATn.TXEN1 are set.

A priority scheme determines which one of all qualifying message objects is transmitted first. It is assumed that message object a (MOa) and message object b (MOb) are two message objects qualified for transmission. MOb is a list successor of MOa. For both message objects, CAN messages CANa and CANb are defined (identifier, IDE, and RTR are taken from the message-specific bit fields and bits MOARn.ID, MOARn.IDE and MOSTATn.DIR).

If both message objects belong to the same priority class (identical PRI bit field in register MOARn), MOa has a higher transmit priority than MOb if one of the following conditions is fulfilled.

- $PRI = 10_B$ and CAN message MOa has higher or equal priority than CAN message MOb with respect to CAN arbitration rules (see [Table 23-19](#) on [Page 23-144](#)).
- $PRI = 01_B$ or $PRI = 11_B$ (priority by list order).

The message object that is qualified for transmission and has highest transmit priority wins the transmit acceptance filtering, and will be transmitted first. All other message objects lose the current transmit acceptance filtering round. They get a new chance in subsequent acceptance filtering rounds.

Controller Area Network Controller (MultiCAN+)

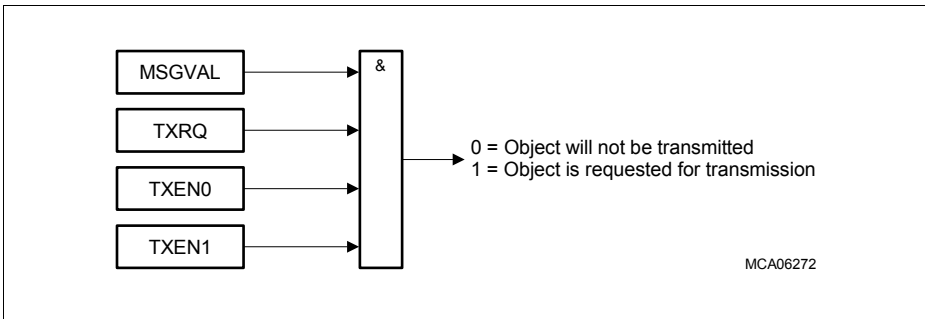


Figure 23-18 Effective Transmit Request of Message Object

23.4.10 Message Postprocessing

After a message object has successfully received or transmitted a frame, the CPU can be notified to perform a postprocessing on the message object. The postprocessing of the MultiCAN+ module consists of two elements:

1. Message interrupts to trigger postprocessing.
2. Message pending registers to collect pending message interrupts into a common structure for postprocessing.

23.4.10.1 Message Object Interrupts

When the storage of a received frame into a message object or the successful transmission of a frame is completed, a message interrupt can be issued. For each message object, a transmit and a receive interrupt can be generated and routed to one of the sixteen CAN interrupt output lines (see [Figure 23-19](#)). A receive interrupt occurs also after a frame storage event that has been induced by a FIFO or a gateway action. The status bits TXPND and RXPND in the Message Object n Status Register are always set after a successful transmission/reception, whether or not the respective message interrupt is enabled.

A third FIFO full interrupt condition of a message object is provided. If bit field MOFCRn.OVIE (Overflow Interrupt Enable) is set, the FIFO full interrupt will be activated depending on the actual message object type.

In case of a Receive FIFO Base Object (MOFCRn.MMC = 0001_B), the FIFO full interrupt is routed to the interrupt output line INT_Om as defined by the transmit interrupt node pointer MOIPRn.TXINP.

In case of a Transmit FIFO Base Object (MOFCRn.MMC = 0010_B), the FIFO full interrupt becomes routed to the interrupt output line INT_Om as defined by the receive interrupt node pointer MOIPRn.RXINP.

Controller Area Network Controller (MultiCAN+)

See also **“Interrupt Control” on Page 23-170** for further processing of the message object interrupts.

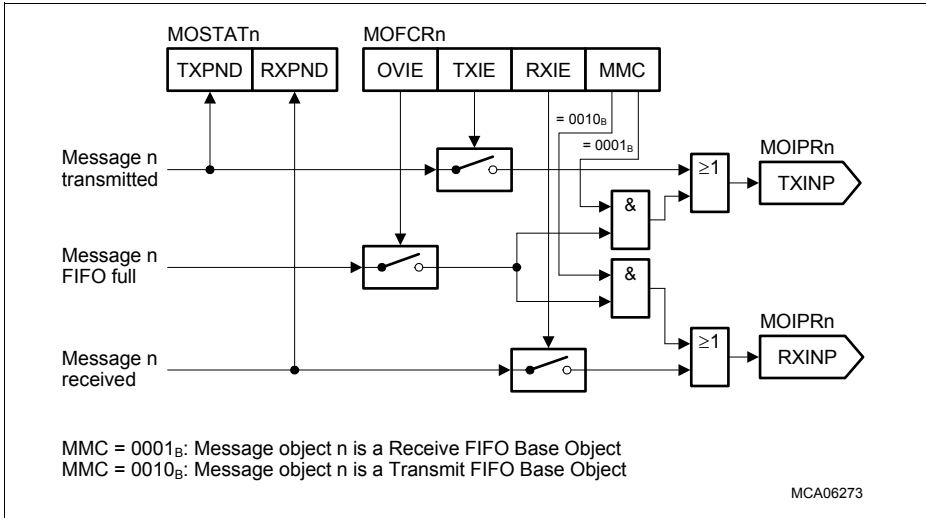


Figure 23-19 Message Interrupt Request Routing

Controller Area Network Controller (MultiCAN+)

23.4.10.2 Pending Messages

When a message interrupt request is generated, a message pending bit is set in one of the Message Pending Registers. There are 8 Message Pending Registers, MSPNDk (k = 0-7) with 32 pending bits available each. The general Figure 23-20 shows the allocation of the message pending bits in case that the maximum possible number of eight Message Pending Registers are implemented and available on the chip.

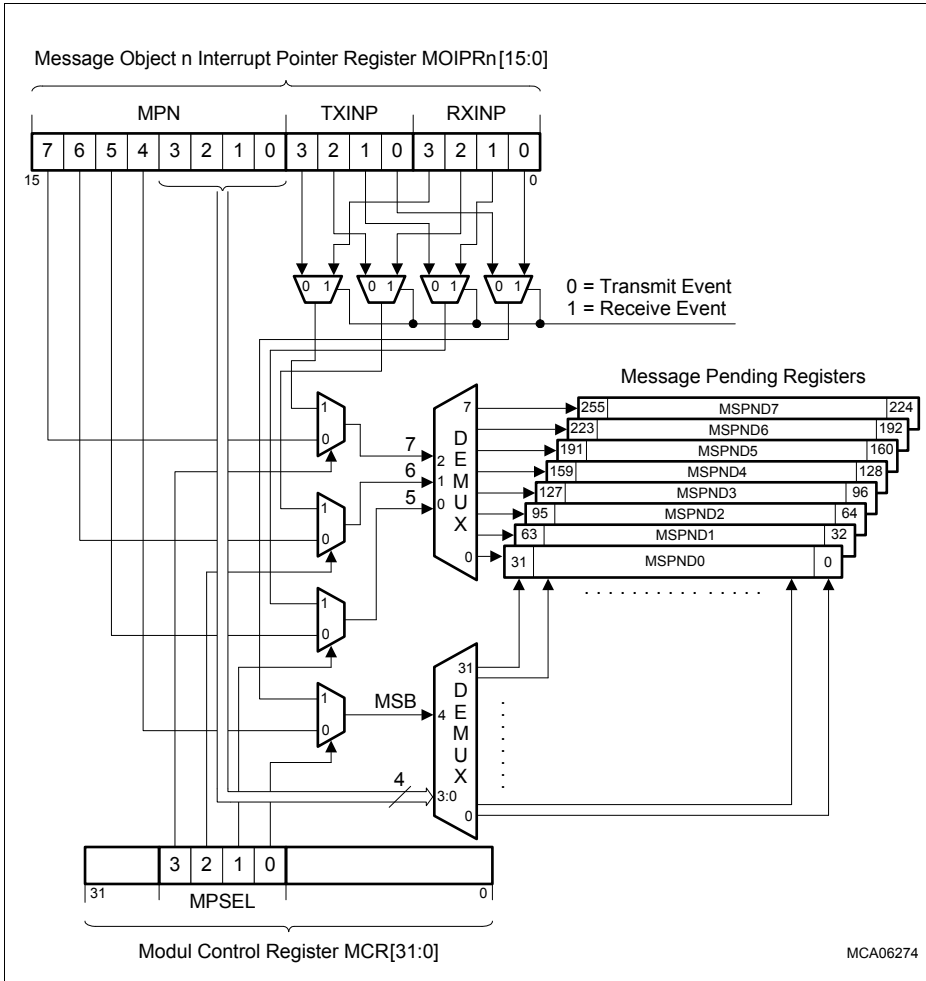


Figure 23-20 Message Pending Bit Allocation

Controller Area Network Controller (MultiCAN+)

The location of a pending bit is defined by two demultiplexers selecting the number k of the MSPNDk registers (3-bit demux), and the bit location within the corresponding MSPNDk register (5-bit demux).

Allocation Case 1

In this allocation case, bit field MCR.MPSEL = 0000_B (see [Page 23-83](#)). The location selection consists of 2 parts:

- The upper three bits of MOIPRn.MPN (MPN[7:5]) select the number k of a Message Pending Register MSPNDk in which the pending bit will be set.
- The lower five bits of MOIPRn.MPN (MPN[4:0]) select the bit position (0-31) in MSPNDk for the pending bit to be set.

Allocation Case 2

In this allocation case, bit field MCR.MPSEL is taken into account for pending bit allocation. Bit field MCR.MPSEL makes it possible to include the interrupt request node pointer for reception (MOIPRn.RXINP) or transmission (MOIPRn.TXINP) for pending bit allocation in such a way that different target locations for the pending bits are used in receive and transmit case. If MPSEL = 1111_B, the location selection operates in the following way:

- At a transmit event, the upper 3 bits of TXINP determine the number k of a Message Pending Register MSPNDk in which the pending bit will be set. At a receive event, the upper 3 bits of RXINP determine the number k .
- The bit position (0-31) in MSPNDk for the pending bit to be set is selected by the lowest bit of TXINP or RXINP (selects between low and high half-word of MSPNDk) and the four least significant bits of MPN.

General Hints

The Message Pending Registers MSPNDk can be written by software. Bits that are written with 1 are left unchanged, and bits which are written with 0 are cleared. This makes it possible to clear individual MSPNDk bits with a single register write access. Therefore, access conflicts are avoided when the MultiCAN+ module (hardware) sets another pending bit at the same time when software writes to the register.

Each Message Pending Register MSPNDk is associated with a Message Index Register MSIDk (see [Page 23-89](#)) which indicates the lowest bit position of all set (1) bits in Message Pending Register k . The MSIDk register is a read-only register that is updated immediately when a value in the corresponding Message Pending Register k is changed by software or hardware.

23.4.11 Message Object Data Handling

This chapter describes the handling capabilities for the Message Object Data of the MultiCAN+ module.

23.4.11.1 Frame Reception

After the reception of a message, it is stored in a message object according to the scheme shown in [Figure 23-21](#). The MultiCAN+ module not only copies the received data into the message object, and it provides advanced features to enable consistent data exchange between MultiCAN+ and CPU.

MSGVAL

Bit MSGVAL (Message Valid) in the Message Object n Status Register MOSTATn is the main switch of the message object. During the frame reception, information is stored in the message object only when MSGVAL = 1. If bit MSGVAL is reset by the CPU, the MultiCAN+ module stops all ongoing write accesses to the message object. Now the message object can be re-configured by the CPU with subsequent write accesses to it without being disturbed by the MultiCAN+.

RTSEL

When the CPU re-configures a message object during CAN operation (for example, clears MSGVAL, modifies the message object and sets MSGVAL again), the following scenario can occur:

1. The message object wins receive acceptance filtering.
2. The CPU clears MSGVAL to re-configure the message object.
3. The CPU sets MSGVAL again after re-configuration.
4. The end of the received frame is reached. As MSGVAL is set, the received data is stored in the message object, a message interrupt request is generated, gateway and FIFO actions are processed, etc.

After the re-configuration of the message object (after step 3 above) the storage of further received data may be undesirable. This can be achieved through bit MOSTATn.RTSEL (Receive/Transmit Selected) that makes it possible to disconnect a message object from an ongoing frame reception.

When a message object wins the receive acceptance filtering, its RTSEL bit is set by the MultiCAN+ module to indicate an upcoming frame delivery. The MultiCAN+ module checks RTSEL whether it is set on successful frame reception to verify that the object is still ready for receiving the frame. The received frame is then stored in the message object (along with all subsequent actions such as message interrupts, FIFO & gateway actions, flag updates) only if RTSEL = 1.

When a message object is invalidated during CAN operation (resetting bit MSGVAL), RTSEL should be cleared before setting MSGVAL again (latest with the same write

Controller Area Network Controller (MultiCAN+)

access that sets MSGVAL) to prevent the storage of a frame that belongs to the old context of the message object. Therefore, a message object re-configuration should consist of the following steps:

1. Clear MSGVAL bit
2. Re-configure the message object while MSGVAL = 0
3. Clear RTSEL bit and set MSGVAL again

RXEN

Bit MOSTATn.RXEN enables a message object for frame reception. A message object can receive CAN messages from the CAN bus only if RXEN = 1. The MultiCAN+ module evaluates RXEN only during receive acceptance filtering. After receive acceptance filtering, RXEN is ignored and has no further influence on the actual storage of a received message in a message object.

Bit RXEN enables the “soft phase out” of a message object: after clearing RXEN, a currently received CAN message for which the message object has won acceptance filtering is still stored in the message object but for subsequent messages the message object no longer wins receive acceptance filtering.

RXUPD, NEWDAT and MSGLST

An ongoing frame storage process is indicated by the RXUPD (Receive Updating) flag in the MOSTATn register. RXUPD is set with the start and cleared with the end of a message object update, which consists of frame storage as well as flag updates.

After storing the received frame (identifier, IDE bit, DLC; including the Data Field for Data Frames), the NEWDAT (New Data) bit of the message object is set. If NEWDAT was already set before it becomes set again, bit MSGLST (Message Lost) is set to indicate a data loss condition.

The RXUPD and NEWDAT flags can help to read consistent frame data from the message object during an ongoing CAN operation. The following steps are recommended to be executed:

1. Clear NEWDAT bit.
2. Read message content (identifier, data etc.) from the message object.
3. Check that both, NEWDAT and RXUPD, are cleared. If this is not the case, go back to step 1.
4. When step 3 was successful, the message object contents are consistent and has not been updated by the MultiCAN+ module while reading.

Bits RXUPD, NEWDAT and MSGLST have the same behavior for the reception of Data as well as Remote Frames.

Controller Area Network Controller (MultiCAN+)

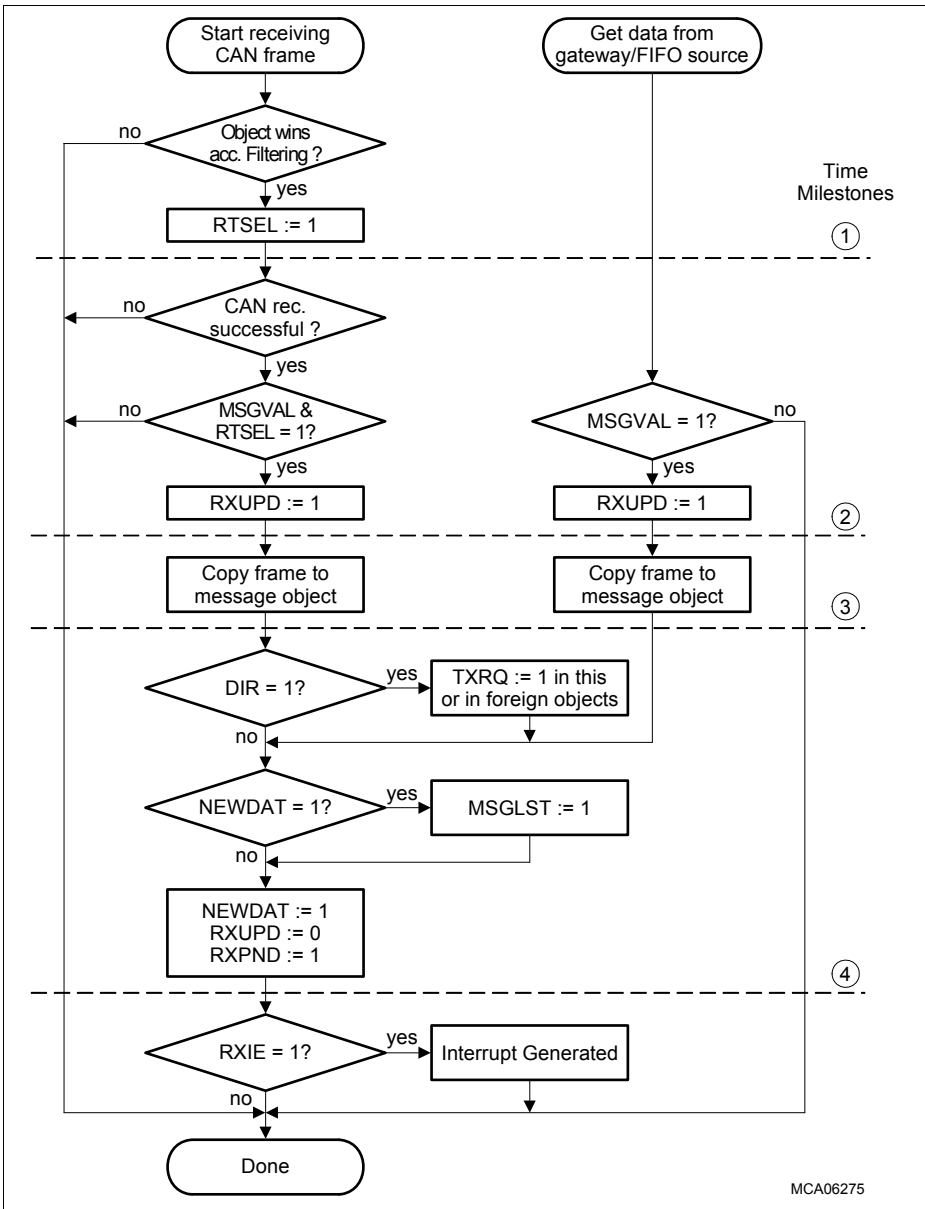


Figure 23-21 Reception of a Message Object

Controller Area Network Controller (MultiCAN+)
23.4.11.2 Frame Transmission

The process of a message object transmission is shown in [Figure 23-22](#). Along with the copy of the message object content to be transmitted (identifier, IDE bit, RTR = DIR bit, DLC, including the Data Field for Data Frames) into the internal transmit buffer of the assigned CAN node, several status flags are also served and monitored to control consistent data handling.

The transmission process of a message object starting after the transmit acceptance filtering is identical for Remote and Data Frames.

MSGVAL, TXRQ, TXEN0, TXEN1

A message can only be transmitted if all four bits in registers MOSTATn, MSGVAL (Message Valid), TXRQ (Transmit Request), TXEN0 (Transmit Enable 0), TXEN1 (Transmit Enable 1) are set as shown in [Figure 23-18](#). Although these bits are equivalent with respect to the transmission process, they have different semantics:

Table 23-8 Message Transmission Bit Definitions

Bit	Description
MSGVAL	Message Valid This is the main switch bit of the message object.
TXRQ	Transmit Request This is the standard transmit request bit. This bit must be set whenever a message object should be transmitted. TXRQ is cleared by hardware at the end of a successful transmission, except when there is new data (indicated by NEWDAT = 1) to be transmitted. When bit MOFCRn.STT (“Single Transmit Trial”) is set, TXRQ becomes already cleared when the contents of the message object are copied into the transmit frame buffer of the CAN node. A received remote request (after a Remote Frame reception) sets bit TXRQ to request the transmission of the requested data frame.
TXEN0	Transmit Enable 0 This bit can be temporarily cleared by software to suppress the transmission of this message object when it writes new content to the Data Field. This avoids transmission of inconsistent frames that consist of a mixture of old and new data. Remote requests are still accepted when TXEN0 = 0, but transmission of the Data Frame is suspended until transmission is re-enabled by software (setting TXEN0).

Controller Area Network Controller (MultiCAN+)
Table 23-8 Message Transmission Bit Definitions (cont'd)

Bit	Description
TXEN1	<p>Transmit Enable 1</p> <p>This bit is used in transmit FIFOs to select the message object that is transmit active within the FIFO structure.</p> <p>For message objects that are not transmit FIFO elements, TXEN1 can either be set permanently to 1 or can be used as a second independent transmission enable bit.</p>

RTSEL

When a message object has been identified to be transmitted next after transmission acceptance filtering, bit MOSTATn.RTSEL (Receive/Transmit Selected) is set.

When the message object is copied into the internal transmit buffer, bit RTSEL is checked, and the message is transmitted only if RTSEL = 1. After the successful transmission of the message, bit RTSEL is checked again and the message postprocessing is only executed if RTSEL = 1.

For a complete re-configuration of a valid message object, the following steps should be executed:

1. Clear MSGVAL bit
2. Re-configure the message object while MSGVAL = 0
3. Clear RTSEL and set MSGVAL

Clearing of RTSEL ensures that the message object is disconnected from an ongoing/scheduled transmission and no message object processing (copying message to transmit buffer including clearing NEWDAT, clearing TXRQ, time stamp update, message interrupt, etc.) within the old context of the object can occur after the message object becomes valid again, but within a new context.

NEWDAT

When the contents of a message object have been transferred to the internal transmit buffer of the CAN node, bit MOSTATn.NEWDAT (New Data) is cleared by hardware to indicate that the transmit message object data is no longer new.

When the transmission of the frame is successful and NEWDAT is still cleared (if no new data has been copied into the message object meanwhile), TXRQ (Transmit Request) is cleared automatically by hardware.

If, however, the NEWDAT bit has been set again by the software (because a new frame should be transmitted), TXRQ is not cleared to enable the transmission of the new data.

Controller Area Network Controller (MultiCAN+)

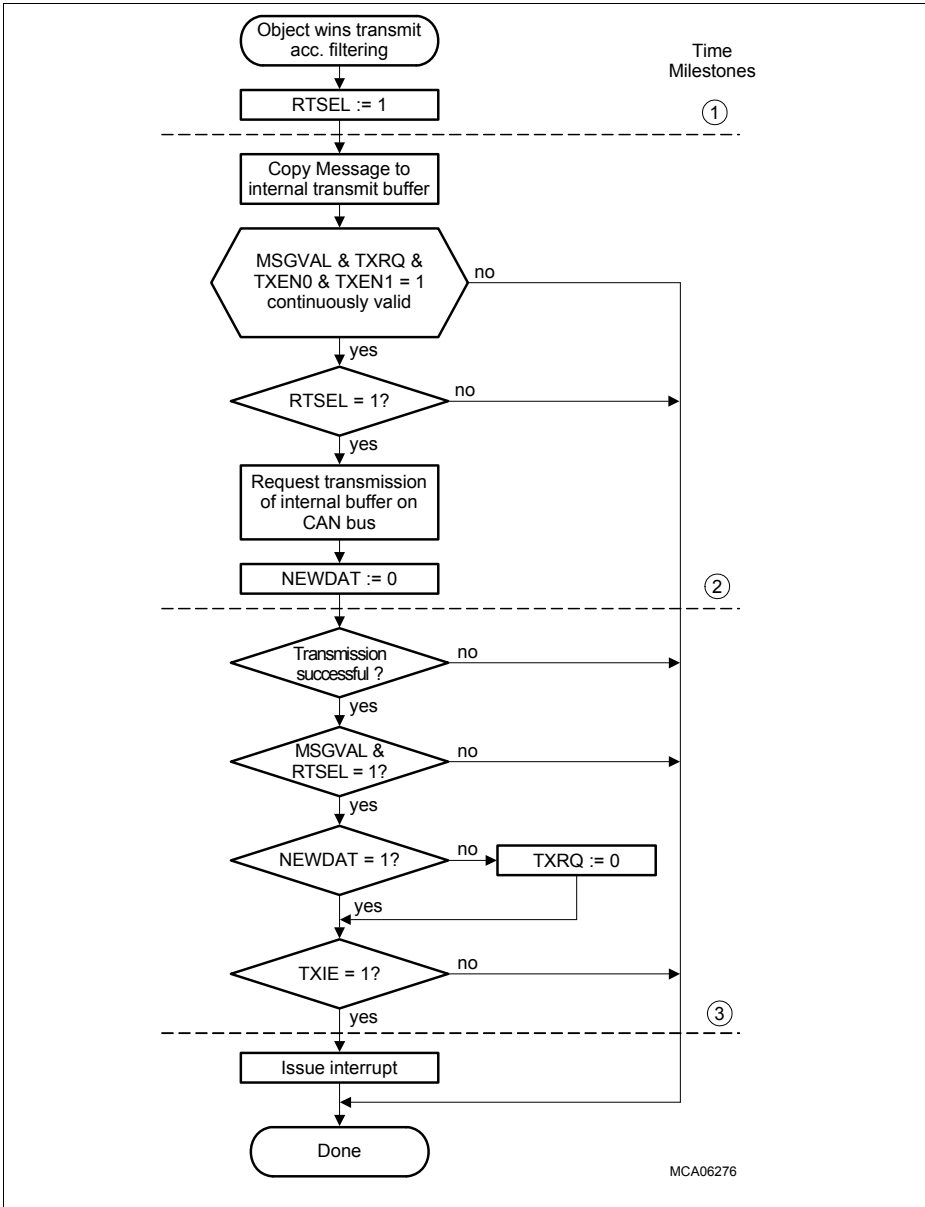


Figure 23-22 Transmission of a Message Object

23.4.12 Message Object Functionality

This chapter describes the functionality of the Message Objects in the MultiCAN+ module.

23.4.12.1 Standard Message Object

A message object is selected as standard message object when bit field MOFCRn.MMC = 0000_B (see [Page 23-113](#)). The standard message object can transmit and receive CAN frames according to the basic rules described in the previous sections. Additional services such as Single Data Transfer Mode or Single Transmit Trial (see following sections) are available and can be individually selected.

23.4.12.2 Single Data Transfer Mode

Single Data Transfer Mode is a useful feature in order to broadcast data over the CAN bus without unintended duplication of information. Single Data Transfer Mode is selected via bit MOFCRn.SDT.

Message Reception

When a received message stored in a message object is overwritten by a new received message, the contents of the first message are lost and replaced with the contents of the new received message (indicated by MSGLST = 1).

If SDT is set (Single Data Transfer Mode activated), bit MSGVAL of the message object is automatically cleared by hardware after the storage of a received Data or Remote Frame. This prevents the reception of further messages.

Message Transmission

When a message object receives a series of multiple remote requests, it transmits several Data Frames in response to the remote requests. If the data within the message object has not been updated in the time between the transmissions, the same data can be sent more than once on the CAN bus.

In Single Data Transfer Mode (SDT = 1), this is avoided because MSGVAL is automatically cleared after the successful transmission of a Data or Remote Frame.

23.4.12.3 Single Transmit Trial

If the bit STT in the message object function register is set (STT = 1), the transmission request is cleared (TXRQ = 0) when the frame contents of the message object have been copied to the internal transmit buffer of the CAN node. Thus, the transmission of the message object is not tried again when it fails due to CAN bus errors.

23.4.12.4 Message Object Format (Classical CAN & CAN FD)

Message objects are used for transmission or reception of CAN data frames. The transmit and receive behavior of a Classical CAN node in Classical CAN (ISO 11898-1:2003(E)) or CAN FD (ISO 11898-1 DIS version 2014) are determined by a node control register bit (i.e. NCRx.FDEN) and two message object function control register bits (i.e. MOFCRn.FDF and MOFCRn.BRS) as shown in [Table 23-18](#).

A CAN node which has been set to work in Classical CAN mode i.e. (NCRx.FDEN = 0) would be able to transmit message objects programmed for Classical CAN Frames (i.e. MOFCRn.FDF = 0, MOFCRn.BRS = 0/1) only. When receiving Classical CAN data frames in a node that works in Classical CAN mode the message object function control register bits (i.e. MOFCRn.FDF and MOFCRn.BRS) are not used and not updated.

In the case when a CAN node set to work in a particular mode (e.g. Classical CAN mode) and receives a message object request for transmission or reception that does not correspond with the CAN mode (i.e. MOFCRn.FDF = 1, MOFCRn.BRS = 0/1), the transmission would be cancelled (i.e. corresponding MOSTATn.TXRQ cleared) and CAN frame not received in message object. The CAN node is not blocked and able to transmit or receive other message objects.

However a CAN node that has been set to work in CAN FD mode i.e. (NCRx.FDEN = 1) is able to transmit and receive Classical CAN Data Frames, Long Data Frames or Long + Fast Data Frames. For message object transmission, message object function control bits (i.e. MOFCRn.FDF & MOFCRn.BRS) determines the CAN frame format to be used and functions as status register bits during message reception.

Note: Remote Frame Requests do not change MOFCRn.FDF and MOFCRn.BRS bits.

23.4.12.5 Message Object FIFO Structure

In case of high CPU load it may be difficult to process a series of CAN frames in time. This may happen if multiple messages are received or must be transmitted in short time.

Therefore, a FIFO buffer structure is available to avoid loss of incoming messages and to minimize the setup time for outgoing messages. The FIFO structure can also be used to automate the reception or transmission of a series of CAN messages and to generate a single message interrupt when the whole CAN frame series is done.

There can be several FIFOs in parallel. The number of FIFOs and their size are limited only by the number of available message objects. A FIFO can be installed, resized and de-installed at any time, even during CAN operation.

The basic structure of a FIFO is shown in [Figure 23-23](#). A FIFO consists of one base object and n slave objects. The slave objects are chained together in a list structure (similar as in message object lists). The base object may be allocated to any list. Although [Figure 23-23](#) shows the base object as a separate part beside the slave objects, it is also possible to integrate the base object at any place into the chain of slave objects. This means that the base object is slave object, too (not possible for gateways).

Controller Area Network Controller (MultiCAN+)

The absolute object numbers of the message objects have no impact on the operation of the FIFO.

The base object does not need to be allocated to the same list as the slave objects. Only the slave object must be allocated to a common list (as they are chained together). Several pointers (BOT, CUR and TOP) that are located in the Message Object n FIFO/Gateway Pointer Register MOFGPRn link the base object to the slave objects, regardless whether the base object is allocated to the same or to another **list** than the slave objects.

The smallest FIFO would be a single message object which is both, FIFO base and FIFO slave (not very useful). The biggest possible FIFO structure would include all message objects of the MultiCAN+ module. Any FIFO sizes between these limits are possible.

In the FIFO base object, the FIFO boundaries are defined. Bit field MOFGPRn.BOT of the base object points to (includes the number of) the bottom slave object in the FIFO structure. The MOFGPRn.TOP bit field points to (includes the number of) the top slave object in the FIFO structure. The MOFGPRn.CUR bit field points to (includes the number of) the slave object that is actually selected by the MultiCAN+ module for message transfer. When a message transfer takes place with this object, CUR is set to the next message object in the list structure of the slave objects (CUR = PNEXT of current object). If CUR was equal to TOP (top of the FIFO reached), the next update of CUR will result in CUR = BOT (wrap-around from the top to the bottom of the FIFO). This scheme represents a circular FIFO structure where the bit fields BOT and TOP establish the link from the last to the first element.

Bit field MOFGPRn.SEL of the base object can be used for monitoring purposes. It makes it possible to define a slave object within the list at which a message interrupt is generated whenever the CUR pointer reaches the value of the SEL pointer. Thus SEL makes it possible to detect the end of a predefined message transfer series or to issue a warning interrupt when the FIFO becomes full.

Controller Area Network Controller (MultiCAN+)

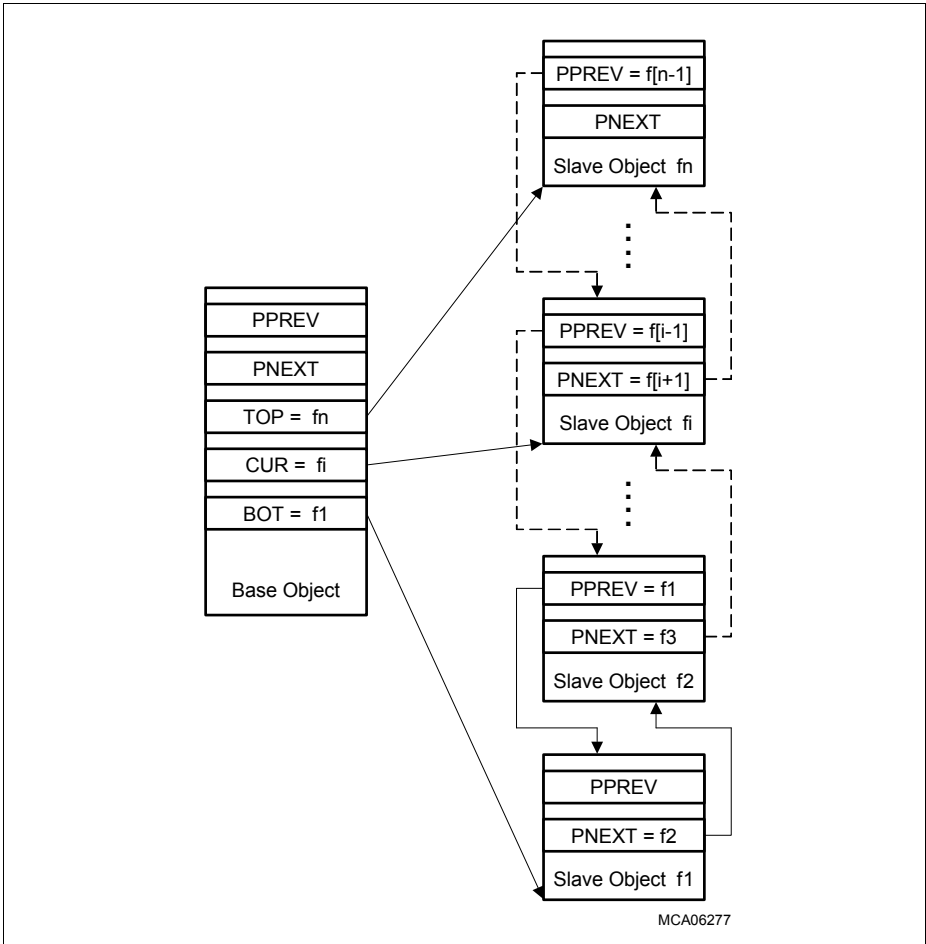


Figure 23-23 FIFO Structure with FIFO Base Object and n FIFO Slave Objects

23.4.12.6 Receive FIFO

The Receive FIFO structure is used to buffer incoming (received) Remote or Data Frames.

A Receive FIFO is selected by setting $\text{MOFCRn.MMC} = 0001_{\text{B}}$ in the FIFO base object. This MMC code automatically designates a message object as FIFO base object. The message modes of the FIFO slave objects are not relevant for the operation of the Receive FIFO.

When the FIFO base object receives a frame from the CAN node it belongs to, the frame is not stored in the base object itself but in the message object that is selected by the base object's MOFGPRn.CUR pointer. This message object receives the CAN message as if it is the direct receiver of the message. However, $\text{MOFCRn.MMC} = 0000_{\text{B}}$ is implicitly assumed for the FIFO slave object, and a standard message delivery is performed. The actual message mode (MMC setting) of the FIFO slave object is ignored. For the slave object, no acceptance filtering takes place that checks the received frame for a match with the identifier, IDE bit, and DIR bit.

With the reception of a CAN frame, the current pointer CUR of the base object is set to the number of the next message object in the FIFO structure. This message object will then be used to store the next incoming message.

If bit field MOFCRn.OVIE ("Overflow Interrupt Enable") of the FIFO base object is set and the current pointer MOFGPRn.CUR becomes equal to MOFGPRn.SEL , a FIFO overflow interrupt request is generated. This interrupt request is generated on interrupt node TXINP of the base object immediately after the storage of the received frame in the slave object. Transmit interrupts are still generated if TXIE is set.

A CAN message is stored in FIFO base and slave object only if $\text{MSGVAL} = 1$.

In order to avoid direct reception of a message by a slave message object, as if it was an independent message object and not a part of a FIFO, the bit RXEN of each slave object must be cleared. The setting of the bit RXEN is "don't care" only if the slave object is located in a list not assigned to a CAN node.

23.4.12.7 Transmit FIFO

The Transmit FIFO structure is used to buffer a series of Data or Remote Frames that must be transmitted. A transmit FIFO consists of one base message object and one or more slave message objects.

A Transmit FIFO is selected by setting $\text{MOFCRn.MMC} = 0010_{\text{B}}$ in the FIFO base object. Unlike the Receive FIFO, slave objects assigned to the Transmit FIFO must explicitly set their bit fields $\text{MOFCRn.MMC} = 0011_{\text{B}}$. The CUR pointer in all slave objects must point back to the Transmit FIFO Base Object (to be initialized by software).

Controller Area Network Controller (MultiCAN+)

The MOSTATn.TXEN1 bits (Transmit Enable 1) of all message objects except the one which is selected by the CUR pointer of the base object must be cleared by software. TXEN1 of the message (slave) object selected by CUR must be set. CUR (of the base object) may be initialized to any FIFO slave object.

When tagging the message objects of the FIFO as valid to start the operation of the FIFO, then the base object must be tagged valid (MSGVAL = 1) first.

Before a Transmit FIFO becomes de-installed during operation, its slave objects must be tagged invalid (MSGVAL = 0).

The Transmit FIFO uses the TXEN1 bit in the Message Object Status Register of all FIFO elements to select the actual message for transmission. Transmit acceptance filtering evaluates TXEN1 for each message object and a message object can win transmit acceptance filtering only if its TXEN1 bit is set. When a FIFO object has transmitted a message, the hardware clears its TXEN1 bit in addition to standard transmit postprocessing (clear TXRQ, transmit interrupt etc.), and moves the CUR pointer in the next FIFO base object to be transmitted. TXEN1 is set automatically (by hardware) in the next message object. Thus, TXEN1 moves along the Transmit FIFO structure as a token that selects the active element.

If bit field MOFCRn.OVIE (“Overflow Interrupt Enable”) of the FIFO base object is set and the current pointer CUR becomes equal to MOFGPRn.SEL, a FIFO overflow interrupt request is generated. The interrupt request is generated on interrupt node RXINP of the base object after postprocessing of the received frame. Receive interrupts are still generated for the Transmit FIFO base object if bit RXIE is set.

23.4.12.8 Gateway Mode

The Gateway Mode makes it possible to establish an automatic information transfer between two independent CAN buses without CPU interaction.

The Gateway Mode operates on message object level. In Gateway mode, information is transferred between two message objects, resulting in an information transfer between the two CAN nodes to which the message objects are allocated. A gateway may be established with any pair of CAN nodes, and there can be as many gateways as there are message objects available to build the gateway structure.

Gateway Mode is selected by setting MOFCRs.MMC = 0100_B for the gateway source object s. The gateway destination object d is selected by the MOFGPRd.CUR pointer of the source object. The gateway destination object only needs to be valid (its MSGVAL = 1). All other settings are not relevant for the information transfer from the source object to the destination object.

Controller Area Network Controller (MultiCAN+)

Gateway source object behaves as a standard message object with the difference that some additional actions are performed by the MultiCAN+ module when a CAN frame has been received and stored in the source object (see [Figure 23-24](#)):

1. If bit MOFCRs.DLCC is set, the data length code MOFCRs.DLC is copied from the gateway source object to the gateway destination object.
2. If bit MOFCRs.IDC is set, the identifier MOARs.ID and the identifier extension MOARs.IDE are copied from the gateway source object to the gateway destination object.
3. If bit MOFCRs.DATC is set, the data bytes stored in the two data registers MODATALs and MODATAHs are copied from the gateway source object to the gateway destination object. All 8 data bytes are copied, even if MOFCRs.DLC indicates less than 8 data bytes.
4. If bit MOFCRs.GDFS is set, the transmit request flag MOSTATd.TXRQ is set in the gateway destination object.
5. The receive pending bit MOSTATd.RXPND and the new data bit MOSTATd.NEWDAT are set in the gateway destination object.
6. A message interrupt request is generated for the gateway destination object if its MOSTATd.RXIE is set.
7. The current object pointer MOFGPRs.CUR of the gateway source object is moved to the next destination object according to the FIFO rules as described on [Page 23-57](#).
A gateway with a single (static) destination object is obtained by setting MOFGPRs.TOP = MOFGPRs.BOT = MOFGPRs.CUR = destination object.

The link from the gateway source object to the gateway destination object works in the same way as the link from a FIFO base to a FIFO slave. This means that a gateway with an integrated destination FIFO may be created; in [Figure 23-23](#), the object on the left is the gateway source object and the message object on the right side is the gateway destination objects.

The gateway operates equivalent for the reception of data frames (source object is receive object, i.e. DIR = 0) as well as for the reception of Remote Frames (source object is transmit object).

Controller Area Network Controller (MultiCAN+)

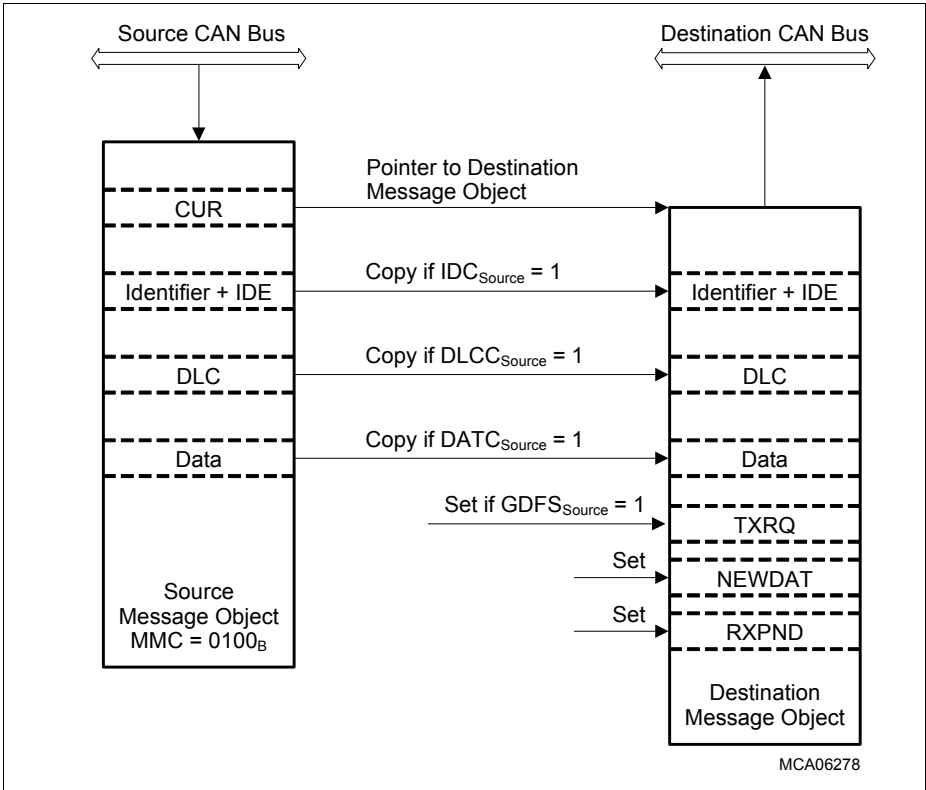


Figure 23-24 Gateway Transfer from Source to Destination

23.4.12.9 Foreign Remote Requests

When a Remote Frame has been received on a CAN node and is stored in a message object, a transmit request is set to trigger the answer (transmission of a Data Frame) to the request or to automatically issue a secondary request. If the Foreign Remote Request Enable bit MOFCRn.FRREN is cleared in the message object in which the remote request is stored, MOSTATn.TXRQ is set in the same message object.

If bit FRREN is set (FRREN = 1: foreign remote request enabled), TXRQ is set in the message object that is referenced by pointer MOFGPRn.CUR. The value of CUR is, however, not changed by this feature.

Although the foreign remote request feature works independently of the selected message mode, it is especially useful for gateways to issue a remote request on the source bus of a gateway after the reception of a remote request on the gateway destination bus. According to the setting of FRREN in the gateway destination object, there are two capabilities to handle remote requests that appear on the destination side (assuming that the source object is a receive object and the destination is a transmit object, i.e. $DIR_{source} = 0$ and $DIR_{destination} = 1$):

FRREN = 0 in the Gateway Destination Object

1. A Remote Frame is received by gateway destination object.
2. TXRQ is set automatically in the gateway destination object.
3. A Data Frame with the current data stored in the destination object is transmitted on the destination bus.

FRREN = 1 in the Gateway Destination Object

1. A Remote Frame is received by gateway destination object.
2. TXRQ is set automatically in the gateway source object (must be referenced by CUR pointer of the destination object).
3. A remote request is transmitted by the source object (which is a receive object) on the source CAN bus.
4. The receiver of the remote request responds with a Data Frame on the source bus.
5. The Data Frame is stored in the source object.
6. The Data Frame is copied to the destination object (gateway action).
7. TXRQ is set in the destination object (assuming $GDFS_{source} = 1$).
8. The new data stored in the destination object is transmitted on the destination bus, in response to the initial remote request on the destination bus.

23.4.12.10 CAN FD - 64 byte Messages

In order to support higher than 8 data bytes payload (e.g 64bytes data payload) using current message object structure the data byte buffer (MODATALn and MODATAHn) has to be extended.

Controller Area Network Controller (MultiCAN+)

Thus an additional CAN FD 64 bytes message mode, MOFCR.MMC=5 is added. When CAN FD 64 bytes message mode is selected, additional message objects are used to store the extra data bytes.

The additional message objects used are specified by the pointer on MOFGPR.BOT and MOFGPR.TOP register bits. (i.e those message object registers (MOFCR, MOFGPR, MOIPR, MOAMR, MODATAL, MODATAH and MOAR) are used to store the extra data bytes with their alternate register view on EMOzDATA_n at the same address location. MOCTR is the only register that is not used for data storage it retains its original function.

As an example for a 64 byte message, data bytes 0-7 are stored in the message object, similar to a standard message object, data bytes 8-35 are stored in the message object to which MOFGPR.BOT refers to and data bytes 36-63 are stored in the message object to which MOFGPR.TOP refers to. Data byte 0 refers to the first byte transferred within a CAN frame. For shorter than 64 bytes messages, unused data bytes are padded upon reception and ignored upon transmission.

The additional message objects chosen to store the extra data bytes must not take part in CAN communication i.e either they are allocated to a list that does not belong to an active CAN node or have that particular message object MOCTR register bits MOCTR_n.RXEN, MOCTR_n.TXEN0 (and/or TXEN1, TXRQ) cleared, i.e (RXEN=0, TXEN0=0, TXEN1=0 and TXRQ=0). MOCTR.MSGVAL remains the same, data is stored in the extra message objects when MSGVAL is set. When MSGVAL of the base object is cleared, an ongoing data transfer to TOP and BOT objects are aborted, thereby freeing objects BOT and TOP. Clearing MSGVAL bits of BOT, TOP objects and the extra message objects do not stop an ongoing data transfer and are not considered at all.

For optimum performance it is recommended to place the extra message objects outside active CAN node lists, so they do not take part in CAN acceptance filtering.

23.4.13 Measurement for Oscillator Calibration

The chip can be driven by an oscillator which is less accurate than required by the CAN specification, if it is possible to detect the deviation and to change the oscillator frequency during runtime. The MultiCAN module can measure the time interval between the start of a CAN frame and a certain edge on the bus. The timer hardware is accessed via Measurement Control Register MECR (see [Page 23-91](#)) and Measurement Status Register MESTAT (see [Page 23-93](#)).

A 16-bit timer is cleared and started by the first falling (dominant) edge of a CAN frame on the input line of the CAN node (selected by MECR.NODE). The timer is incremented by the module control clock f_{CLC} . The timer content is captured when an edge specified by MECR.ANYED is detected on the input line of the CAN node. However, the timer must have reached threshold MECR.TH before; so edges occurring earlier cannot cause a capture. Only the first edge after threshold is considered, and the node itself may not be sending. Bit field CAPT in the status register contains the captured timer value, and bit CAPRED indicates the type of edge that caused the capture. On a capture event, bit

Controller Area Network Controller (MultiCAN+)

CAPE in the same register will be set. If bit MECR.CAPEIE = 1, a node interrupt will be generated. With the starting edge of the following CAN frame, a new measurement cycle begins. To avoid an overflow on the measurement and to get a measurement result MESTAT.CAPE bit has to be cleared before the measurement end condition occurs.

To avoid wrong capture results due to timer overflow, the timer will be stopped when $FFFF_H$ is reached. If MECR.TH = 0000_H , the timer is always stopped and the capture function is disabled.

Some important issues should be mentioned:

- The timing of the measured CAN frame should be as exact as possible.
- No unadjusted CAN stations - including the selected MultiCAN node - should participate actively in the communication. The selected node can be set to Analyzer Mode.
- When setting the timer threshold and evaluating the capture result, the tolerances including the oscillator deviation must be considered.
- It may not be possible to use a well-defined CAN message for measurement. Then worst-case-scenarios according to the CAN protocol must be observed. E.g. in a data or remote frame, the first rising (recessive) edge might occur after five bit times, and the next falling one after additional five bit times.
- Also error and overload frames which may lead to unexpected capture results should be taken into account.
- Edges occurring a considerable time after the start of a CAN frame will yield good resolution but may produce ambiguous results if the oscillator is not yet sufficiently adjusted. Therefore, only one step for measurement and calibration may not be sufficient.
- As there is only one oscillator on the chip, its calibration influences the timing of the whole system and particularly the timing of other CAN nodes.

Controller Area Network Controller (MultiCAN+)
23.5 Use Case Example MultiCAN+

This section explains the core functionality of the MultiCAN+ module with a code example. The example realizes sending a CAN message via internal CAN-bus from node 0 to node 1.

The MultiCAN+ module for the TC26x B-Step consists of module (i.e MultiCAN with 5 CAN nodes), representing serial communication interfaces. Each CAN node can either be connected to a port or to the internal CAN bus (see). So a maximum of CAN channels can be realized with the MultiCAN+ module (For more information and further functions see [Section 23.2](#)).

All CAN nodes share a common set of 256 message objects. A message object function like a container for a specific CAN message; both transmitting and receiving of CAN messages can be realized. The message objects are organized in double-chained lists, each list is dedicated to a certain CAN node (list 1 is node 0, list 2 is node 1, etc.). After a reset, all message objects are unallocated and therefore assigned to list 0, this list does not belong to a CAN node. During initialization it is possible to allocate the message objects individually to certain CAN node lists. Each allocated message object can be set up with all necessary information like the message ID etc. (see chapter 26.2 and following for further explanations of the CAN module).

In the use case example CAN node 0 will send a CAN message via internal CAN bus to CAN node 1. The transmitting message object (MO) will be MO 0 and the receiving message object will be MO 1. As soon as the CAN frame successfully receives at MO 1 a receive interrupt will occur. The initialization process follows the following order:

1. Load global MultiCAN+ module registers
2. Initialize the CAN nodes
3. Allocate the message objects to the CAN nodes
4. Initialize the message objects
5. Start the CAN nodes
6. Start transmit request for CAN message from node 0 to node 1.
7. Receive message at node 1, Rx interrupt occurs.

Step description to initialize the MultiCAN+ module:

(Line 1) reset ENDINIT to get access to ENDINIT protected registers. (see also ENDINIT protection chapter 8.9.3(LINK)).

(Line 2) enable control of the module, in clock control register CLC (**CLC**).

(Line 3) read back (dummy variable has to be defined). The reading process ensures that the write process from line 2 is done.

(Line 4) load fractional divider to $f_{CAN} = f_{ASYN_CAN}$ (see also 26.3.2 Clock Control (**Fractional Divider**), f_{ASYN_CAN} see **MCR**).

(Line 5) set ENDINIT to lock the protected register again.

Controller Area Network Controller (MultiCAN+)

(Line 6) set clk source to f_{ASYN_CAN} in module control register MCR(**MCR**).

Note: The reconfiguration of the clk source must be done by two writes and a certain delay between these. See MCR for more details.

(Line 7) The setting of protection CCE[6] and INIT[0] activates the initialization and configuration mode for CAN node 0. This is necessary for the upcoming changes in CAN node 0. (see also node configuration register NCRn (**CAN_NCRx (x = 0-4)**))

(Line 8) This example uses the internal loop-back mode, so this line connects CAN node 0 to the internal CAN Bus. (see also Figure 26-13(**Figure 23-16**) and node port control register NPCR(**CAN_NPCRx (x = 0-4)**))

(Line 9) The CAN bit time is subdivided into different segments. (see also 26.3.6.1 Bit Timing Unit(**Bit Timing Unit**)). Assuming $f_{CAN} = 100\text{MHz}$ this line then sets the baudrate to 500kBaud and the sample point to 80% in the node bit timing register NBTR (**CAN_NBTRx (x = 0-4)**).

Note: The FM PLL should be off. Otherwise a stable CAN communication is not ensured

(Line 10 - 12) same as line 7 - 9 but with node 1.

(Line 13) The allocation of the message objects to certain CAN node lists function via a list command panel. The panel control register PANCTR(**PANCTR**) can only be written if both busy flags are reseted. Therefore this while loop waits until the register is ready.

(Line 14) This line allocates message object 0 to list 1 (belongs to CAN node 0).

(Line 15) see line 13.

(Line 16) This line allocates message object 1 to list 2 (belongs to CAN node 0).

(Line 17) Initialize MO 0 in the message object register MOCTR(**CAN_MOCTRz (z = 0-254)**) as transmit MO by setting bits [27:25]. Set also MSGVAL[21], that's the main switch bit of the MO.

(Line 18) this line sets the message data length of MO 0 to 8 bytes.(see also message object function control register MOFCR (**CAN_MOFCRn (n = 0-255)**)).

(Line 19) the message ID of MO 0 gets set to standard message (11 bit length) with a message ID of 0xFF in the Message object arbitration register MOAR (**CAN_MOARn (n = 0-255)**).

(Line 20) Initialize MO 1 in the message object register MOCTR as receive MO by setting bit RXEN[23]. Set also MSGVAL[21], that's the main switch bit of the MO.

(Line 21) the receive interrupt gets enabled in the message object function control register MOFCR .

(Line 22) same as Line 19 but for MO 1. The same message ID is necessary to receive CAN messages from the bus with this specific ID.

(Line 23) the Rx interrupt gets connected to the CAN interrupt output line 1.(see also MOIPR(**CAN_MOIPRn (n = 0-255)**))

Controller Area Network Controller (MultiCAN+)

(Line 24) This line enables the Rx interrupt in the service request control register SRC_CANINT1 and sets the interrupt priority to CAN_SR1INT (1...255)

(Line 25) The interruptHandlerInstall function gets called (this function has to be written first, see interrupt handler example (Chapter 17.10.1) for more details). This function installs the interrupt service routine (ISR) entry address in the interrupt vector array with the priority CAN_SR1INT.

(Line 26) The 4 lower data bytes are getting loaded in the MODATAL(**CAN_MODATALn (n = 0-255)**) register. the data itself is here just an example.

(Line 27) The 4 higher data bytes are getting loaded in the MODATAH(**CAN_MODATAHn (n = 0-255)**) register. (just an data example as well)

(Line 28) This line resets INIT[0] and CCE[6] bit in the node control register from node 0. The node is now synchronizing itself to the bus.

(Line 29) same as line 28, but with node 1.

(Line 30) The NEWDATA bit gets set in the MOCTR register, this should always be done after a write process in the data registers from line 26/27. The transmit request bit TXRQ gets set, this starts the transmission of the CAN message. The RTSEL gets reset to ensure the transmission.

(Line 31) If the message is received the MOCTR.NEWDAT[19] bit in the receive MO 1 is set. This line just waits for the reception. The occurrence of the Rx interrupt can also be used as a proof that the message arrived.

Note: The Rx ISR function prototype would be: void CAN1_Rx_irq (void); here could be your ISR code;

Initialization of the MultiCAN+ module:

```
// Load global MultiCAN+ registers:
(1)  SCU_vResetENDINIT (0);    // enable ENDINIT reg
(2)  CAN_CLC = 0x000;         // disable module control
(3)  dummy = CAN_CLC;        //read to check the made changes
(4)  CAN_FDR = (1 << 14) | 0x3FF;    //DM=1,STEP=1023
(5)  SCU_vSetENDINIT (0);    // lock ENDINIT reg
(6)  CAN_MCR = 0x1;         // CLKSEL=1

//Init CAN nodes 0 and 1:
(7)  CAN_NCR0 = (1 << 6) | 1;    // CCE=1,INIT=1
(8)  CAN_NPCR0 = (1 << 8);    // LBM=1
(9)  CAN_NBTR0 = 0x3EC9;    //set bit segments
(10) CAN_NCR1 = (1 << 6) | 1;    // CCE=1,INIT=1
(11) CAN_NPCR1 = (1 << 8);    // LBM=1
```

Controller Area Network Controller (MultiCAN+)

```

(12) CAN_NBTR1 = 0x3EC9;          //set bit segments
//Allocate message objects to CAN nodes:
(13) while(CAN_PANCTR.U & (0x00000100 | 0x00000200)); // busy?
(14) CAN_PANCTR = (1 << 24)|(0 << 16)| 2; // MO 0 to list 1
(15) while(CAN_PANCTR.U & (0x00000100 | 0x00000200)); // busy?
(16) CAN_PANCTR = (2 << 24)|(1 << 16)| 2; // MO 1 to list 2
//Init MO_0 (list 1, node 0)
(17) CAN_MOCTR0 = (1<<27)|(1<<26)|(1<<25)|(1<<21); // set to Tx
(18) CAN_MOFCR0 = (8 << 24);          // DLC=8
(19) CAN_MOAR0 = (1 << 30)|(0xFF << 18); // PRI=01, ID=0xFF
//Init MO_1 (list 2, node 1)
(20) CAN_MOCTR1 = (1 << 23)|(1 << 21); // set to Rx
(21) CAN_MOFCR1 = (1 << 16);          // RXIE=1
(22) CAN_MOAR1 = (1 << 30)|(0xFF << 18); // PRI=01, ID=0xFF
(23) CAN_MOIPR1 = 0x1;              // RXINP=1 -> select INT_01
(24) SRC_CANINT1 = (1 << 10) | CAN_SRN1INT; // enable INT_01
(25) interruptHandlerInstall (CAN_SRN1INT, & CAN1_Rx_irq);
// Load Data, Start CAN nodes
(26) CAN_MODATAL0 = 0x0D0D0D0D; // load data, lower 4 Bytes
(27) CAN_MODATAH0 = 0x0E0E0E0E; // load data, higher 4 Bytes
(28) CAN_NCR0 &= ~(1<<6)| 1; // reset CCE and INIT
(29) CAN_NCR1 &= ~(1<<6)| 1; // reset CCE and INIT
// set transmit request to send message
(30) CAN_MOCTR0 = (1<<24)|(1<<19)|(1<<6); //TXRQ=1,NEWDAT=1
(31) while((CAN_MOSTAT1.B.NEWDAT) != 1); //check if Rx

```

Note: Before an interrupt is able to occur, the interrupt system has to be globally enabled. The Interrupt Control Register (ICR) holds the global interrupt enable bit (ICR.IE) which enables the CPU service request system. Most compiler support the attribute (or similar):

__enable();

to set this bit. (See also Architecture Manual for more details)

Controller Area Network Controller (MultiCAN+)

23.6 MultiCAN+ Kernel Registers

This section describes the kernel registers of the MultiCAN+ module. All MultiCAN+ kernel register names described in this section are also referenced in other parts of the TC26x B-Step User’s Manual by the module name prefix “CAN_”.

MultiCAN+ Kernel Register Overview

The MultiCAN+ Kernel include three blocks of registers:

- Global Module Registers
- Node Registers, for each CAN node *x*
- Message Object Registers, for each message object *n*

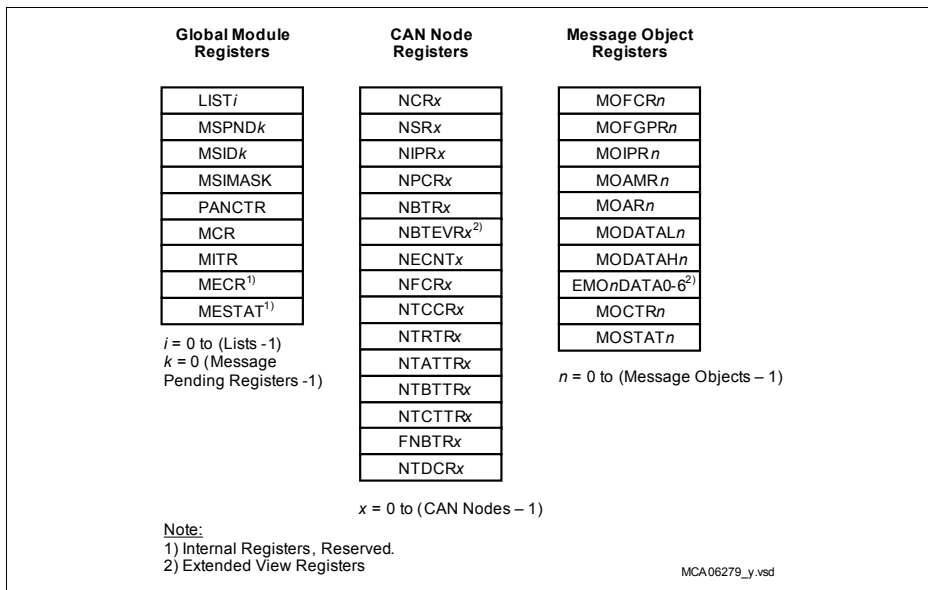


Figure 23-25 MultiCAN+ Kernel Registers

The registers of the MultiCAN+ module kernel are listed below.

Table 23-9 Registers Address Space - MultiCAN+ Kernel Registers

Module	Base Address	End Address	Note
CAN	F001 8000 _H	F001 BFFF _H	-

Controller Area Network Controller (MultiCAN+)
Table 23-10 Registers Overview - MultiCAN+ Kernel Registers

Short Name	Description	Offset Addr ¹⁾	Access Mode ²⁾		Reset	Description see
			Read	Write		
Global Module Registers						
LISTi	List Register i	0100 _H + i × 4 _H	U, SV	U, SV, P	Application Reset	Page 23-86
MSPNDk	Message Pending Register k	0140 _H + k × 4 _H	U, SV	U, SV, P	Application Reset	Page 23-88
MSIDk	Message Index Register k	0180 _H + k × 4 _H	U, SV	U, SV, P	Application Reset	Page 23-89
MSIMASK	Message Index Mask Register	01C0 _H	U, SV	U, SV, P	Application Reset	Page 23-90
PANCTR	Panel Control Register	01C4 _H	U, SV	U, SV, P	Application Reset	Page 23-79
MCR	Module Control Register	01C8 _H	U, SV	U, SV, P	Application Reset	Page 23-83
MITR	Module Interrupt Trigger Reg.	01CC _H	U, SV	U, SV, P	Application Reset	Page 23-85
MECR	Measurement Control Register	01D0 _H	U, SV	U, SV, P	Application Reset	Page 23-91
MESTAT	Measurement Status Register	01D4 _H	U, SV	U, SV, P	Application Reset	Page 23-93
CAN Node Registers						
NCRx	Node x Control Register	0200 _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-94

Controller Area Network Controller (MultiCAN+)
Table 23-10 Registers Overview - MultiCAN+ Kernel Registers (cont'd)

Short Name	Description	Offset Addr ¹⁾	Access Mode ²⁾		Reset	Description see
			Read	Write		
NSRx	Node x Status Register	0204 _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-98
x	Node x Interrupt Pointer Reg.	0208 _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-10 2
NPCR _x	Node x Port Control Register	020C _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-10 4
NBTR _x	Node x Bit Timing Register	0210 _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-10 5
NBTEVR _x	Node x Bit Timing Extended View Register	0210 _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-10 7
NECNT _x	Node x Error Counter Register	0214 _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-11 2
NFCR _x	Node x Frame Counter Register	0218 _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-11 3
NTCCR _x	Node x Timer Clock Control Register	021C _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-11 7
NTRTR _x	Node x Timer Receive Timeout Register	0220 _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-11 9
NTATTR _x	Node x Timer A Transmit Trigger Register	0224 _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-12 0
NTBTTR _x	Node x Timer B Transmit Trigger Register	0228 _H + x × 100 _H	U, SV	U, SV, P	Application Reset	Page 23-12 1

Controller Area Network Controller (MultiCAN+)
Table 23-10 Registers Overview - MultiCAN+ Kernel Registers (cont'd)

Short Name	Description	Offset Addr ¹⁾	Access Mode ²⁾		Reset	Description see
			Read	Write		
NTCTTRx	Node x Timer C Transmit Trigger Register	$022C_H + x \times 100_H$	U, SV	U, SV, P	Application Reset	Page 23-12 2
FNBRx	Fast Node x Bit Timing Register	$0238_H + x \times 100_H$	U, SV	U, SV, P	Application Reset	Page 23-10 9
NTDCRx	Node x Transmitter Delay Compensation Register	$023C_H + x \times 100_H$	U, SV	U, SV, P	Application Reset	Page 23-11 0

Message Object Registers

MOFCRn	Message Object n Function Control Register	$1000_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	Page 23-13 3
MOFGPRn	Message Object n FIFO/Gateway Pointer Register	$1004_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	Page 23-13 9
MOIPRn	Message Object n Interrupt Pointer Register	$1008_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	Page 23-13 1
MOAMRn	Message Object n Acceptance Mask Register	$100C_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	Page 23-14 1
MODATALn	Message Object n Data Register Low	$1010_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	Page 23-14 5
MODATAHn	Message Object n Data Register High	$1014_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	Page 23-14 6

Controller Area Network Controller (MultiCAN+)
Table 23-10 Registers Overview - MultiCAN+ Kernel Registers (cont'd)

Short Name	Description	Offset Addr ¹⁾	Access Mode ²⁾		Reset	Description see
			Read	Write		
EMOnDAT A	Extended Message Object n Data Register 0	$1000_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	Page 23-14 7
	Extended Message Object n Data Register 1	$1004_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	
	Extended Message Object n Data Register 2	$1008_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	
	Extended Message Object n Data Register 3	$100C_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	
	Extended Message Object n Data Register 4	$1010_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	
	Extended Message Object n Data Register 5	$1014_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	
	Extended Message Object n Data Register 6	$1018_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	
MOARn	Message Object n Arbitration Register	$1018_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	Page 23-14 2
MOCTRn MOSTATn	Message Object n Control Reg. Message Object n Status Reg.	$101C_H + n \times 20_H$	U, SV	U, SV, P	Application Reset	Page 23-12 3 Page 23-12 6

1) The absolute register address is calculated as follows:

Module Base Address ([Table 23-9](#)) + Offset Address (shown in this column)

Further, the following ranges for parameters i, k, x, and n are valid: i = 0-15, k = 0-7, x = 0-4, n = 0-255.

2) Accesses to empty addresses: nBE

List of Access Protection Abbreviations

U - User Mode

Controller Area Network Controller (MultiCAN+)

SV - Supervisor Mode

BE - Bus Error

nBE - no Bus Error

P - Access Protection, as defined by the ACCEN Register

E - ENDINIT

SE - Safety ENDINIT

Controller Area Network Controller (MultiCAN+)

Figure 23-26 shows the MultiCAN+ register address map.

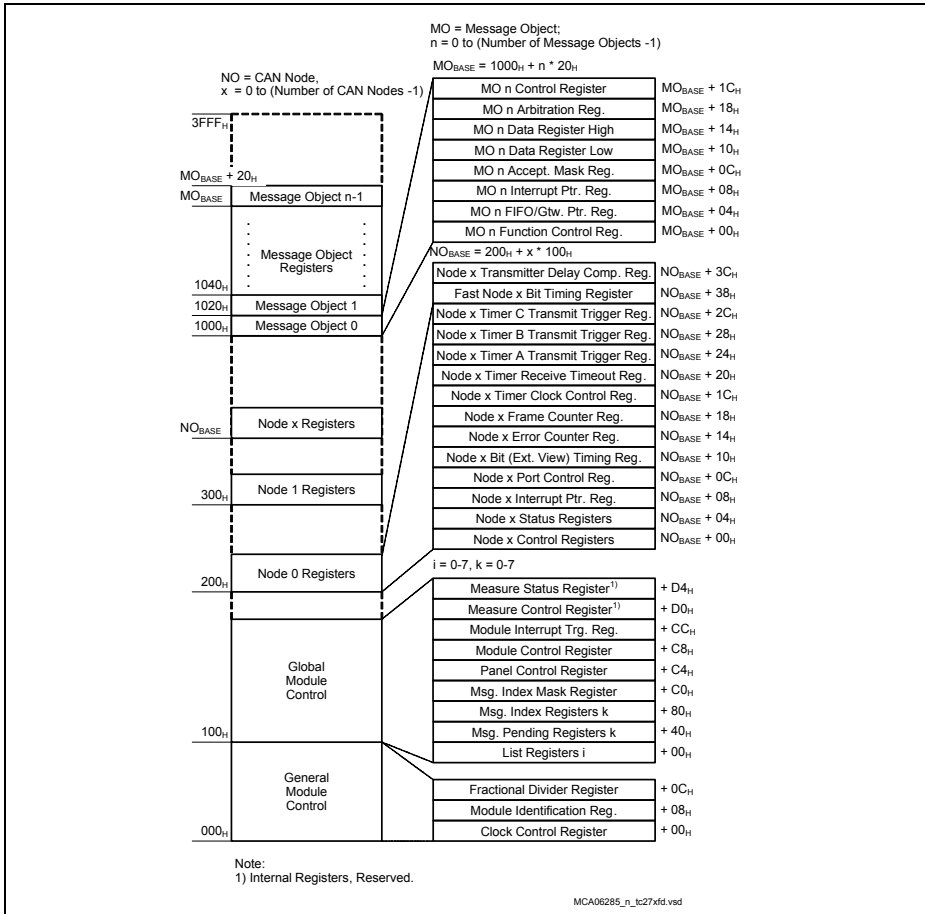


Figure 23-26 MultiCAN+ Register Address Map

23.6.1 Global Module Registers

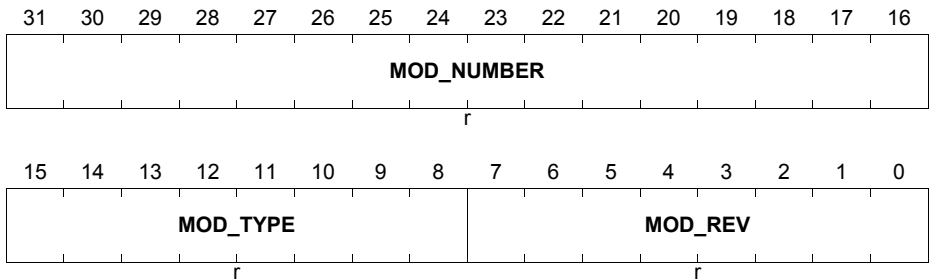
All list operations such as allocation, de-allocation and relocation of message objects within the list structure are performed via the Command Panel. It is not possible to modify the list structure directly by software by writing to the message objects and the LIST registers.

Module Identification Register.

Controller Area Network Controller (MultiCAN+)

ID

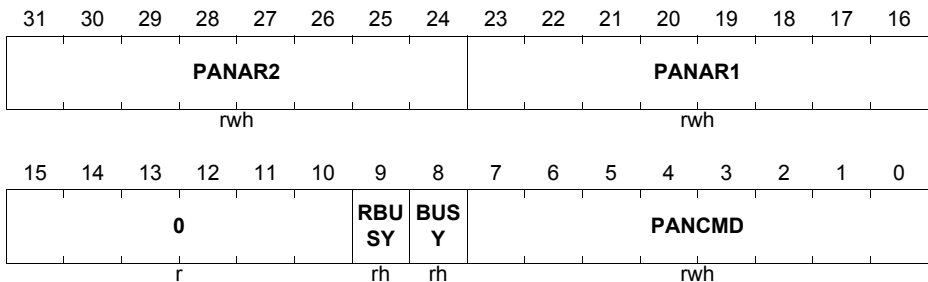
Module Identification Register (008_H) Reset Value: 00B5 C0XX_H



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number MOD_REV defines the revision number. The value of a module revision starts with 01 _H (first revision).
MOD_TYPE	[15:8]	r	Module Type C0 _H Define the module as a 32-bit module.
MOD_NUMBER	[31:16]	r	Module Number Value This bit field defines the MultiCAN+ module identification number (=00B5H)

Controller Area Network Controller (MultiCAN+)

The Panel Control Register PANCTR is used to start a new command by writing the command arguments and the command code into its bit fields.

PANCTR
Panel Control Register
(1C4_H)
Reset Value: 0000 0301_H


Field	Bits	Type	Description
PANCMD	[7:0]	rwh	Panel Command This bit field is used to start a new command by writing a panel command code into it. At the end of a panel command, the NOP (no operation) command code is automatically written into PANCMD. The coding of PANCMD is defined in Table 23-11 .
BUSY	8	rh	Panel Busy Flag 0 _B Panel has finished command and is ready to accept a new command. 1 _B Panel operation is in progress.
RBUSY	9	rh	Result Busy Flag 0 _B No update of PANAR1 and PANAR2 is scheduled by the list controller. 1 _B A list command is running (BUSY = 1) that will write results to PANAR1 and PANAR2, but the results are not yet available.
PANAR1	[23:16]	rwh	Panel Argument 1 See Table 23-11 .
PANAR2	[31:24]	rwh	Panel Argument 2 See Table 23-11 .
0	[15:10]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)
Panel Commands

A panel operation consists of a command code (PANCMD) and up to two panel arguments (PANAR1, PANAR2). Commands that have a return value deliver it to the PANAR1 bit field. Commands that return an error flag deliver it to bit 31 of the Panel Control Register, this means bit 7 of PANAR2.

Table 23-11 Panel Commands

PANCMD	PANAR2	PANAR1	Command Description
00 _H	–	–	No Operation Writing 00 _H to PANCMD has no effect. No new command is started.
01 _H	Result: Bit 7: ERR Bit 6-0: undefined	–	Initialize Lists Run the initialization sequence to reset the CTRL and LIST fields of all message objects. List registers LIST[7:0] are set to their reset values. This results in the de-allocation of all message objects. The initialization command requires that bits NCRx.INIT and NCRx.CCE are set for all CAN nodes. Bit 7 of PANAR2 (ERR) reports the success of the operation: 0 _B Initialization was successful 1 _B Not all NCRx.INIT and NCRx.CCE bits are set. Therefore, no initialization is performed. The initialize lists command is automatically performed with each reset of the MultiCAN+ module, but with the exception that all message object registers are reset, too.
02 _H	Argument: List Index	Argument: Message Object Number	Static Allocate Allocate message object to a list. The message object is removed from the list that it currently belongs to, and appended to the end of the list, given by PANAR2. This command is also used to deallocate a message object. In this case, the target list is the list of unallocated elements (PANAR2 = 0).

Controller Area Network Controller (MultiCAN+)

Table 23-11 Panel Commands (cont'd)

PANCMD	PANAR2	PANAR1	Command Description
03 _H	Argument: List Index Result: Bit 7: ERR Bit 6-0: undefined	Result: Message Object Number	Dynamic Allocate Allocate the first message object of the list of unallocated objects to the selected list. The message object is appended to the end of the list. The message number of the message object is returned in PANAR1. An ERR bit (bit 7 of PANAR2) reports the success of the operation: 0 _B Success. 1 _B The operation has not been performed because the list of unallocated elements was empty.
04 _H	Argument: Destination Object Number	Argument: Source Object Number	Static Insert Before Remove a message object (source object) from the list that it currently belongs to, and insert it before a given destination object into the list structure of the destination object. The source object thus becomes the predecessor of the destination object.
05 _H	Argument: Destination Object Number Result: Bit 7: ERR Bit 6-0: undefined	Result: Object Number of inserted object	Dynamic Insert Before Insert a new message object before a given destination object. The new object is taken from the list of unallocated elements (the first element is chosen). The number of the new object is delivered as a result to PANAR1. An ERR bit (bit 7 of PANAR2) reports the success of the operation: 0 _B Success. 1 _B The operation has not been performed because the list of unallocated elements was empty.

Controller Area Network Controller (MultiCAN+)

Table 23-11 Panel Commands (cont'd)

PANCMD	PANAR2	PANAR1	Command Description
06 _H	Argument: Destination Object Number	Argument: Source Object Number	Static Insert Behind Remove a message object (source object) from the list that it currently belongs to, and insert it behind a given destination object into the list structure of the destination object. The source object thus becomes the successor of the destination object.
07 _H	Argument: Destination Object Number Result: Bit 7: ERR Bit 6-0: undefined	Result: Object Number of inserted object	Dynamic Insert Behind Insert a new message object behind a given destination object. The new object is taken from the list of unallocated elements (the first element is chosen). The number of the new object is delivered as result to PANAR1. An ERR bit (bit 7 of PANAR2) reports the success of the operation: 0 _B Success. 1 _B The operation has not been performed because the list of unallocated elements was empty.
08 _H - FF _H	–	–	Reserved

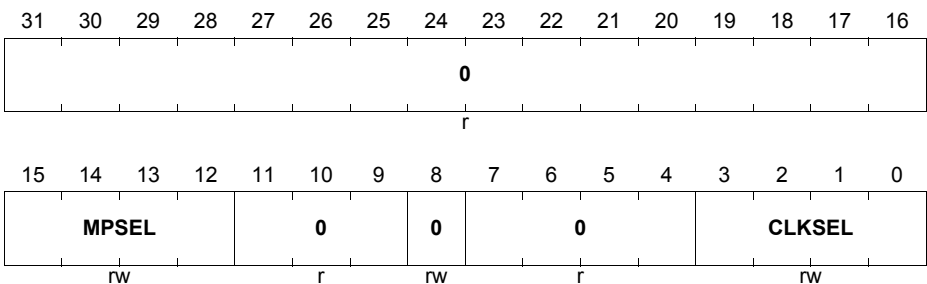
Controller Area Network Controller (MultiCAN+)

The Module Control Register MCR contains basic settings that determine the operation of the MultiCAN+ module.

The write access to the lowest byte of the MCR register is possible only if the CCE bits of all CAN nodes are set (NCRx.CCE bits). The NCRx.INIT bits will be automatically set when the lowest byte of the MCR register is written, independent of the setting of the CCE bits. The INIT bits have to be reset by software in order to activate the CAN nodes.

The reconfiguration of the clock source has to be done by using two writes: first a write of zero to the CLKSEL bit field, and then a second write defining the new clock source. Between the first and the second write a delay of $4 / f_A + 2 / f_{CAN}$ number of cycles must be inserted by software, where f_A is the frequency being switched off with the first write. Exception: in case that f_{ASYN_CAN} is selected as the baud rate logic clock (MCR.CLKSEL = 1), no delay cycles between the writes are necessary. In both cases, simply using one write defining the new clock source is not allowed.

Note: If the baud rate logic is supplied from an unstable clock source, or no clock at all, the CAN functionality is not guaranteed.

MCR
Module Control Register
(1C8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CLKSEL	[3:0]	rw	Baud Rate Logic Clock Select 0000 _B No clock supplied 0001 _B f_{ASYN_CAN} 0010 _B Oscillator Clock 0100 _B E-Ray clock (f_{ERAY}) 1000 _B hard wired to 0 ... _B not allowed
0	8	rw	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

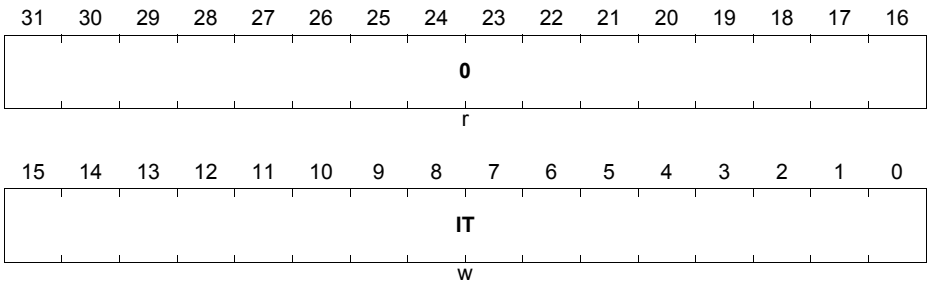
Field	Bits	Type	Description
MPSEL	[15:12]	rw	<p>Message Pending Selector</p> <p>Bit field MPSEL makes it possible to select the bit position of the message pending bit after a message reception/transmission by a mixture of the MOIPRn register bit fields RXINP, TXINP, and MPN. Selection details are given in Figure 23-20 on Page 23-48.</p>
0	[31:16], [11:9],[7:4]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Controller Area Network Controller (MultiCAN+)

The Interrupt Trigger Register ITR is used to trigger interrupt requests on each interrupt output line by software.

MITR

Module Interrupt Trigger Register (1CC_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
IT	[15:0]	w	Interrupt Trigger Writing a 1 to IT[m] (m = 0-15) generates an interrupt request on interrupt output line INT_O[m]. Writing a 0 to IT[m] has no effect. Bit field IT is always read as 0. Multiple interrupt requests can be generated with a single write operation to MITR by writing a 1 to several bit positions of IT.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

List Pointer and List Register

Each CAN node has a list that determines the allocated message objects. Additionally, a list of all unallocated objects is available. Furthermore, general purpose lists are available which are not associated to a CAN node. The List Registers are assigned in the following way:

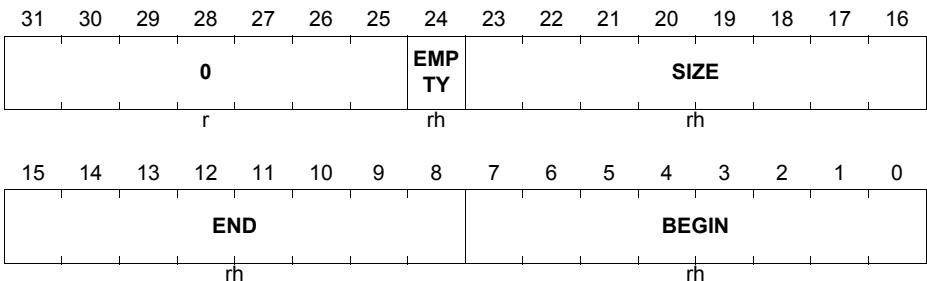
- LIST0 provides the list of all unallocated objects
- LIST1 provides the list for CAN node 0
- LIST2 provides the list for CAN node 1
- ...
- LIST $\underline{5}$ provides the list for CAN node $\underline{4}$

LIST0

List Register 0 (100_H) **Reset Value: 00FF FF00_H**

LISTn (n = 1-15)

List Register n (100_H+n*4_H) **Reset Value: 0100 0000_H**



Field	Bits	Type	Description
BEGIN	[7:0]	rh	List Begin BEGIN indicates the number of the first message object in list i.
END	[15:8]	rh	List End END indicates the number of the last message object in list i.
SIZE	[23:16]	rh	List Size SIZE indicates the number of elements in the list i. SIZE = number of list elements - 1 SIZE = 0 indicates that list i is empty.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
EMPTY	24	rh	List Empty Indication 0 _B At least one message object is allocated to list i. 1 _B No message object is allocated to the list i. List i is empty.
0	[31:25]	r	Reserved Read as 0.

Controller Area Network Controller (MultiCAN+)

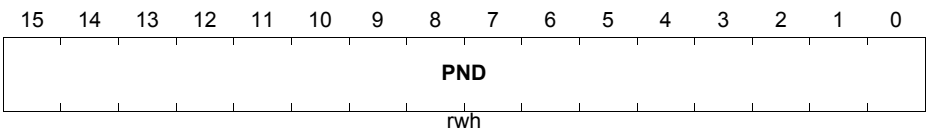
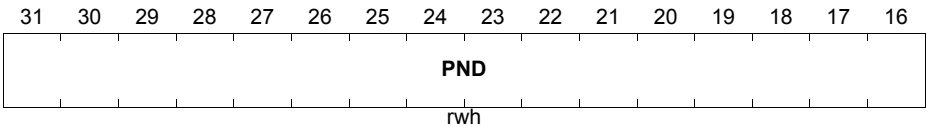
Message Notifications

When a message object n generates an interrupt request upon the transmission or reception of a message, then the request is routed to the interrupt output line selected by the bit field MOIPRn.TXINP or MOIPRn.RXINP of the message object n. As there are more message objects than interrupt output lines, an interrupt routine typically processes requests from more than one message object. Therefore, a priority selection mechanism is implemented in the MultiCAN+ module to select the highest priority object within a collection of message objects.

The Message Pending Register MSPNDk contains the pending interrupt notification of list i.

MSPNDk (k = 0-7)

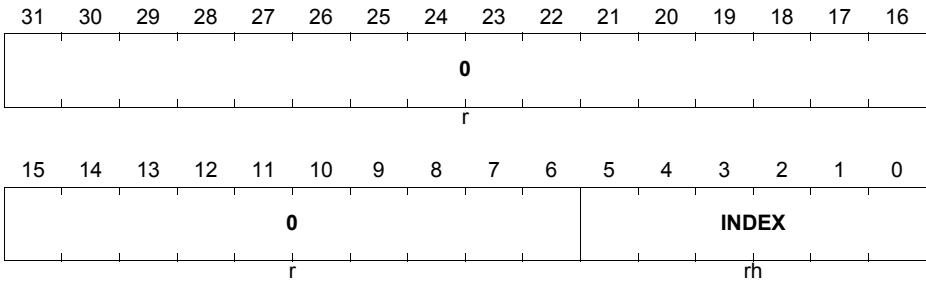
Message Pending Register k **(140_H+k*4_H)** **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
PND	[31:0]	rwh	Message Pending When a message interrupt occurs, the message object sets a bit in one of the MSPND register, where the bit position is given by the MPN[4:0] field of the IPR register of the message object. The register selection n is given by the higher bits of MPN. The register bits can be cleared by software (write 0). Writing a 1 has no effect.

Controller Area Network Controller (MultiCAN+)

Each Message Pending Register has a Message Index Register MSIDk associated with it. The Message Index Register shows the active (set) pending bit with lowest bit position within groups of pending bits.

MSIDk (k = 0-7)
Message Index Register k **(180_H+k*4_H)** **Reset Value: 0000 0020_H**


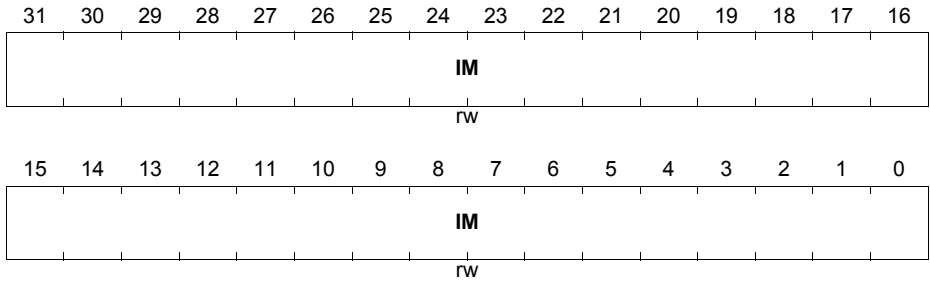
Field	Bits	Type	Description
INDEX	[5:0]	rh	Message Pending Index The value of INDEX is given by the bit position i of the pending bit of MSPNDk with the following properties: <ol style="list-style-type: none"> MSPNDk[i] & IM[i] = 1 i = 0 or MSPNDk[i-1:0] & IM[i-1:0] = 0 If no bit of MSPNDk satisfies these conditions then INDEX reads 100000 _B . Thus INDEX shows the position of the first pending bit of MSPNDk, in which only those bits of MSPNDk that are selected in the Message Index Mask Register are taken into account.
0	[31:6]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

The Message Index Mask Register MSIMASK selects individual bits for the calculation of the Message Pending Index. The Message Index Mask Register is used commonly for all Message Pending registers and their associated Message Index registers.

MSIMASK

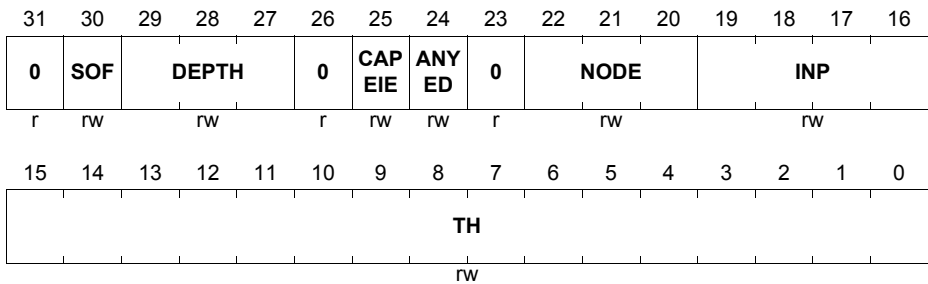
Message Index Mask Register (1C0_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
IM	[31:0]	rw	Message Index Mask Only those bits in MSPNDk for which the corresponding Index Mask bits are set contribute to the calculation of the Message Index.

Controller Area Network Controller (MultiCAN+)

The Measurement Control Register MECR controls the CAN edge timing measurement function for calibration purposes.

MECR
Measurement Control Register (1D0_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
TH	[15:0]	rw	Threshold This bit field contains the threshold value for the measurement timer. If TH = 0000 _H , the timer is stopped and the capture function is disabled.
INP	[19:16]	rw	Interrupt Node Pointer INP selects the interrupt output line INT_O _m (m = 0-15) for a capture event interrupt. 0000 _B Interrupt output line INT_O0 is selected. 0001 _B Interrupt output line INT_O1 is selected. ... _B ... 1110 _B Interrupt output line INT_O14 is selected. 1111 _B Interrupt output line INT_O15 is selected.
NODE	[22:20]	rw	Node This bit field determines the CAN node x whose input line RXDCAN _x is used for start and capture of the measurement timer. 000 _B Node 0 001 _B Node 1 ... _B ... 100 _B Node 4

Controller Area Network Controller (MultiCAN+)

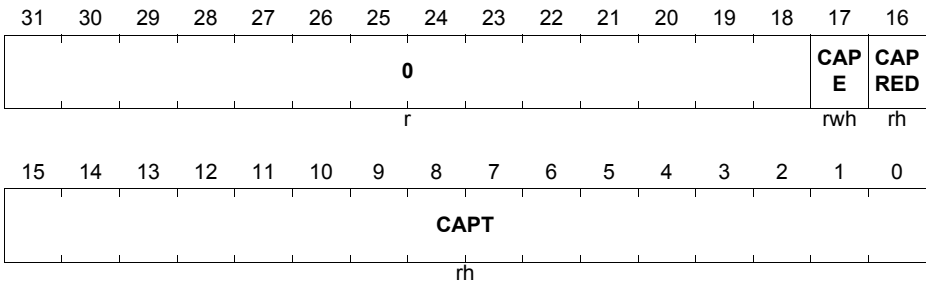
Field	Bits	Type	Description
ANYED	24	rw	Any Edge This bit enables capture on any edge of CAN input line specified by NODE. 0_B Capture on falling (dominant) edge only 1_B Capture on rising (recessive) or falling (dominant) edge
CAPEIE	25	rw	Capture Event Interrupt Enable This bit enables the capture event interrupt. 0_B Capture event interrupt is disabled 1_B Capture event interrupt is enabled Bit field INP selects the interrupt output line which becomes activated at this type of interrupt.
DEPTH	[29:27]	rw	Digital Glitch Filter Depth DEPTH determines the number of input samples clocked with f_{CLC} that are taken into account for the calculation of the floating average. The higher DEPTH is chosen to be, the longer the glitches that are suppressed and the longer the delay of the input signal introduced by this filter. 000_B off, default 001_B Filter depth of 8 cycles 010_B Filter depth of 16 cycles 011_B Filter depth of 32 cycles 100_B Filter depth of 64 cycles 101_B Filter depth of 128 cycles 110_B Filter depth of 255 cycles 111_B not allowed, reserved
SOF	30	rw	Start Of Frame This bit selects falling edge or any edge as measurement for start of frame detection. 0_B Measurement starts with any falling edge 1_B Measurement starts with falling Start of Frame edge. i.e any falling edge that occurs while the CAN node is in idle state
0	23, 26, 31	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

The Measurement Status Register MESTAT contains the status information of the CAN edge timing measurement.

MESTAT

Measurement Status Register (1D4_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
CAPT	[15:0]	rh	<p>Captured Timer</p> <p>This bit field contains the captured measurement timer content.</p> <p>The timer itself is cleared and started by the first falling (dominant) edge of a CAN frame on the input line of the CAN node specified by MECR.NODE. The timer is incremented by the module control clock f_{CLC} and will be stopped when FFFF_H is reached. If MECR.TH = 0000_H, the timer is always stopped.</p> <p>A capture will take place if all the following conditions are met:</p> <ol style="list-style-type: none"> MECR.TH > 0000_H Timer is cleared and started by new frame Timer reaches MECR.TH This node is not sending and first edge (as specified by MECR.ANYED) after 3. occurs on input line <p>Capture will be repeated for the following CAN frames until MECR.TH is cleared.</p>
CAPRED	16	rh	<p>Captured Rising Edge</p> <p>This bit indicates the type of edge that caused the last capture event.</p> <p>0_B Capture occurred on falling (dominant) edge</p> <p>1_B Capture occurred on rising (recessive) edge</p>

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
CAPE	17	rwh	Capture Event This flag is set on a capture event. It must be reset by software. 0_B No capture event has occurred since last flag reset 1_B Capture event has occurred since last flag reset An interrupt request is generated if MECR.CAPEIE = 1. If CAPE=1 then no further measurement results are posted to MESTAT.CAPT and MESTAT.CAPRED. CAPE bit has to be cleared to re-enable update of MESTAT.CAPT and MESTAT.CAPRED.
0	[31:18]	r	Reserved Read as 0; should be written with 0.

23.6.2 CAN Node Registers

The CAN node registers are built in for each CAN node of the MultiCAN+ module. They contain information that is directly related to the operation of the CAN nodes and are shared among the nodes.

The Node Control Register contains basic settings that determine the operation of the CAN node.

CAN_NCRx (x = 0-4)
Node x Control Register ($200_H + x * 100_H$) **Reset Value: 0000 0041_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						FD EN	SUS EN	CAL M	CCE	TXDI S	CAN DIS	ALIE	LECI E	TRIE	INIT
r						rw	rw	rw	rw	rw	rw	rw	rw	rw	rwh

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
INIT	0	rwh	<p>Node Initialization</p> <p>0_B Resetting bit INIT enables the participation of the node in the CAN traffic. If the CAN node is in the bus-off state, the ongoing bus-off recovery (which does not depend on the INIT bit) is continued. With the end of the bus-off recovery sequence the CAN node is allowed to take part in the CAN traffic. If the CAN node is not in the bus-off state, a sequence of 11 consecutive recessive bits must be detected before the node is allowed to take part in the CAN traffic.</p> <p>1_B Setting this bit terminates the participation of this node in the CAN traffic. Any ongoing frame transfer is cancelled and the transmit line goes recessive. If the CAN node is in the bus-off state, then the running bus-off recovery sequence is continued. If the INIT bit is still set after the successful completion of the bus-off recovery sequence, i.e. after detecting 128 sequences of 11 consecutive recessive bits (11 × 1), then the CAN node leaves the bus-off state but remains inactive as long as INIT remains set.</p> <p>Bit INIT is automatically set when the CAN node enters the bus-off state (see Page 23-30).</p>
TRIE	1	rw	<p>Transfer Interrupt Enable</p> <p>TRIE enables the transfer interrupt of CAN node x. This interrupt is generated after the successful reception or transmission of a CAN frame in node x.</p> <p>0_B Transfer interrupt is disabled. 1_B Transfer interrupt is enabled.</p> <p>Bit field NIPRx.TRINP selects the interrupt output line which becomes activated at this type of interrupt.</p>

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
LECIE	2	rw	<p>LEC Indicated Error Interrupt Enable</p> <p>LECIE enables the last error code interrupt of CAN node x. This interrupt is generated with each hardware update of bit field NSRx.LEC with LEC > 0 (CAN protocol error).</p> <p>0_B Last error code interrupt is disabled. 1_B Last error code interrupt is enabled.</p> <p>Bit field NIPRx.LECINP selects the interrupt output line which becomes activated at this type of interrupt.</p>
ALIE	3	rw	<p>Alert Interrupt Enable</p> <p>ALIE enables the alert interrupt of CAN node x. This interrupt is generated by any one of the following events:</p> <ul style="list-style-type: none"> • A change of bit NSRx.BOFF • A change of bit NSRx.EWRN • A List Length Error, which also sets bit NSRx.LLE • A List Object Error, which also sets bit NSRx.LOE <p>0_B Alert interrupt is disabled. 1_B Alert interrupt is enabled.</p> <p>Bit field NIPRx.ALINP selects the interrupt output line which becomes activated at this type of interrupt.</p>
CANDIS	4	rw	<p>CAN Disable</p> <p>Setting this bit disables the CAN node. The CAN node first waits until it is bus-idle or bus-off. Then bit NSRx.SUSACK and NCRx.INIT is automatically set, and an alert interrupt is generated if bit ALIE is set.</p>
TXDIS	5	rw	<p>Transmit Disable</p> <p>Setting this bit disables the transmission on CAN node x as soon as bus-idle is reached. Reception and bits in MOSTATn, e.g. TXRQ, will not be influenced.</p>
CCE	6	rw	<p>Configuration Change Enable</p> <p>0_B The Bit Timing Register, the Port Control Register, Error Counter Register and NCRx.FDEN bit may only be read. All attempts to modify them are ignored.</p> <p>1_B The Bit Timing Register, the Port Control Register, Error Counter Register and NCRx.FDEN bit may be read and written.</p>

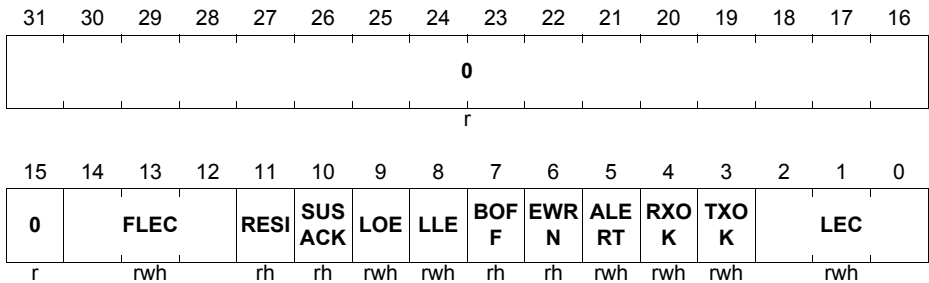
Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
CALM	7	rw	<p>CAN Analyzer Mode</p> <p>If this bit is set, then the CAN node operates in Analyzer Mode. This means that messages may be received, but not transmitted. No acknowledge is sent on the CAN bus upon frame reception. Active-error flags are sent recessive instead of dominant. The transmit line is continuously held at recessive (1) level.</p> <p>Bit CALM can be written only while bit INIT is set.</p>
SUSEN	8	rw	<p>Suspend Enable</p> <p>This bit makes it possible to set the CAN node into Suspend Mode via OCDS (on chip debug support):</p> <p>0_B An OCDS suspend trigger is ignored by the CAN node.</p> <p>1_B An OCDS suspend trigger disables the CAN node: As soon as the CAN node becomes bus-idle or bus-off, bit INIT is internally forced to 1 to disable the CAN node. The actual value of bit INIT remains unchanged.</p> <p>Bit SUSEN is reset via OCDS Reset.</p>
FDEN	9	rw	<p>CAN Flexible Data-Rate Enable</p> <p>This bit enables the CAN FD feature:</p> <p>0_B CAN FD disabled. Message transfer for CAN frames using classical CAN Format.</p> <p>1_B CAN FD enabled. Message transfer for CAN frames using CAN FD Format¹⁾.</p> <p>Bit FDEN is protected by CCE and INIT bit. Please see Section 23.4.12.4 and Table 23-18</p>
0	[31:10]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

1) Message transmission in Classical CAN, Long Frame or Long + Fast CAN FD Frames. Message Reception according to CAN FD Protocol Specification, (i.e Able to Receive Classical CAN Format Frames ISO11898-1 Long Frame and Long + Fast CAN FD Frames)

Controller Area Network Controller (MultiCAN+)

The Node Status Register NSRx reports errors as well as successfully transferred CAN frames.

CAN_NSRx (x = 0-4)
Node x Status Register
(204_H+x*100_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
LEC	[2:0]	rwh	Last Error Code This bit field indicates the type of the last (most recent) CAN error. The encoding of this bit field is described in Table 23-12 .
TXOK	3	rwh	Message Transmitted Successfully 0 _B No successful transmission since last (most recent) flag reset. 1 _B A message has been transmitted successfully (error-free and acknowledged by at least another node). TXOK must be reset by software (write 0). Writing 1 has no effect.
RXOK	4	rwh	Message Received Successfully 0 _B No successful reception since last (most recent) flag reset. 1 _B A message has been received successfully. RXOK must be reset by software (write 0). Writing 1 has no effect.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
ALERT	5	rwh	Alert Warning The ALERT bit is set upon the occurrence of one of the following events (the same events which also trigger an alert interrupt if ALIE is set): <ul style="list-style-type: none"> • A change of bit NSRx.BOFF • A change of bit NSRx.EWRN • A List Length Error, which also sets bit NSRx.LLE • A List Object Error, which also sets bit NSRx.LOE ALERT must be reset by software (write 0). Writing 1 has no effect.
EWRN	6	rh	Error Warning Status 0 _B No warning limit exceeded. 1 _B One of the error counters REC or TEC reached the warning limit EWRNLVL.
BOFF	7	rh	Bus-off Status 0 _B CAN controller is not in the bus-off state. 1 _B CAN controller is in the bus-off state.
LLE	8	rwh	List Length Error 0 _B No List Length Error since last (most recent) flag reset. 1 _B A List Length Error has been detected during message acceptance filtering. The number of elements in the list that belongs to this CAN node differs from the list SIZE given in the list termination pointer. LLE must be reset by software (write 0). Writing 1 has no effect.
LOE	9	rwh	List Object Error 0 _B No List Object Error since last (most recent) flag reset. 1 _B A List Object Error has been detected during message acceptance filtering. A message object with wrong LIST index entry in the Message Object Status Register has been detected. LOE must be reset by software (write 0). Writing 1 has no effect.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
SUSACK	10	rh	Suspend Acknowledge 0 _B The CAN node is not in Suspend Mode or a suspend request is pending, but the CAN node has not yet reached bus-idle or bus-off. 1 _B The CAN node is in Suspend Mode: The CAN node is inactive (bit NCR.INIT internally forced to 1) due to an OCDS suspend request.
RESI	11	rh	Received Error State Indicator Flag This bit is an error flag that is set when the ESI flag in a received CAN FD frame is set. 0 _B Last received CAN FD message did not have its ESI flag set. 1 _B Last received CAN FD message had its ESI flag set.
FLEC	[14:12]	rwh	Fast Last Error Code This bit field indicates the type of the last (most recent) CAN error that occurred in the data phase of a CAN FD format frame with its BRS flag set. Coding is the same as for LEC. The encoding of this bit field is described in Table 23-12 . This field will be cleared to zero when a CAN FD frame with its BRS flag set has been transferred (reception or transmission) without error.
0	[31:15]	r	Reserved Read as 0; should be written with 0.

Encoding of the LEC Bit field
Table 23-12 Encoding of the LEC Bit field

LEC Value	Signification
000 _B	No Error: No error was detected for the last (most recent) message on the CAN bus.
001 _B	Stuff Error: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.
010 _B	Form Error: A fixed format part of a received frame has the wrong format.

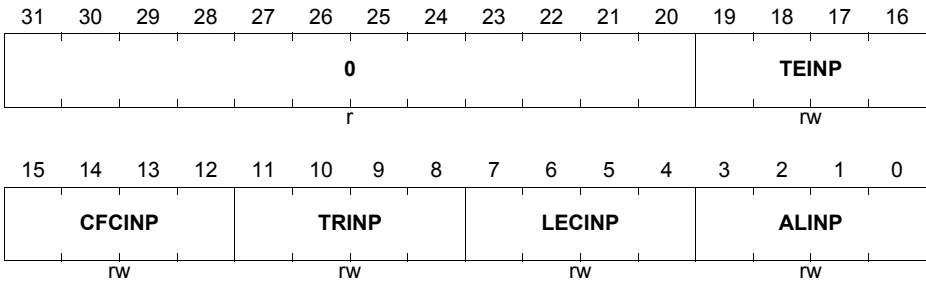
Controller Area Network Controller (MultiCAN+)
Table 23-12 Encoding of the LEC Bit field (cont'd)

LEC Value	Signification
011 _B	Ack Error: The transmitted message was not acknowledged by another node.
100 _B	Bit1 Error: During a message transmission, the CAN node tried to send a recessive level (1) outside the arbitration field and the acknowledge slot, but the monitored bus value was dominant.
101 _B	Bit0 Error: Two different conditions are signaled by this code: <ol style="list-style-type: none"> 1. During transmission of a message (or acknowledge bit, active-error flag, overload flag), the CAN node tried to send a dominant level (0), but the monitored bus value was recessive. 2. During bus-off recovery, this code is set each time a sequence of 11 recessive bits has been monitored. The CPU may use this code as indication that the bus is not continuously disturbed.
110 _B	CRC Error: The CRC checksum of the received message was incorrect.
111 _B	CPU write to LEC: Whenever the CPU writes the value 111 to LEC, it takes the value 111. Whenever the CPU writes another value to LEC, the written LEC value is ignored.

Note: When a frame in CAN FD format reached data phase with BRS flag set, the next CAN event (error/valid frame) will be shown in FLEC instead of LEC. An error in a fixed stuff bit of CAN FD CRC will be shown as Form and not Stuff Error. Correspondingly CAN event (error/valid frame) will be shown back at LEC after the first bit of the CRC delimiter when CAN FD switches from Data bit rate to Nominal bit rate.

Controller Area Network Controller (MultiCAN+)

The five interrupt pointers in the Node Interrupt Pointer Register NIPRx select one out of the sixteen interrupt outputs individually for each type of CAN node interrupt. See also [Page 23-34](#) for more CAN node interrupt details.

CAN_NIPRx (x = 0-4)
Node x Interrupt Pointer Register (208_H+x*100_H)
Reset Value: 0000 0000_H


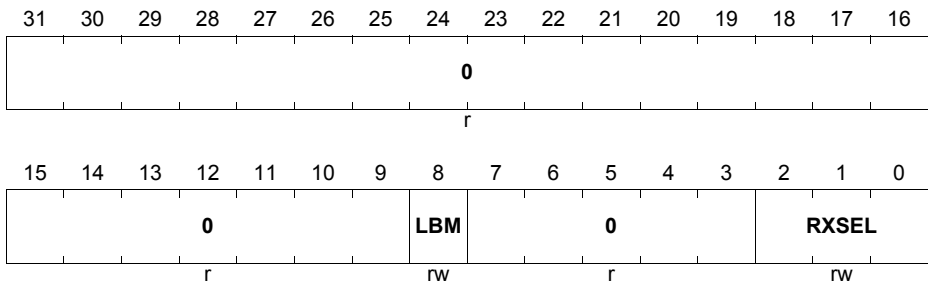
Field	Bits	Type	Description
ALINP	[3:0]	rw	Alert Interrupt Node Pointer ALINP selects the interrupt output line INT_0m (m = 0-15) for an alert interrupt of CAN Node x. 0000 _B Interrupt output line INT_00 is selected. 0001 _B Interrupt output line INT_01 is selected. ... _B ... 1110 _B Interrupt output line INT_014 is selected. 1111 _B Interrupt output line INT_015 is selected.
LECINP	[7:4]	rw	Last Error Code Interrupt Node Pointer LECINP selects the interrupt output line INT_0m (m = 0-15) for an LEC interrupt of CAN Node x. 0000 _B Interrupt output line INT_00 is selected. 0001 _B Interrupt output line INT_01 is selected. ... _B ... 1110 _B Interrupt output line INT_014 is selected. 1111 _B Interrupt output line INT_015 is selected.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
TRINP	[11:8]	rw	Transfer OK Interrupt Node Pointer TRINP selects the interrupt output line INT_Om (m = 0-15) for a transfer OK interrupt of CAN Node x. 0000 _B Interrupt output line INT_O0 is selected. 0001 _B Interrupt output line INT_O1 is selected. ... _B ... 1110 _B Interrupt output line INT_O14 is selected. 1111 _B Interrupt output line INT_O15 is selected.
CFCINP	[15:12]	rw	Frame Counter Interrupt Node Pointer CFCINP selects the interrupt output line INT_Om (m = 0-15) for a frame counter overflow interrupt of CAN Node x. 0000 _B Interrupt output line INT_O0 is selected. 0001 _B Interrupt output line INT_O1 is selected. ... _B ... 1110 _B Interrupt output line INT_O14 is selected. 1111 _B Interrupt output line INT_O15 is selected.
TEINP	[19:16]	rw	Timer Event Interrupt Node Pointer TEINP selects the interrupt output line INT_Om (m = 0-15) for a timer event interrupt of CAN Node x. 0000 _B Interrupt output line INT_O0 is selected. 0001 _B Interrupt output line INT_O1 is selected. ... _B ... 1110 _B Interrupt output line INT_O14 is selected. 1111 _B Interrupt output line INT_O15 is selected.
0	[31:20]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

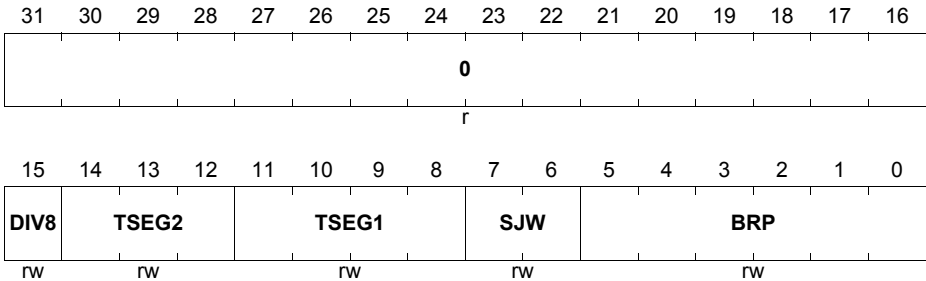
The Node Port Control Register NPCRx configures the CAN bus transmit/receive ports. NPCRx can be written only if bit NCRx.CCE is set.

CAN_NPCRx (x = 0-4)
Node x Port Control Register (20C_H+x*100_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
RXSEL	[2:0]	rw	Receive Select RXSEL selects one out of 8 possible receive inputs. The CAN receive signal is performed only through the selected input. <i>Note: In TC26x B-Step, only specific combinations of RXSEL are available (see also “Node Receive Input Selection” on Page 23-166 for description and the page before for RXSEL selections).</i>
LBM	8	rw	Loop-Back Mode 0 _B Loop-Back Mode is disabled. 1 _B Loop-Back Mode is enabled. This node is connected to an internal (virtual) loop-back CAN bus. All CAN nodes which are in Loop-Back Mode are connected to this virtual CAN bus so that they can communicate with each other internally. The external transmit line is forced recessive in Loop-Back Mode.
0	[7:3], [31:9]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

The Node Bit Timing Register NBTRx contains all parameters to set up the bit timing for the CAN transfer. NBTRx can be written only if bit NCRx.CCE is set. Please note that NBTRx is a register with two views, depending on NCRx.FDEN settings different view applies. (i.e When NCRx.FDEN=0, CAN_NBTRx view applies and when NCRx.FDEN=1, CAN_NBTEVRx applies)

CAN_NBTRx (x = 0-4)
Node x Bit Timing Register $(210_H + x * 100_H)$ **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
BRP	[5:0]	rw	Baud Rate Prescaler The duration of one time quantum is given by (BRP + 1) clock cycles if DIV8 = 0. The duration of one time quantum is given by 8 × (BRP + 1) clock cycles if DIV8 = 1.
SJW	[7:6]	rw	(Re) Synchronization Jump Width (SJW + 1) time quanta are allowed for re-synchronization.
TSEG1	[11:8]	rw	Time Segment Before Sample Point (TSEG1 + 1) time quanta is the user-defined nominal time between the end of the synchronization segment and the sample point. It includes the propagation segment, which takes into account signal propagation delays. The time segment may be lengthened due to re-synchronization. Valid values for TSEG1 are 2 to 15.

Controller Area Network Controller (MultiCAN+)

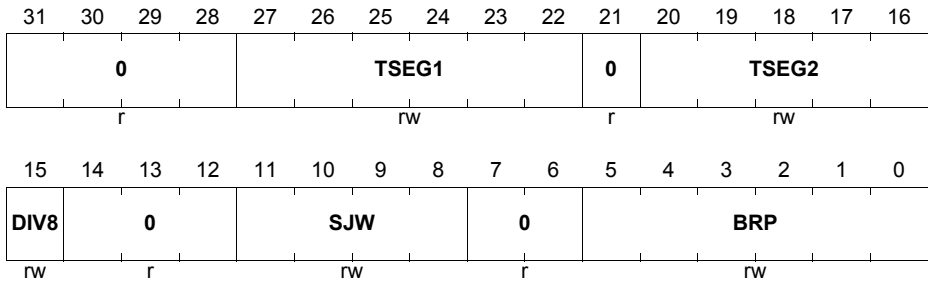
Field	Bits	Type	Description
TSEG2	[14:12]	rw	Time Segment After Sample Point (TSEG2 + 1) time quanta is the user-defined nominal time between the sample point and the start of the next synchronization segment. It may be shortened due to re-synchronization. Valid values for TSEG2 are 1 to 7.
DIV8	15	rw	Divide Prescaler Clock by 8 0 _B A time quantum lasts (BRP+1) clock cycles. 1 _B A time quantum lasts 8 × (BRP+1) clock cycles.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

The Node Bit Timing Extended View Register NBTEVRx is applicable only when NCRx.FDEN=1 (CAN FD enabled). NBTEVRx contains all parameters to set up CAN bit timing for Nominal Bit Rate. NBTRx register can be written only if bit NCRx.CCE is set.

CAN_NBTEVRx (x = 0-4)

Node x Bit Timing Extended View Register(210_H+x*100_H) Reset Value: 0000 0000_H



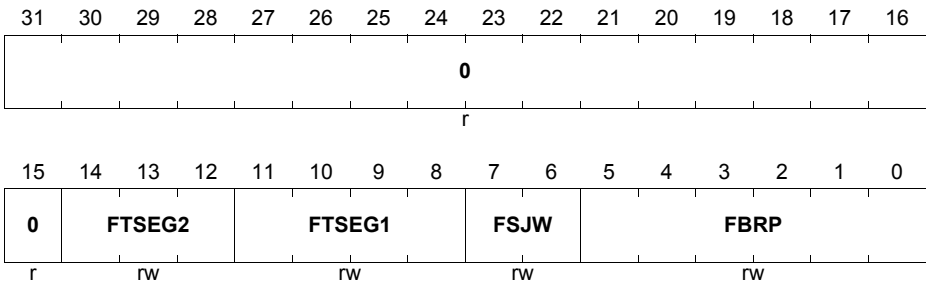
Field	Bits	Type	Description
BRP	[5:0]	rw	Baud Rate Prescaler The duration of one time quantum is given by (BRP + 1) clock cycles if DIV8 = 0. The duration of one time quantum is given by 8 × (BRP + 1) clock cycles if DIV8 = 1.
SJW	[11:8]	rw	(Re) Synchronization Jump Width (SJW + 1) time quanta are allowed for re-synchronization.
DIV8	15	rw	Divide Prescaler Clock by 8 0 _B A time quantum lasts (BRP+1) clock cycles. 1 _B A time quantum lasts 8 × (BRP+1) clock cycles.
TSEG2	[20:16]	rw	Time Segment After Sample Point (TSEG2 + 1) time quanta is the user-defined nominal time between the sample point and the start of the next synchronization segment. It may be shortened due to re-synchronization. Valid values for TSEG2 are 1 to 31.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
TSEG1	[27:22]	rw	Time Segment Before Sample Point (TSEG1 + 1) time quanta is the user-defined nominal time between the end of the synchronization segment and the sample point. It includes the propagation segment, which takes into account signal propagation delays. The time segment may be lengthened due to re-synchronization. Valid values for TSEG1 are 2 to 63.
0	[7:6], [14:12], 21, [31:28]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

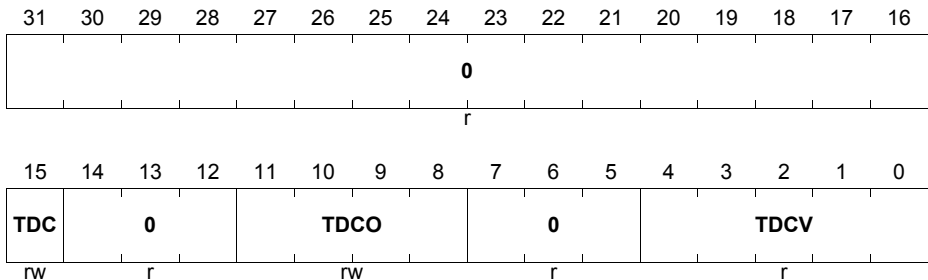
The Fast Node Bit Timing Register, FNBTRx contains all parameters to set up CAN bit timing for Data Bit Rate. FNBTRx can be written only if bit NCRx.CCE is set.

CAN_FNBTRx (x = 0-4)
Fast Node x Bit Timing Register (238_H+x*100_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
FBRP	[5:0]	rw	Fast Baud Rate Prescaler The duration of one time quantum is given by (BRP + 1) clock cycles.
FSJW	[7:6]	rw	Fast (Re) Synchronization Jump Width (SJW + 1) time quanta are allowed for re-synchronization.
FTSEG1	[11:8]	rw	Fast Time Segment Before Sample Point (TSEG1 + 1) time quanta is the user-defined nominal time between the end of the synchronization segment and the sample point. It includes the propagation segment, which takes into account signal propagation delays. The time segment may be lengthened due to re-synchronization.
FTSEG2	[14:12]	rw	Fast Time Segment After Sample Point (TSEG2 + 1) time quanta is the user-defined nominal time between the sample point and the start of the next synchronization segment. It may be shortened due to re-synchronization.
0	[31:15]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

The Transmitter Delay Compensation Register, TDCRx contains all parameters to setup the Transmitter Delay Compensation Feature and the corresponding status bits. NTDCRx register can be written only if bit NCRx.CCE is set.

CAN_NTDCRx (x = 0-4)
Node x Transmitter Delay Compensation Register
 $(23C_H + x * 100_H)$
Reset Value: 0000 0000_H


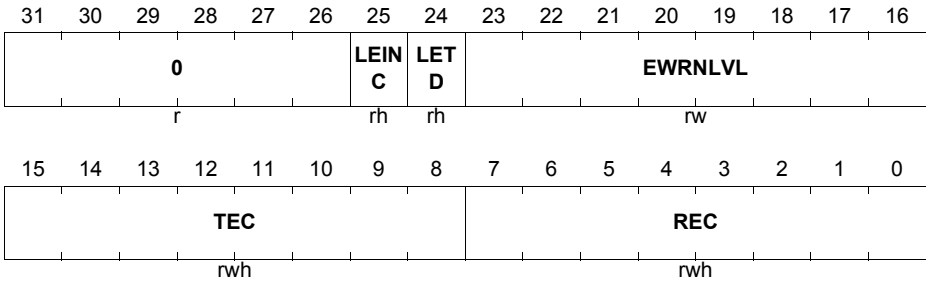
Field	Bits	Type	Description
TDCV	[4:0]	r	Transmitter Delay Compensation Value This bit field shows the secondary sample point which is the sum of the measured Transmitter Delay (from CAN Transmit to Receive) and Transmitter Delay compensation offset, NTDCRz.TDCO. Valid values for TDCV are 0 to 31 times of time quanta t_Q .
TDCO	[11:8]	rw	Transmitter Delay Compensation Offset This bit field defines the Transmitter Delay compensation offset that is added to the measured Transmitter Delay (from CAN Transmit to Receive) which forms the secondary sample point. Valid values for TDCO are 0 to 15 times of time quanta t_Q .
TDC	15	rw	Transmitter Delay Compensation Enable This bit enables the Transmitter Delay Compensation feature: 0 _B Transmitter Delay Compensation disabled. 1 _B Transmitter Delay Compensation enabled.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
0	[7:5], [14:12], [31:16]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

The Node Error Counter Register NECNTx contains the CAN receive and transmit error counter as well as some additional bits to ease error analysis. NECNTx can be written only if bit NCRx.CCE is set.

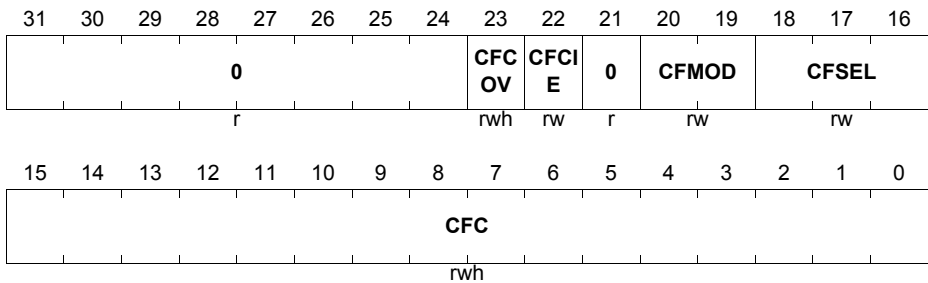
CAN_NECNTx (x = 0-4)
Node x Error Counter Register (214_H+x*100_H) **Reset Value: 0060 0000_H**


Field	Bits	Type	Description
REC	[7:0]	rwh	Receive Error Counter Bit field REC contains the value of the receive error counter of CAN node x.
TEC	[15:8]	rwh	Transmit Error Counter Bit field TEC contains the value of the transmit error counter of CAN node x.
EWRNLVL	[23:16]	rw	Error Warning Level Bit field EWRNLVL determines the threshold value (warning level, default 96) to be reached in order to set the corresponding error warning bit EWRN.
LETD	24	rh	Last Error Transfer Direction 0 _B The last error occurred while the CAN node x was receiver (REC has been incremented). 1 _B The last error occurred while the CAN node x was transmitter (TEC has been incremented).
LEINC	25	rh	Last Error Increment 0 _B The last error led to an error counter increment of 1. 1 _B The last error led to an error counter increment of 8.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
0	[31:26]	r	Reserved Read as 0; should be written with 0.

The Node Frame Counter Register NFCRx contains the actual value of the frame counter as well as control and status bits of the frame counter.

CAN_NFCRx (x = 0-4)
Node x Frame Counter Register (218_H+x*100_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
CFC	[15:0]	rwh	CAN Frame Counter In Frame Count Mode (CFMOD = 00 _B), this bit field contains the frame count value. In Time Stamp Mode (CFMOD = 01 _B), this bit field contains the captured bit time count value, captured with the start of a new frame. In all Bit Timing Analysis Modes ¹⁾ (CFMOD = 10 _B), CFC always displays the number of f_{CAN} clock cycles (measurement result) minus 1. Example: a CFC value of 34 in measurement mode CFSEL = 000 _B means that 35 f_{CAN} clock cycles have been elapsed between the most recent two dominant edges on the receive input. In Error Count Mode (CFMOD = 11 _B), this bit field contains the total amount of error frames received or error detected by the node.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
CFSEL	[18:16]	rw	<p>CAN Frame Count Selection</p> <p>This bit field selects the function of the frame counter for the chosen frame count mode.</p> <p>Frame Count Mode</p> <p>Bit 0 If Bit 0 of CFSEL is set, then CFC is incremented each time a foreign frame (i.e. a frame not matching to a message object) has been received on the CAN bus.</p> <p>Bit 1 If Bit 1 of CFSEL is set, then CFC is incremented each time a frame matching to a message object has been received on the CAN bus.</p> <p>Bit 2 If Bit 2 of CFSEL is set, then CFC is incremented each time a frame has been transmitted successfully by the node.</p> <p>Time Stamp Mode</p> <p>The frame counter is incremented (internally) at the beginning of a new bit time. The value is sampled during the SOF bit of a new frame. The sampled value is visible in the CFC field.</p> <p>Bit Timing Mode</p> <p>The available bit timing measurement modes are shown in Table 23-13. If CFCIE is set, then an interrupt on request node x (where x is the CAN node number) is generated with a CFC update.</p> <p>Error Count Mode</p> <p>The frame counter is incremented when an error frame is received or an error is detected by the node. (001_B to 110_B) (see Table 23-12 for Encoding of the LEC Bit field).</p>
CFMOD	[20:19]	rw	<p>CAN Frame Counter Mode</p> <p>This bit field determines the operation mode of the frame counter.</p> <p>00_B Frame Count Mode: The frame counter is incremented upon the reception and transmission of frames.</p> <p>01_B Time Stamp Mode: The frame counter is used to count bit times.</p> <p>10_B Bit Timing Mode: The frame counter is used for analysis of the bit timing.</p> <p>11_B Error Count Mode: The frame counter is used for counting when an error frame is received or an error is detected by the node.</p>

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
CFCIE	22	rw	CAN Frame Count Interrupt Enable CFCIE enables the CAN frame counter overflow interrupt of CAN node x. 0 _B CAN frame counter overflow interrupt is disabled. 1 _B CAN frame counter overflow interrupt is enabled. Bit field NIPRx.CFCINP selects the interrupt output line that is activated at this type of interrupt.
CFCOV	23	rwh	CAN Frame Counter Overflow Flag Flag CFCOV is set upon a frame counter overflow (transition from FFFF _H to 0000 _H). In bit timing analysis mode, CFCOV is set upon an update of CFC. An interrupt request is generated if CFCIE = 1. 0 _B No overflow has occurred since last flag reset. 1 _B An overflow has occurred since last flag reset. CFCOV must be reset by software.
0	21, [31:24]	r	Reserved Read as 0; should be written with 0.

1) The value of NFCRx.CFC is valid one module cycle later when NFCRx.CFCOV is set.

Bit Timing Analysis Modes
Table 23-13 Bit Timing Analysis Modes (CFMOD = 10)

CFSEL	Measurement
000 _B	Whenever a dominant edge (transition from 1 to 0) is monitored on the receive input, the time (measured in clock cycles) between this edge and the most recent dominant edge is stored in CFC.
001 _B	Whenever a recessive edge (transition from 0 to 1) is monitored on the receive input the time (measured in clock cycles) between this edge and the most recent dominant edge is stored in CFC.
010 _B	Whenever a dominant edge is received as a result of a transmitted dominant edge, the time (clock cycles) between both edges is stored in CFC.
011 _B	Whenever a recessive edge is received as a result of a transmitted recessive edge, the time (clock cycles) between both edges is stored in CFC.
100 _B	Whenever a dominant edge that qualifies for synchronization is monitored on the receive input, the time (measured in clock cycles) between this edge and the most recent sample point is stored in CFC.

Controller Area Network Controller (MultiCAN+)
Table 23-13 Bit Timing Analysis Modes (CFMOD = 10) (cont'd)

CFSEL	Measurement
101 _B	<p>With each sample point, the time (measured in clock cycles) between the start of the new bit time and the start of the previous bit time is stored in CFC[11:0].</p> <p>Additional information is written to CFC[15:12] at each sample point: CFC[15]: Transmit value of actual bit time CFC[14]: Receive sample value of actual bit time CFC[13:12]: CAN bus information (see Table 23-14)</p>
110 _B	Reserved, do not use this combination.
111 _B	Reserved, do not use this combination.

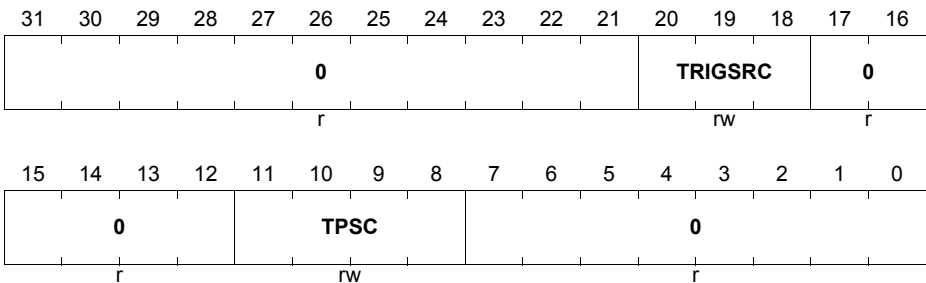
Table 23-14 CAN Bus State Information

CFC[13:12]	CAN Bus State
00 _B	<p>NoBit</p> <p>The CAN bus is idle, performs bit (de-) stuffing or is in one of the following frame segments: SOF, SRR, CRC, delimiters, first 6 EOF bits, IFS.</p>
01 _B	<p>NewBit</p> <p>This code represents the first bit of a new frame segment. The current bit is the first bit in one of the following frame segments: Bit 10 (MSB) of standard ID (transmit only), RTR, reserved bits, IDE, DLC(MSB), bit 7 (MSB) in each data byte and the first bit of the ID extension.</p>
10 _B	<p>Bit</p> <p>This code represents a bit inside a frame segment with a length of more than one bit (not the first bit of those frame segments that is indicated by NewBit). The current bit is processed within one of the following frame segments: ID bits (except first bit of standard ID for transmission and first bit of ID extension), DLC (3 LSB) and bits 6-0 in each data byte.</p>
11 _B	<p>Done</p> <p>The current bit is in one of the following frame segments: Acknowledge slot, last bit of EOF, active/passive-error frame, overload frame. Two or more directly consecutive Done codes signal an Error Frame.</p>

Controller Area Network Controller (MultiCAN+)

The Node x Timer Clock Control, Node x Timer Receive Timeout and Node x Timer A/B/C Transmit Trigger Registers offer additional timing functions for the node.

The Node x Timer Clock Control Register NTCCR_x controls the functions of the node timer.

CAN_NTCCR_x (x = 0-4)
Node x Timer Clock Control Register (21C_H+x*100_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
0	[7:0]	r	Reserved Read as 0; should be written with 0.
TPSC	[11:8]	rw	Timer Prescaler The duration of one timer clock is given by (TPSC + 1) CAN bit times for all NTCCR _x .TRIGSRC settings.
0	[17:12]	r	Reserved Read as 0; should be written with 0.
TRIGSRC	[20:18]	rw	Trigger Source This bit selects the trigger source for the different modes in the node timer. 000 _B Node x Timer is decremented per f_{CLC} timing to 0. 001 _B System Timer (STM) trigger event enabled. Node x Timer is decremented per STM trigger event prescaled by (TPSC + 1). 010 _B General Timer (GTM) trigger event enabled. Node x Timer is decremented per GTM trigger event prescaled by (TPSC + 1). 011 _B Reserved, do not use. ... _B ... 111 _B Reserved, do not use.

Controller Area Network Controller (MultiCAN+)

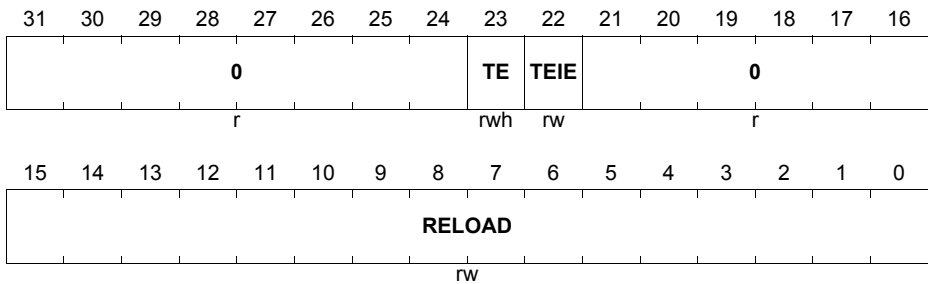
Field	Bits	Type	Description
0	[31:21]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

The Node x Timer Receive Timeout Register NTRTx controls the node timing functions for Receive Timeout Mode.

CAN_NTRTRx (x = 0-4)

Node x Timer Receive Timeout Register (220_H+x*100_H) Reset Value: 0000 0000_H



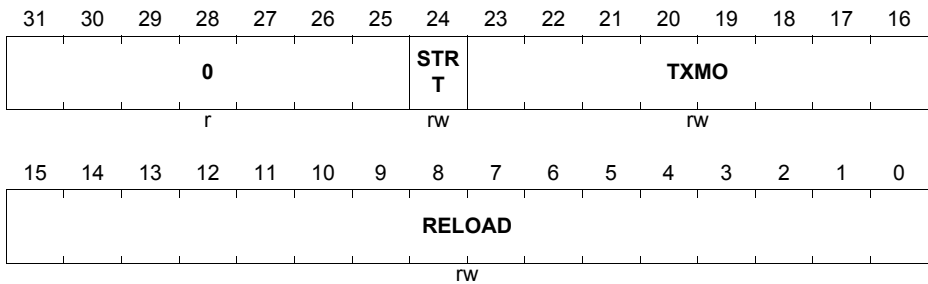
Field	Bits	Type	Description
RELOAD	[15:0]	rw	Reload Value This bit field contains the reload value for the timer. The timer will restart when RELOAD is written.
0	[21:16]	r	Reserved Read as 0; should be written with 0.
TEIE	22	rw	Timer Event Interrupt Enable This bit enables the node timer event interrupt of CAN node x. 0 _B Timer event interrupt is disabled 1 _B Timer event interrupt is enabled Bit field NIPRx.TEINP selects the interrupt output line which becomes activated at this type of interrupt.
TE	23	rwh	Timer Event This flag is set on a node timer transition from 1 to 0 in Receive Timeout Mode. This bit must be reset (i.e Write to '0') by software, writing a '1' has no effect. 0 _B No timer event has occurred since last flag reset 1 _B Timer event has occurred since last flag reset An interrupt request is generated if TEIE = 1.
0	[31:24]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

The Node x Timer A Transmit Trigger Register NTATTx controls the node timing functions for Transmit Trigger Mode.

CAN_NTATTRx (x = 0-4)

Node x Timer A Transmit Trigger Register (224_H+x*100_H) Reset Value: 0000 0000_H



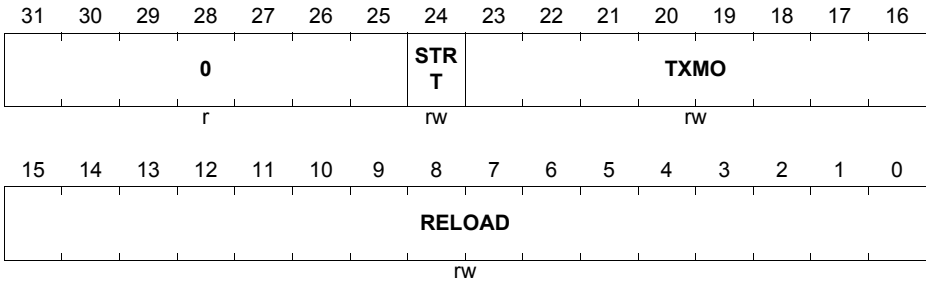
Field	Bits	Type	Description
RELOAD	[15:0]	rw	Reload Value This bit field contains the reload value for the timer. The timer will restart when RELOAD is written.
TXMO	[23:16]	rw	Transmit Message Object On a transmit trigger event selected by bit TRIGSRC, the transmit request bit MOSTATn.TXRQ of message object number n will be set. Note: The NTATTRx.START timer has to be stopped before a new value can be programmed into TXMO, i.e 1) Program NTATTR.START=0 2) Program NTATTR.TXMO with new value
STRT	24	rw	Timer Start This bit field controls the operation of the timer. 0 _B Timer is stopped. 1 _B Timer is started.
0	[31:25]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

The Node x Timer B Transmit Trigger Register NTBTTx controls the node timing functions for Transmit Trigger Mode.

CAN_NTBTTRx (x = 0-4)

Node x Timer B Transmit Trigger Register (228_H+x*100_H) Reset Value: 0000 0000_H



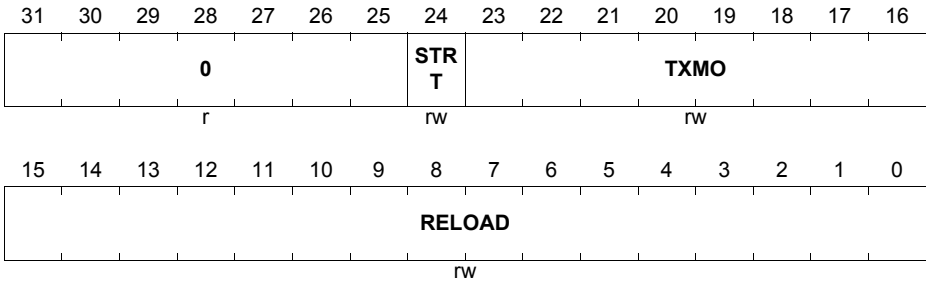
Field	Bits	Type	Description
RELOAD	[15:0]	rw	Reload Value This bit field contains the reload value for the timer. The timer will restart when RELOAD is written.
TXMO	[23:16]	rw	Transmit Message Object On a transmit trigger event selected by bit TRIGSRC, the transmit request bit MOSTATn.TXRQ of message object number n will be set. Note: The NTBTTRx.START timer has to be stopped before a new value can be programmed into TXMO, i.e 1) Program NTBTTRx.START=0 2) Program NTBTTRx.TXMO with new value
STRT	24	rw	Timer Start This bit field controls the operation of the timer. 0 _B Timer is stopped. 1 _B Timer is started.
0	[31:25]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

The Node x Timer C Transmit Trigger Register NTCTTx controls the node timing functions for Transmit Trigger Mode.

CAN_NTCTTRx (x = 0-4)

Node x Timer C Transmit Trigger Register (22C_H+x*100_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
RELOAD	[15:0]	rw	Reload Value This bit field contains the reload value for the timer. The timer will restart when RELOAD is written.
TXMO	[23:16]	rw	Transmit Message Object On a transmit trigger event selected by bit TRIGSRC, the transmit request bit MOSTATn.TXRQ of message object number n will be set. Note: The NTCTTRx.START timer has to be stopped before a new value can be programmed into TXMO, i.e 1) Program NTCTTR.START=0 2) Program NTCTTR.TXMO with new value
STRT	24	rw	Timer Start This bit field controls the operation of the timer. 0 _B Timer is stopped. 1 _B Timer is started.
0	[31:25]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

23.6.3 Message Object Registers

The Message Object Control Register MOCTR_n and the Message Object Status Register MOSTAT_n are located at the same address offset within a message object address block (offset address 1C_H). The MOCTR_n is a write-only register that makes it possible to set/reset CAN transfer related control bits through software. Therefore the reset value is written as 0x0, even though the read part of the register has a different reset value.

CAN_MOCTR_z (z = 0-254)

Message Object z Control Register(101C_H+z*20_H)

Reset Value: 00000000_H

CAN_MOCTR255

Message Object 255 Control Register(FFC_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				SET DIR	SET TXE N1	SET TXE N0	SET TXR Q	SET RXE N	SET RTS EL	SET MSG VAL	SET MSG LST	SET NEW DAT	SET RXU PD	SET TXP ND	SET RXP ND
w				w	w	w	w	w	w	w	w	w	w	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				RES DIR	RES TXE N1	RES TXE N0	RES TXR Q	RES RXE N	RES RTS EL	RES MSG VAL	RES MSG LST	RES NEW DAT	RES RXU PD	RES TXP ND	RES RXP ND
w				w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RESRXPND, SETRXPND	0, 16	w	Reset/Set Receive Pending These bits control the set/reset condition for RXPND (see Table 23-15).
RESTXPND, SETTXPND	1, 17	w	Reset/Set Transmit Pending These bits control the set/reset condition for TXPND (see Table 23-15).
RESRXUPD, SETRXUPD	2, 18	w	Reset/Set Receive Updating These bits control the set/reset condition for RXUPD (see Table 23-15).
RESNEWDAT, SETNEWDAT	3, 19	w	Reset/Set New Data These bits control the set/reset condition for NEWDAT (see Table 23-15).

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
RESMSGST, SETMSGST	4, 20	w	Reset/Set Message Lost These bits control the set/reset condition for MSGST (see Table 23-15).
RESMSGVAL, SETMSGVAL	5, 21	w	Reset/Set Message Valid These bits control the set/reset condition for MSGVAL (see Table 23-15).
RESRTSEL, SETRTSEL	6, 22	w	Reset/Set Receive/Transmit Selected These bits control the set/reset condition for RTSEL (see Table 23-15).
RESRXEN, SETRXEN	7, 23	w	Reset/Set Receive Enable These bits control the set/reset condition for RXEN (see Table 23-15).
RESTXRQ, SETTXRQ	8, 24	w	Reset/Set Transmit Request These bits control the set/reset condition for TXRQ (see Table 23-15).
RESTXEN0, SETTXEN0	9, 25	w	Reset/Set Transmit Enable 0 These bits control the set/reset condition for TXEN0 (see Table 23-15).
RESTXEN1, SETTXEN1	10, 26	w	Reset/Set Transmit Enable 1 These bits control the set/reset condition for TXEN1 (see Table 23-15).
RESDIR, SETDIR	11, 27	w	Reset/Set Message Direction These bits control the set/reset condition for DIR (see Table 23-15).
0	[15:12], [31:28]	w	Reserved Should be written with 0.

Table 23-15 Reset/Set Conditions for Bits in Register MOCTRn

RESy Bit ¹⁾	SETy Bit	Action on Write
Write 0	Write 0	Leave element unchanged
	No write	
No write	Write 0	
Write 1	Write 1	

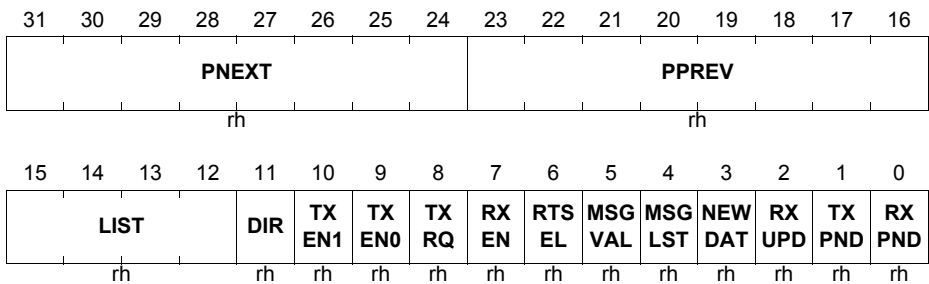
Controller Area Network Controller (MultiCAN+)
Table 23-15 Reset/Set Conditions for Bits in Register MOCTRn (cont'd)

RESy Bit¹⁾	SETy Bit	Action on Write
Write 1	Write 0	Reset element
	No write	
Write 0	Write 1	Set element
No write		

1) The parameter “y” stands for the second part of the bit name (“RXPND”, “TXPND”, ... up to “DIR”).

Controller Area Network Controller (MultiCAN+)

The MOSTATn is a read-only register that indicates message object list status information such as the number of the current message object predecessor and successor message object, as well as the list number to which the message object is assigned.

CAN_MOSTAT0
Message Object 0 Status Register (101C_H)
Reset Value: 0100 0000_H
CAN_MOSTATn (n = 1-254)
Message Object n Status Register(101C_H+n*20_H)
Reset Value: ((n+1)*01000000_H)+((n-1)*00010000_H)
CAN_MOSTAT255
Message Object 255 Status Register (2FFC_H)
Reset Value: FFFE0000_H


Field	Bits	Type	Description
RXPND	0	rh	Receive Pending 0 _B No CAN message has been received. 1 _B A CAN message has been received by the message object n, either directly or via gateway copy action. RXPND is set by hardware and must be reset by software.
TXPND	1	rh	Transmit Pending 0 _B No CAN message has been transmitted. 1 _B A CAN message from message object n has been transmitted successfully over the CAN bus. TXPND is set by hardware and must be reset by software.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
RXUPD	2	rh	Receive Updating 0 _B No receive update ongoing. 1 _B Message identifier, DLC, and data of the message object are currently updated.
NEWDAT	3	rh	New Data 0 _B No update of the message object n since last flag reset. 1 _B Message object n has been updated. NEWDAT is set by hardware after a received CAN frame has been stored in message object n. NEWDAT is cleared by hardware when a CAN transmission of message object n has been started. NEWDAT should be set by software after the new transmit data has been stored in message object n to prevent the automatic reset of TXRQ at the end of an ongoing transmission.
MSGLST	4	rh	Message Lost 0 _B No CAN message is lost. 1 _B A CAN message is lost because NEWDAT has become set again when it has already been set.
MSGVAL	5	rh	Message Valid 0 _B Message object n is not valid. 1 _B Message object n is valid. Only a valid message object takes part in CAN transfers.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
RTSEL	6	rh	<p>Receive/Transmit Selected</p> <p>0_B Message object n is not selected for receive or transmit operation.</p> <p>1_B Message object n is selected for receive or transmit operation.</p> <p>Frame Reception: RTSEL is set by hardware when message object n has been identified for storage of a CAN frame that is currently received. Before a received frame becomes finally stored in message object n, a check is performed to determine if RTSEL is set. Thus the CPU can suppress a scheduled frame delivery to this message object n by clearing RTSEL by software.</p> <p>Frame Transmission: RTSEL is set by hardware when message object n has been identified to be transmitted next. A check is performed to determine if RTSEL is still set before message object n is actually set up for transmission and bit NEWDAT is cleared. It is also checked that RTSEL is still set before its message object n is verified due to the successful transmission of a frame. RTSEL needs to be checked only when the context of message object n changes, and a conflict with an ongoing frame transfer shall be avoided. In all other cases, RTSEL can be ignored. RTSEL has no impact on message acceptance filtering. RTSEL is not cleared by hardware.</p>
RXEN	7	rh	<p>Receive Enable</p> <p>0_B Message object n is not enabled for frame reception.</p> <p>1_B Message object n is enabled for frame reception.</p> <p>RXEN is evaluated for receive acceptance filtering only.</p>

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
TXRQ	8	rh	<p>Transmit Request</p> <p>0_B No transmission of message object n is requested.</p> <p>1_B Transmission of message object n on the CAN bus is requested.</p> <p>The transmit request becomes valid only if TXRQ, TXEN0, TXEN1 and MSGVAL are set. TXRQ is set by hardware if a matching Remote Frame has been received correctly. TXRQ is reset by hardware if message object n has been transmitted successfully and NEWDAT is not set again by software.</p>
TXEN0	9	rh	<p>Transmit Enable 0</p> <p>0_B Message object n is not enabled for frame transmission.</p> <p>1_B Message object n is enabled for frame transmission.</p> <p>Message object n can be transmitted only if both bits, TXEN0 and TXEN1, are set.</p> <p>The user may clear TXEN0 in order to inhibit the transmission of a message that is currently updated, or to disable automatic response of Remote Frames.</p>
TXEN1	10	rh	<p>Transmit Enable 1</p> <p>0_B Message object n is not enabled for frame transmission.</p> <p>1_B Message object n is enabled for frame transmission.</p> <p>Message object n can be transmitted only if both bits, TXEN0 and TXEN1, are set.</p> <p>TXEN1 is used by the MultiCAN+ module for selecting the active message object in the Transmit FIFOs.</p>

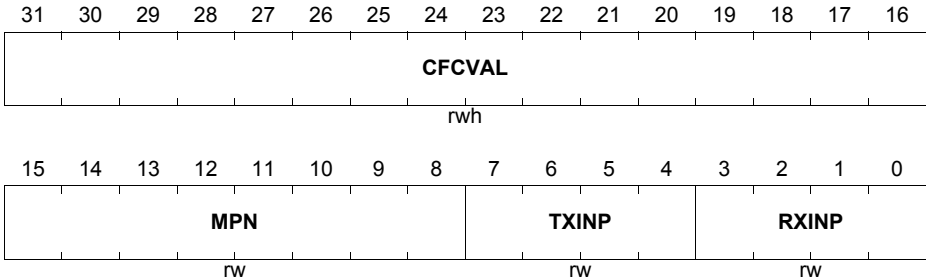
Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
DIR	11	rh	Message Direction 0_B Receive Object selected: With TXRQ = 1, a Remote Frame with the identifier of message object n is scheduled for transmission. On reception of a Data Frame with matching identifier, the message is stored in message object n. 1_B Transmit Object selected: If TXRQ = 1, message object n is scheduled for transmission of a Data Frame. On reception of a Remote Frame with matching identifier, bit TXRQ is set.
LIST	[15:12]	rh	List Allocation LIST indicates the number of the message list to which message object n is allocated. LIST is updated by hardware when the list allocation of the object is modified by a panel command.
PPREV	[23:16]	rh	Pointer to Previous Message Object PPREV holds the message object number of the previous message object in a message list structure.
PNEXT	[31:24]	rh	Pointer to Next Message Object PNEXT holds the message object number of the next message object in a message list structure.

Table 23-16 MOSTATn Reset Values

Message Object	PNEXT	PPREV	Reset Value
0	1	0	0100 0000 _H
1	2	0	0200 0000 _H
2	3	1	0301 0000 _H
3	4	2	0402 0000 _H
...
<u>255</u>	<u>255</u>	<u>254</u>	<u>FFFE</u> 0000 _H

The Message Object Interrupt Pointer Register MOIPRn holds the message interrupt pointers, the message pending number, and the frame counter value of message object n.

Controller Area Network Controller (MultiCAN+)
CAN_MOIPRn (n = 0-255)
Message Object n Interrupt Pointer Register
(1008_H+n*20_H)
Reset Value: 0000 0000_H


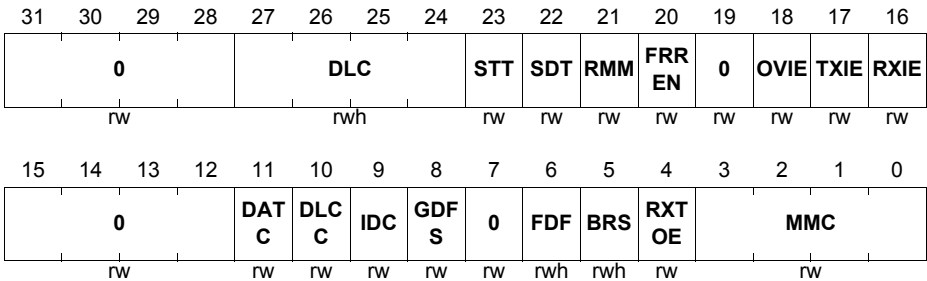
Field	Bits	Type	Description
RXINP	[3:0]	rw	Receive Interrupt Node Pointer RXINP selects the interrupt output line INT_Om (m = 0-15) for a receive interrupt event of message object n. RXINP can also be taken for message pending bit selection (see Page 23-48). 0000 _B Interrupt output line INT_O0 is selected. 0001 _B Interrupt output line INT_O1 is selected. ... _B ... 1110 _B Interrupt output line INT_O14 is selected. 1111 _B Interrupt output line INT_O15 is selected.
TXINP	[7:4]	rw	Transmit Interrupt Node Pointer TXINP selects the interrupt output line INT_Om (m = 0-15) for a transmit interrupt event of message object n. TXINP can also be taken for message pending bit selection (see Page 23-48). 0000 _B Interrupt output line INT_O0 is selected. 0001 _B Interrupt output line INT_O1 is selected. ... _B ... 1110 _B Interrupt output line INT_O14 is selected. 1111 _B Interrupt output line INT_O15 is selected.
MPN	[15:8]	rw	Message Pending Number This bit field selects the bit position of the bit in the Message Pending Register that is set upon a message object n receive/transmit interrupt.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
CFCVAL	[31:16]	rwh	CAN Frame Counter Value When a message is stored in message object n or message object n has been successfully transmitted, the CAN frame counter value NFCRx.CFC is then copied to CFCVAL.

Controller Area Network Controller (MultiCAN+)

The Message Object Function Control Register MOFCR_n contains bits that select and configure the function of the message object. It also holds the CAN data length code.

CAN_MOFCR_n (n = 0-255)
Message Object n Function Control Register
(1000_H+n*20_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
MMC	[3:0]	rw	Message Mode Control MMC controls the message mode of message object n. 0000 _B Standard Message Object 0001 _B Receive FIFO Base Object 0010 _B Transmit FIFO Base Object 0011 _B Transmit FIFO Slave Object 0100 _B Gateway Source Object 0101 _B CANFD 64 bytes Message Mode ... _B ... 1111 _B Do not use
RXTOE	4	rw	Receive Time-Out Enable RXTOE enables participation of message in receive time-out check which is performed via a Timer controlled by the register NTRTR of the corresponding node. 0 _B Message does not take part in receive time-out check 1 _B Message takes part in receive time-out check Applicable to receive messages and to transmit messages waiting for remote frames.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
BRS	5	rwh	Bit Rate Switch 0 _B Message Object transmission/reception without bit rate switching. 1 _B Message Object transmission/reception with bit rate switching. Please see Section 23.4.12.4 and Table 23-18
FDf	6	rwh	CAN FD Frame Format 0 _B Message Object transmission/reception in Classical CAN Frame Format. 1 _B Message Object transmission/reception in CAN FD Format (new DLC coding and CRC) Please see Section 23.4.12.4 and Table 23-18
GDFS	8	rw	Gateway Data Frame Send 0 _B TXRQ is unchanged in the destination object. 1 _B TXRQ is set in the gateway destination object after the internal transfer from the gateway source to the gateway destination object. Applicable only to a gateway source object; ignored in other nodes.
IDC	9	rw	Identifier Copy 0 _B The identifier of the gateway source object is not copied. 1 _B The identifier of the gateway source object (after storing the received frame in the source) is copied to the gateway destination object. Applicable only to a gateway source object; ignored in other nodes.
DLCC	10	rw	Data Length Code Copy 0 _B Data length code is not copied. 1 _B Data length code of the gateway source object (after storing the received frame in the source) is copied to the gateway destination object. Applicable only to a gateway source object; ignored in other nodes.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
DATC	11	rw	<p>Data Copy</p> <p>0_B Data fields are not copied.</p> <p>1_B Data fields in registers MODATALn and MODATAHn of the gateway source object (after storing the received frame in the source) are copied to the gateway destination.</p> <p>Applicable only to a gateway source object; ignored in other nodes.</p>
RXIE	16	rw	<p>Receive Interrupt Enable</p> <p>RXIE enables the message receive interrupt of message object n. This interrupt is generated after reception of a CAN message (independent of whether the CAN message is received directly or indirectly via a gateway action).</p> <p>0_B Message receive interrupt is disabled.</p> <p>1_B Message receive interrupt is enabled.</p> <p>Bit field MOIPRn.RXINP selects the interrupt output line which becomes activated at this type of interrupt.</p>
TXIE	17	rw	<p>Transmit Interrupt Enable</p> <p>TXIE enables the message transmit interrupt of message object n. This interrupt is generated after the transmission of a CAN message.</p> <p>0_B Message transmit interrupt is disabled.</p> <p>1_B Message transmit interrupt is enabled.</p> <p>Bit field MOIPRn.TXINP selects the interrupt output line which becomes activated at this type of interrupt.</p>

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
OVIE	18	rw	<p>Overflow Interrupt Enable</p> <p>OVIE enables the FIFO full interrupt of message object n. This interrupt is generated when the pointer to the current message object (CUR) reaches the value of SEL in the FIFO/Gateway Pointer Register.</p> <p>0_B FIFO full interrupt is disabled. 1_B FIFO full interrupt is enabled.</p> <p>If message object n is a Receive FIFO base object, bit field MOIPRn.TXINP selects the interrupt output line which becomes activated at this type of interrupt. If message object n is a Transmit FIFO base object, bit field MOIPRn.RXINP selects the interrupt output line which becomes activated at this type of interrupt. For all other message object modes, bit OVIE has no effect.</p>
FRREN	20	rw	<p>Foreign Remote Request Enable</p> <p>Specifies whether the TXRQ bit is set in message object n or in a foreign message object referenced by the pointer CUR.</p> <p>0_B TXRQ of message object n is set on reception of a matching Remote Frame. 1_B TXRQ of the message object referenced by the pointer CUR is set on reception of a matching Remote Frame.</p>
RMM	21	rw	<p>Transmit Object Remote Monitoring</p> <p>0_B Remote monitoring is disabled: Identifier, IDE bit, and DLC of message object n remain unchanged upon the reception of a matching Remote Frame. 1_B Remote monitoring is enabled: Identifier, IDE bit, and DLC of a matching Remote Frame are copied to transmit object n in order to monitor incoming Remote Frames. Bit RMM applies only to transmit objects and has no effect on receive objects.</p>

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
SDT	22	rw	<p>Single Data Transfer</p> <p>If SDT = 1 and message object n is not a FIFO base object, then MSGVAL is reset when this object has taken part in a successful data transfer (receive or transmit).</p> <p>If SDT = 1 and message object n is a FIFO base object, then MSGVAL is reset when the pointer to the current object CUR reaches the value of SEL in the FIFO/Gateway Pointer Register.</p> <p>With SDT = 0, bit MSGVAL is not affected.</p>
STT	23	rw	<p>Single Transmit Trial</p> <p>If this bit is set, then TXRQ is cleared on transmission start of message object n. Thus, no transmission retry is performed in case of transmission failure.</p>
DLC	[27:24]	rwh	<p>Data Length Code</p> <p>Bit field determines the number of data bytes for message object n.</p> <p>In Classical CAN Format: A value of DLC > 8 results in a data length of 8 data bytes. If a frame with DLC > 8 is received, the received value is stored in the message object.</p> <p>In CAN FD format, valid values for DLC are 0 to 15. See Table 23-17</p>
0	7, [15:12], 19, [31:28]	rw	<p>Reserved</p> <p>Read as 0 after reset; value last written is read back; should be written with 0.</p>

Coding of DLC in CAN FD
Table 23-17 Coding of DLC in CAN FD

DLC Value	Description
0000 _B	0 Data Byte for Message Object n.
... _B	...
1000 _B	8 Data Byte for Message Object n.
1001 _B	12 Data Byte for Message Object n.
1010 _B	16 Data Byte for Message Object n.

Controller Area Network Controller (MultiCAN+)
Table 23-17 Coding of DLC in CAN FD (cont'd)

DLC Value	Description
1011 _B	20 Data Byte for Message Object n.
1100 _B	24 Data Byte for Message Object n.
1101 _B	32 Data Byte for Message Object n.
1110 _B	48 Data Byte for Message Object n.
1111 _B	64 Data Byte for Message Object n.

CAN FD Transmit And Receive Behavior

Message transmission and reception behavior of CAN data frames is summarized at **Table 23-18** below.

Table 23-18 CAN FD Transmit And Receive Behavior

FDEN	FDF¹⁾	BRS¹⁾	Transmit Behavior
0 _B	0 _B	0 _B	Classical CAN Frames (ISO 11898-1)
0 _B	0 _B	1 _B	Classical CAN Frames (ISO 11898-1)
0 _B	1 _B	0 _B	Transmission Cancelled ²⁾ (i.e TXRQ bit cleared)
0 _B	1 _B	1 _B	Transmission Cancelled ²⁾ (i.e TXRQ bit cleared)
1 _B	0 _B	0 _B	Classical CAN Frames (ISO 11898-1)
1 _B	0 _B	1 _B	Classical CAN Frames (ISO 11898-1)
1 _B	1 _B	0 _B	Long Frame (i.e CAN FD frame with BRS = 0, whole frame transmitted with slow baudrate)
1 _B	1 _B	1 _B	Long + Fast Frame (i.e CAN FD frame with BRS = 1, switching fast baudrate for dataphase)
FDEN	FDF³⁾	BRS³⁾	Receive Behavior⁴⁾
0 _B	0/1 _B	0/1 _B	- Classical CAN Frames
1 _B	0/1 _B	0/1 _B	- Classical CAN Frames - Long Frames - Long + Fast Frames

1) User writes

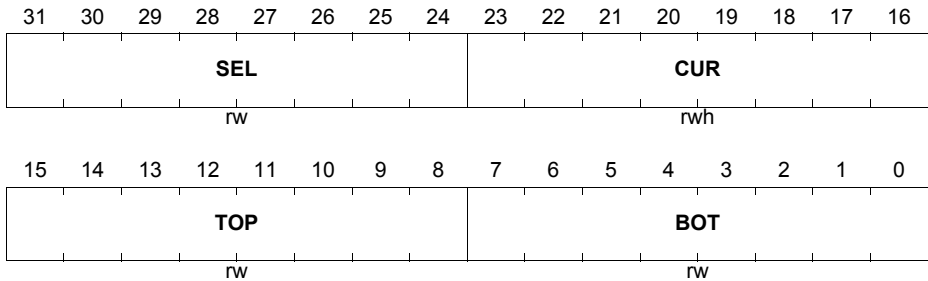
2) TXRQ in message object is cleared upon transmit setup and transmit setup cancelled. CAN node is not blocked and able to transmit other message objects.

3) Hardware writes

4) Please note that Remote Frames Request do not change BRS and FDF bits.

Controller Area Network Controller (MultiCAN+)

The Message Object FIFO/Gateway Pointer register MOFGPRn contains a set of message object link pointers that are used for FIFO and gateway operations.

CAN_MOFGPRn (n = 0-255)
Message Object n FIFO/Gateway Pointer Register
 $(1004_H + n * 20_H)$
Reset Value: 0000 0000_H


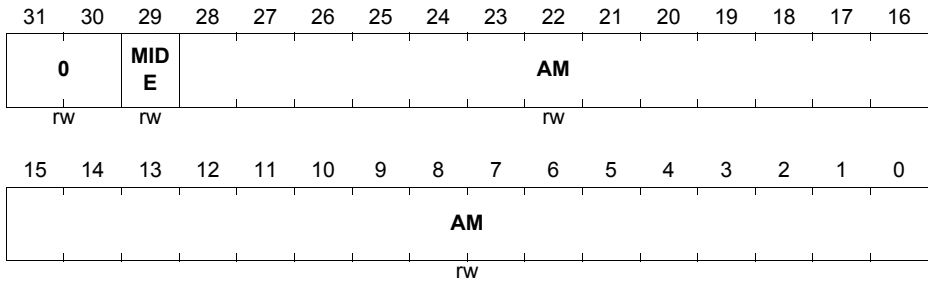
Field	Bits	Type	Description
BOT	[7:0]	rw	Bottom Pointer Bit field BOT points to the first element in a FIFO structure. Or in the case when MOFCR.MMC=5, CAN FD 64 bytes message mode, BOT points to where Data bytes 8-35 are stored in the message object.
TOP	[15:8]	rw	Top Pointer Bit field TOP points to the last element in a FIFO structure. Or in the case when MOFCR.MMC=5, CAN FD 64 bytes message mode, BOT points to where Data bytes 36-63 are stored in the message object.
CUR	[23:16]	rwh	Current Object Pointer Bit field CUR points to the actual target object within a FIFO/Gateway structure. After a FIFO/gateway operation CUR is updated with the message number of the next message object in the list structure (given by PNEXT of the Message Object Status Register) until it reaches the FIFO top element (given by TOP) when it is reset to the bottom element (given by BOT).

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
SEL	[31:24]	rw	Object Select Pointer Bit field SEL is the second (software) pointer to complement the hardware pointer CUR in the FIFO structure. SEL is used for monitoring purposes (FIFO interrupt generation).

Controller Area Network Controller (MultiCAN+)

Message Object n Acceptance Mask Register MOAMRn contains the mask bits for the acceptance filtering of the message object n.

CAN_MOAMRn (n = 0-255)
Message Object n Acceptance Mask Register
 $(100C_H + n * 20_H)$
Reset Value: 3FFF FFFF_H


Field	Bits	Type	Description
AM	[28:0]	rw	Acceptance Mask for Message Identifier Bit field AM is the 29-bit mask for filtering incoming messages with standard identifiers (AM[28:18]) or extended identifiers (AM[28:0]). For standard identifiers, bits AM[17:0] are “don’t care”.
MIDE	29	rw	Acceptance Mask Bit for Message IDE Bit 0 _B Message object n accepts the reception of both, standard and extended frames. 1 _B Message object n receives frames only with matching IDE bit.
0	[31:30]	rw	Reserved Read as 0 after reset; value last written is read back; should be written with 0.

Controller Area Network Controller (MultiCAN+)

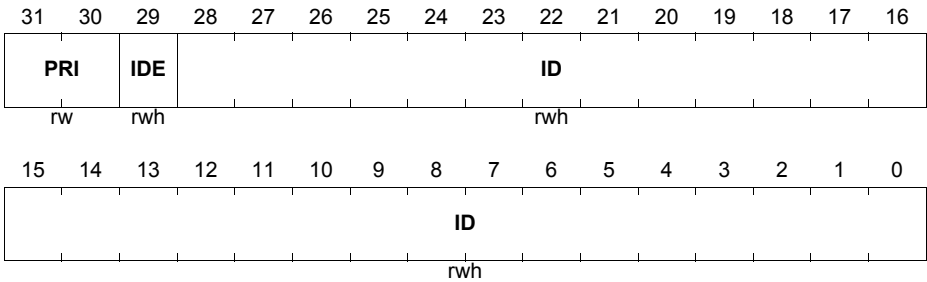
Message Object n Arbitration Register MOARn contains the CAN identifier of the message object.

CAN_MOARn (n = 0-255)

Message Object n Arbitration Register

$(1018_H + n * 20_H)$

Reset Value: 0000 0000_H



Field	Bits	Type	Description
ID	[28:0]	rwh	CAN Identifier of Message Object n Identifier of a standard message (ID[28:18]) or an extended message (ID[28:0]). For standard identifiers, bits ID[17:0] are “don’t care”.
IDE	29	rwh	Identifier Extension Bit of Message Object n 0 _B Message object n handles standard frames with 11-bit identifier. 1 _B Message object n handles extended frames with 29-bit identifier.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
PRI	[31:30]	rw	<p>Priority Class</p> <p>PRI assigns one of the four priority classes 0, 1, 2, 3 to message object n. A lower PRI number defines a higher priority. Message objects with lower PRI value always win acceptance filtering for frame reception and transmission over message objects with higher PRI value. Acceptance filtering based on identifier/mask and list position is performed only between message objects of the same priority class. PRI also determines the acceptance filtering method for transmission:</p> <p>00_B Reserved.</p> <p>01_B Transmit acceptance filtering is based on the list order. This means that message object n is considered for transmission only if there is no other message object with valid transmit request (MSGVAL & TXEN0 & TXEN1 = 1) somewhere before this object in the list.</p> <p>10_B Transmit acceptance filtering is based on the CAN identifier. This means, message object n is considered for transmission only if there is no other message object with higher priority identifier + IDE + DIR (with respect to CAN arbitration rules) somewhere in the list (see Table 23-19).</p> <p>11_B Transmit acceptance filtering is based on the list order (as PRI = 01_B).</p>

Controller Area Network Controller (MultiCAN+)

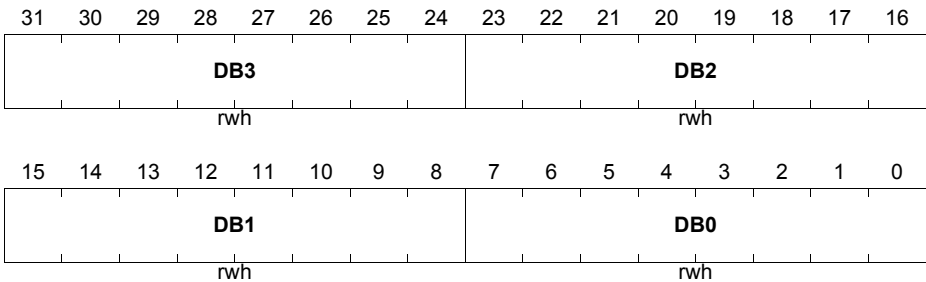
Transmit Priority of Msg. Objects based on CAN Arbitration Rules

Table 23-19 Transmit Priority of Msg. Objects Based on CAN Arbitration Rules

Settings of Arbitrarily Chosen Message Objects A and B, (A has higher transmit priority than B)	Comment
A.MOAR[28:18] < B.MOAR[28:18] (11-bit standard identifier of A less than 11-bit standard identifier of B)	Messages with lower standard identifier have higher priority than messages with higher standard identifier. MOAR[28] is the most significant bit (MSB) of the standard identifier. MOAR[18] is the least significant bit of the standard identifier.
A.MOAR[28:18] = B.MOAR[28:18] A.MOAR.IDE = 0 (send Standard Frame) B.MOAR.IDE = 1 (send Extended Frame)	Standard Frames have higher transmit priority than Extended Frames with equal standard identifier.
A.MOAR[28:18] = B.MOAR[28:18] A.MOAR.IDE = B.MOAR.IDE = 0 A.MOSTAT.DIR = 1 (send Data Frame) B.MOSTAT.DIR = 0 (send Remote Fame)	Standard Data Frames have higher transmit priority than standard Remote Frames with equal identifier.
A.MOAR[28:0] = B.MOAR[28:0] A.MOAR.IDE = B.MOAR.IDE = 1 A.MOSTAT.DIR = 1 (send Data Frame) B.MOSTAT.DIR = 0 (send Remote Frame)	Extended Data Frames have higher transmit priority than Extended Remote Frames with equal identifier.
A.MOAR[28:0] < B.MOAR[28:0] A.MOAR.IDE = B.MOAR.IDE = 1 (29-bit identifier)	Extended Frames with lower identifier have higher transmit priority than Extended Frames with higher identifier. MOAR[28] is the most significant bit (MSB) of the overall identifier (standard identifier MOAR[28:18] and identifier extension MOAR[17:0]). MOAR[0] is the least significant bit (LSB) of the overall identifier.

Controller Area Network Controller (MultiCAN+)

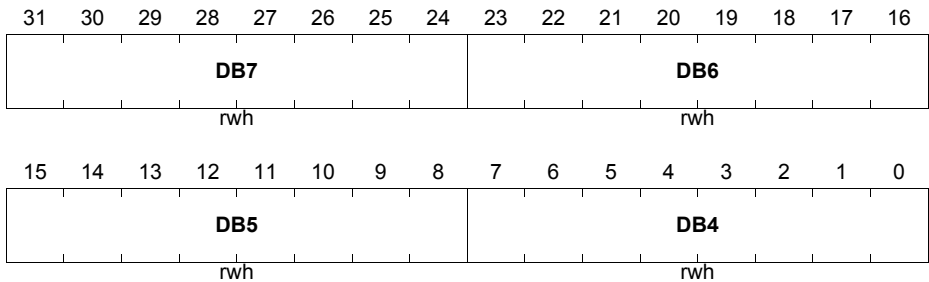
Message Object n Data Register Low MODATALn contains the lowest four data bytes of message object n. Unused data bytes are set to zero upon reception and ignored for transmission.

CAN_MODATALn (n = 0-255)
Message Object n Data Register Low
 $(1010_H + n * 20_H)$
Reset Value: 0000 0000_H


Field	Bits	Type	Description
DB0	[7:0]	rwh	Data Byte 0 of Message Object n
DB1	[15:8]	rwh	Data Byte 1 of Message Object n
DB2	[23:16]	rwh	Data Byte 2 of Message Object n
DB3	[31:24]	rwh	Data Byte 3 of Message Object n

Controller Area Network Controller (MultiCAN+)

Message Object n Data Register High MODATAH contains the highest four data bytes of message object n. Unused data bytes are set to zero upon reception and ignored for transmission.

CAN_MODATAHn (n = 0-255)
Message Object n Data Register High
(1014_H + n*20_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
DB4	[7:0]	rwh	Data Byte 4 of Message Object n
DB5	[15:8]	rwh	Data Byte 5 of Message Object n
DB6	[23:16]	rwh	Data Byte 6 of Message Object n
DB7	[31:24]	rwh	Data Byte 7 of Message Object n

Controller Area Network Controller (MultiCAN+)

Extended Message Object n Data Register represents the alternate register view for MOFCR, MOFGPR, MOIPR, MOAMR, MODATAL, MODATAH and MOAR registers as pointed using (MOFGPR.TOP, MOFGPR.BOT) on the message object with message mode chosen in CAN FD 64 bytes message mode, (i.e MOFCR.MMC = 5). (See [Section 23.4.12.10](#))

CAN_EMOzDATA0 (z = 0-255)
Extended Message Object z Data 0 Register

$$(1000_H + z * 20_H)$$

Reset Value: 0000 0000_H
CAN_EMOzDATA1 (z = 0-255)
Extended Message Object z Data 1 Register

$$(1004_H + z * 20_H)$$

Reset Value: 0000 0000_H
CAN_EMOzDATA2 (z = 0-255)
Extended Message Object z Data 2 Register

$$(1008_H + z * 20_H)$$

Reset Value: 0000 0000_H
CAN_EMOzDATA3 (z = 0-255)
Extended Message Object z Data 3 Register

$$(100C_H + z * 20_H)$$

Reset Value: 0000 0000_H
CAN_EMOzDATA4 (z = 0-255)
Extended Message Object z Data 4 Register

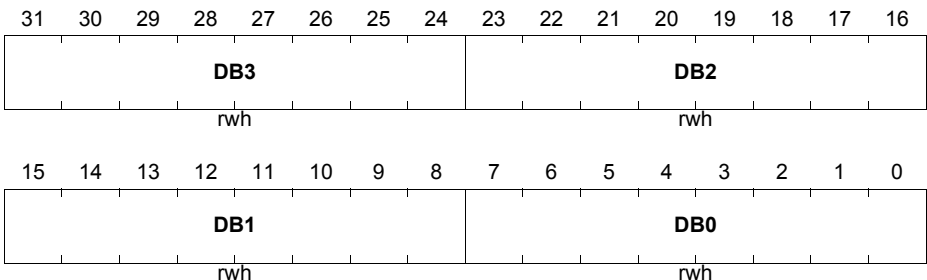
$$(1010_H + z * 20_H)$$

Reset Value: 0000 0000_H
CAN_EMOzDATA5 (z = 0-255)
Extended Message Object z Data 5 Register

$$(1014_H + z * 20_H)$$

Reset Value: 0000 0000_H
CAN_EMOzDATA6 (z = 0-255)
Extended Message Object z Data 6 Register

$$(1018_H + z * 20_H)$$

Reset Value: 0000 0000_H


Field	Bits	Type	Description
DB0	[7:0]	rwh	Data Byte 0 of Message Object n
DB1	[15:8]	rwh	Data Byte 1 of Message Object n

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
DB2	[23:16]	rwh	Data Byte 2 of Message Object n
DB3	[31:24]	rwh	Data Byte 3 of Message Object n

Controller Area Network Controller (MultiCAN+)

23.7 MultiCAN+ Module Implementation

This section describes CAN module interfaces with the clock control, port connections, interrupt control, and address decoding.

23.7.1 Interfaces of the MultiCAN+ Module

shows the TC26x B-Step specific implementation details and interconnections of the MultiCAN+ module. The I/O lines of the MultiCAN+ module (two I/O lines of each CAN node) are connected to the Ports listed in. The MultiCAN+ module is also supplied by clock control, interrupt control, and address decoding logic. MultiCAN+ interrupts can be directed to the DMA, CPU, GTM modules which are able to trigger DMA transfers and GTM, CPU operations.

Controller Area Network Controller (MultiCAN+)

23.7.2 MultiCAN+ Module External Registers

The registers listed in [Figure 23-27](#) are not included in the MultiCAN+ module kernel, some registers must be programmed for proper operation of the MultiCAN+ module.

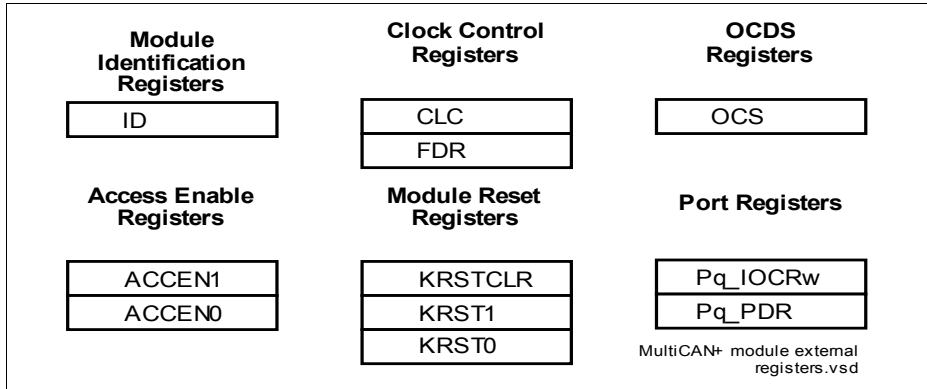


Figure 23-27 CAN Implementation-specific Special Function Registers

Table 23-20 MultiCAN+ Module External Registers

Short Name	Description	Offset Addr	Access Mode		Reset Class	Description see
			Read	Write		
Module Identification Registers						
ID	Module Identification Register	008 _H	U, SV	nBE	Application Reset	Page 23-78
Clock Control Registers						
CLC	Clock Control Register	000 _H	U, SV	SV, E, P	Application Reset	Page 23-16 4
FDR	Fractional Divider Register	00C _H	U, SV	SV, E, P	Application Reset	Page 23-16 5
OCDS Registers						
OCS	OCDS Control and Status Register	0E8 _H	U, SV	SV, P	1	Page 23-15 2
Module Reset Registers						

Controller Area Network Controller (MultiCAN+)
Table 23-20 MultiCAN+ Module External Registers (cont'd)

Short Name	Description	Offset Addr	Access Mode		Reset Class	Description see
			Read	Write		
KRSTCLR	Reset Status Clear Register	0EC _H	U, SV	SV, E, P	3	Page 23-15 9
KRST1	Reset Control Register 1	0F0 _H	U, SV	SV, E, P	3	Page 23-15 8
KRST0	Reset Control Register 0	0F4 _H	U, SV	SV, E, P	3	Page 23-15 6

Access Enable Registers

ACCEN1	Access Enable Register 1	0F8 _H	U, SV	SV, SE	3	Page 23-15 5
ACCEN0	Access Enable Register 0	0FC _H	U, SV	SV, SE	3	Page 23-15 4

Controller Area Network Controller (MultiCAN+)

23.7.2.1 System Registers

OCDS Trigger Bus (OTGB)

The OCDS Control and Status (OCS) register is cleared by Debug Reset. The OCS register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS register value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32bit wide only and requires Supervisor Mode.

OCS
OCDS Control and Status (0E8_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SUS STA	SUS _P	SUS				0								
r	rh	w	rw				r								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												TG _P	TGB	TGS	
r												w	rw	rw	

Field	Bits	Type	Description
TGS	[1:0]	rw	Trigger Set for OTGB0/1 0 _H No Trigger Set output 1 _H TS16_CAN others , reserved
TGB	2	rw	OTGB0/1 Bus Select 0 _B Trigger Set is output on OTGB0 1 _B Trigger Set is output on OTGB1
TG_P	3	w	TGS, TGB Write Protection TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately. Do not use this mode in normal CAN applications, this mode is meant for debugging the peripheral IP. 2 _H Soft suspend of selected CAN nodes. Individually controlled with NCRx.SUSEN. others, reserved
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B CAN nodes are not (yet) suspended 1 _B All selected (SUS) CAN nodes are suspended
0	[23:4], [31:30]	r	Reserved Read as 0; must be written with 0.

Controller Area Network Controller (MultiCAN+)
Access Enable Register 0

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B,... ,EN31 -> TAG ID 011111_B.

ACCEN0
Access Enable Register 0
(0FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Field	Bits	Type	Description
ENn (n = 0-31)	n	r/w	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

Controller Area Network Controller (MultiCAN+)

Access Enable Register 1

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 10000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN0 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

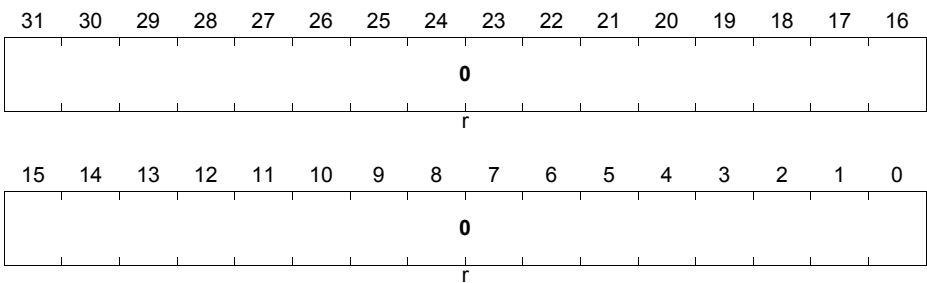
Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 10000_B, EN1 -> TAG ID 100001_B, ... ,EN31 -> TAG ID 111111_B.

ACCEN1

Access Enable Register 1

(0F8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers related to the module kernel that should be reset (kernel 0 or kernel 1). The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing to it with '1'.

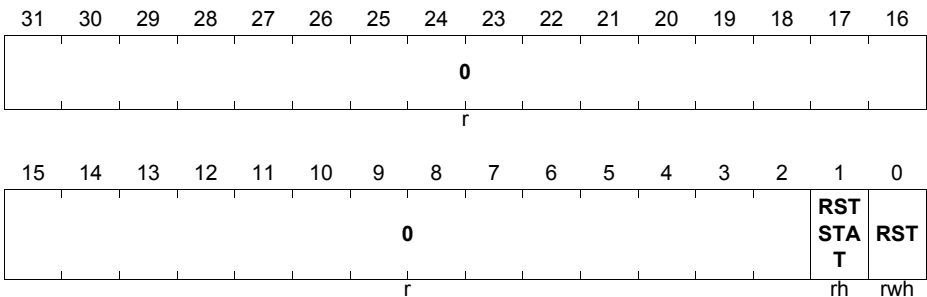
Note: During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

KRST0

Kernel Reset Register 0

(0F4_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set.</p> <p>0_B No kernel reset was requested</p> <p>1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p>

Controller Area Network Controller (MultiCAN+)

Field	Bits	Type	Description
RSTSTAT	1	rw	<p>Kernel Reset Status</p> <p>This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits.</p> <p>0_B No kernel reset was executed 1_B Kernel reset was executed</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.</p>
0	[31:2]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Controller Area Network Controller (MultiCAN+)

Kernel Reset Register 1

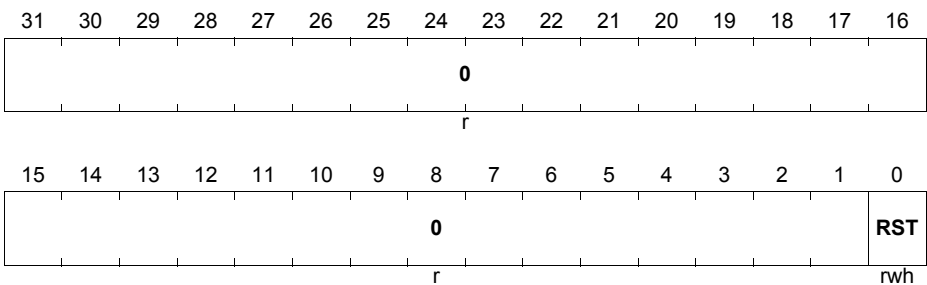
The Kernel Reset Register 1 is used to reset the related module kernel. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers (CAN_KRSTx1.RST and CAN_KRSTx0.RST) related to the module kernel that should be reset (kernel 0 or kernel 1). The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

KRST1

Kernel Reset Register 1

(0F0_H)

Reset Value: 0000 0000_H

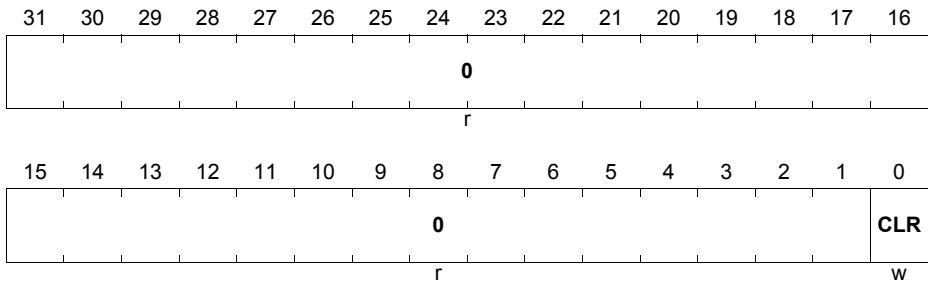


Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set.</p> <p>0_B No kernel reset was requested 1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p>
0	[31:1]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Controller Area Network Controller (MultiCAN+)

Kernel Reset Clear Register

The Kernel Reset Clear Register is used to clear the Kernel Reset Status bit (CAN_KRST0.RSTSTAT).

KRSTCLR
Kernel Reset Status Clear Register (0EC_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Controller Area Network Controller (MultiCAN+)

23.7.3 MultiCAN+ Clock Interconnects With SCU

The MultiCAN+ module clock inputs are connected to System Control Unit (SCU) clock control unit (CCU) as shown in Figure 23-28 and Table 23-21 below.

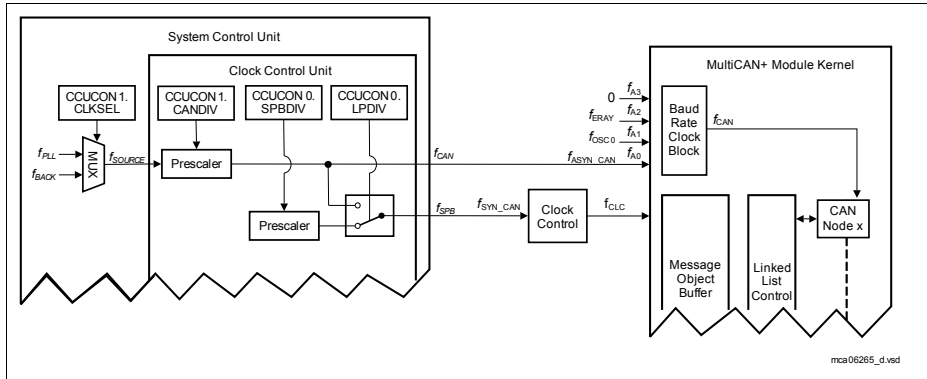


Figure 23-28 MultiCAN+ Clock Interconnects with SCU

Table 23-21 MultiCAN+ Clock Interconnects

CAN Clock Inputs	Connected to	Description
f_{ASYN_CAN}	SCU	f_{ASYN_CAN} of the MultiCAN+ module is one of the clock inputs of the Baud Rate Clock Block, providing the MultiCAN+ module timer clock f_{CAN} . f_{ASYN_CAN} connects to SCU CCU f_{CAN} and is controlled by CCUCON1.CANDIV. The CCUCON1.CANDIV is able to select different values of prescalers based on a selectable f_{PLL} or f_{BACK} source.
f_{SYN_CAN}	SCU	f_{SYN_CAN} of the MultiCAN+ module is the clock input of the Clock Control Register Block, providing the main kernel clock f_{CLC} . f_{SYN_CAN} connects to SCU CCU and is multiplexed between f_{SPB} and f_{CAN} . In normal mode, f_{SYN_CAN} is connected to f_{SPB} where else in pretended networking mode (CCUCON0.LP DIV > 0), f_{SYN_CAN} connects to f_{CAN} of SCU CCU.

Note: The f_{CAN} mentioned on the SCU chapter does not refer to the same f_{CAN} that is mentioned on the MultiCAN+ chapter.

Controller Area Network Controller (MultiCAN+)

11. The f_{CAN} mentioned on the SCU chapter refers in general to connection for MultiCAN+ clocking inputs f_{ASYN_CAN} and f_{SYN_CAN} . Depending on the settings of CCUCON1.CANDIV and CCUCON0.LPDIV different clock sources are connected (See [Table 23-21](#)).
12. The f_{CAN} mentioned on the MultiCAN+ chapter refers to the module timer clock (See [Section 23.7.4.2](#)).

Controller Area Network Controller (MultiCAN+)

23.7.4 Module Clock Generation

This chapter describes the way the module get's its clock.

23.7.4.1 Clock Selection

The bit timing machine and the rest of the MultiCAN+ module are separate frequency domains and can be driven by separate independent frequencies. The bit timing unit can be driven by the SPB bus clock or with the direct oscillator clock, and the rest of the chip is driven only by the SPB bus clock.

The purpose of supplying the bit timing unit with a direct oscillator clock is to avoid the clock jitter added by the PLL, necessary when the chip is driven by a low cost ceramic resonator instead of by high precision quartz crystal.

Selecting the clock source for the bit timing unit is done by programming the bit-field MCR.CLKSEL.

Enabling and disabling the clock of the module by using CLC.DISR affects always both frequency domains, so that when f_{CLC} is switched off, f_A is also switched off.

23.7.4.2 Fractional Divider

As shown in [Figure 23-29](#), the clock signals for the MultiCAN+ module are generated and controlled by a clock control unit. This clock generation unit is responsible for the enable/disable control, the clock frequency adjustment, and the debug clock control. This unit includes two registers:

- CAN_CLC: generation of the module control clock f_{CLC}
- CAN_FDR: frequency control of the module timer clock f_{CAN}

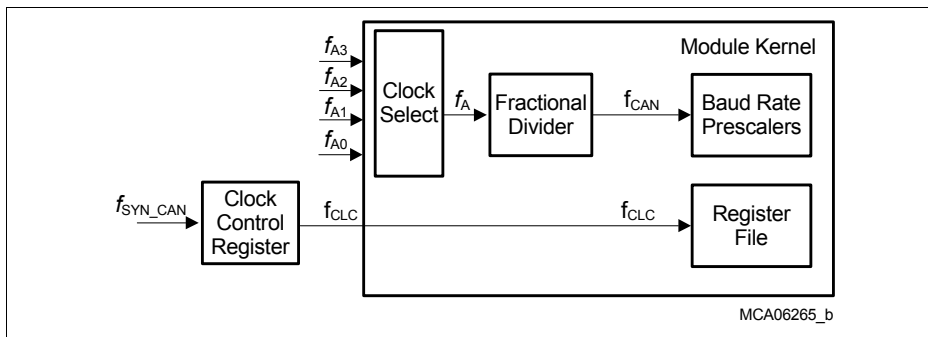


Figure 23-29 MultiCAN+ Module Clock Generation

The f_{SYN_CAN} is identical to f_{SPB} .

Controller Area Network Controller (MultiCAN+)

The module control clock f_{CLC} is used inside the MultiCAN+ module for control purposes such as clocking of control logic and register operations. The frequency of f_{CLC} is identical to the system clock frequency f_{SPB} . The clock control register CAN_CLC makes it possible to enable/disable f_{CLC} under certain conditions.

The module timer clock f_{CAN} is used inside the MultiCAN+ module as input clock for all timing relevant operations (e.g. bit timing). The settings in the CAN_FDR register determine the frequency of the module timer clock f_{CAN} according the following two formulas:

$$f_{CAN} = f_A \times \frac{1}{n} \quad \text{with } n = 1024 - \text{CAN_FDR.STEP} \quad (23.2)$$

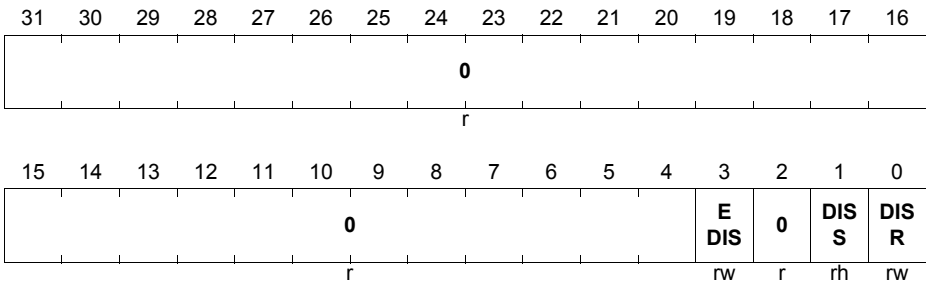
$$f_{CAN} = f_A \times \frac{n}{1024} \quad \text{with } n = 0-1023 \quad (23.3)$$

Equation (23.2) applies to normal divider mode (CAN_FDR.DM = 01_B) of the fractional divider. **Equation (23.3)** applies to fractional divider mode (CAN_FDR.DM = 10_B).

Note: The CAN module is disabled after reset. In general, after reset, the module control clock f_{CLC} must be switched on (writing to register CAN_CLC) before the frequency of the module timer clock f_{CAN} is defined (writing to register CAN_FDR).

Controller Area Network Controller (MultiCAN+)
CAN Clock Control Register

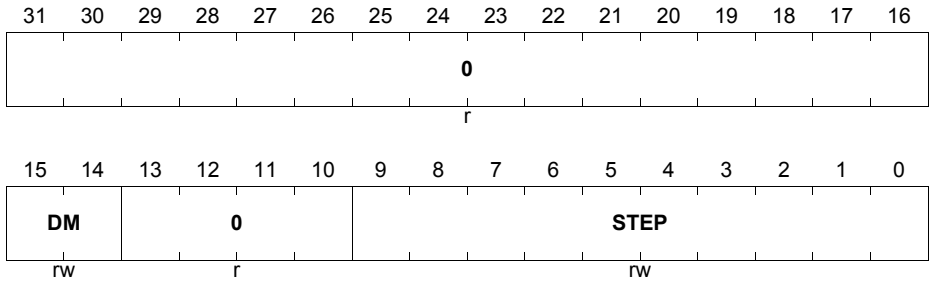
The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the f_{CAN} module clock signal, sleep mode and fast shut-off mode for the module.

CLC
CAN Clock Control Register
(000_H)
Reset Value: 0000 0003_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module. Note that no register access is possible to any register while module is disabled.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
EDIS	3	rw	Sleep Mode Enable Control Used to control module's sleep mode.
0	[31:4], 2	r	Reserved Read as 0; should be written with 0.

Note: The number of module clock cycles (wait states) which are required by the kernel to execute a read or write access depends on the selected CLC clock frequency.

The fractional divider register allows the programmer to control the clock rate of the module timer clock f_{CAN} .

Controller Area Network Controller (MultiCAN+)
FDR
CAN Fractional Divider Register
(00C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
STEP	[9:0]	rw	Step Value Reload or addition value for the result.
DM	[15:14]	rw	Divider Mode This bit field selects normal divider mode, fractional divider mode, and off-state.
0	[13:10], [31:16]	r	Reserved Read as 0; should be written with 0.

23.7.5 Port and I/O Line Control

The interconnections between the MultiCAN+ module and the port I/O lines are controlled in the port logic. Additionally to the port input selection, the following port control operations must be executed:

- Input/output function selection (IOCR registers)
- Pad driver characteristics selection for the outputs (PDR registers)

23.7.5.1 Input/Output Function Selection in Ports

The port input/output control registers contain the bit fields that select the digital output and input driver characteristics such as pull-up/down devices, port direction (input/output), open-drain, and alternate output selections. The I/O lines for the MultiCAN+ module are controlled by the port input/output control registers, which are described in the port chapter. In case of discrepancies between port chapter and CAN chapter, the description in the port chapter is correct.

shows the corresponding pins, sorted by node. Even though the table is pair wise, it is possible to select a different pairing for RXD and TXD. In addition the RXSEL value to be programmed for the Receive Pins is part of this table. For more information on RXSEL please see [Chapter 23.7.5.2](#).

23.7.5.2 Node Receive Input Selection

Additionally to the I/O control selection, as defined in the port chapter, the selection of a CAN node's receive input line requires that bit field RXSEL in its node port control register NPCRx must be set according to . Values for NPCRx.RXSEL other than those of the table mentioned above will result in a recessive receive input for node x. As a hint A results in 0x0, B in 0x1 until H resulting in the value of 0x7.

This feature allows, for example, a CAN node which operates in analyzer mode to monitor the receive operations of its neighbor CAN node.

Controller Area Network Controller (MultiCAN+)

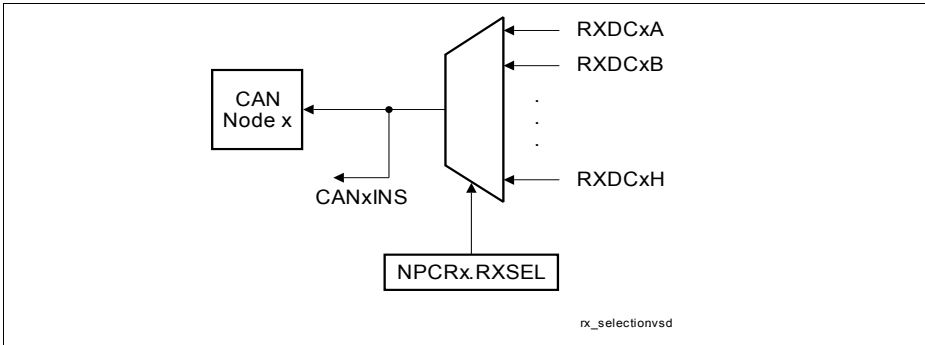


Figure 23-30 CAN Module Receive Input Selection

23.7.5.3 CAN Transmit Trigger Inputs

The CAN transmit trigger inputs of MultiCAN are used to trigger for transmission of a message. In the TC26x B-Step, these input lines are connected as shown in [Table 23-22](#). For the TC26x device the 5th node has no trigger inputs.

Table 23-22 CAN Transmit Trigger Inputs

Receive Input	Connected to	From / to Module
CAN Node 0		
STM_N0_TRIG	stm0.system_irq_o(0)	System Timer Module (STM)
GTM_N0_TRIG	gtm.can_trig_0_o	General Timer Module (GTM)
...		
...
CAN Node 4		
STM_N4_TRIG	stm0.system_irq_o(0)	System Timer Module (STM)
GTM_N4_TRIG	gtm.can_trig_4_o	General Timer Module (GTM)

23.7.5.4 Connections to Interrupt Router Inputs

The interrupt output line INT_O0-15 is connected to the Interrupt Router module, see [Table 23-23](#).

Controller Area Network Controller (MultiCAN+)
Table 23-23 Interrupt Router Inputs

Interrupt Router Input	Connected to CAN Interrupt Output
SRC_CANINT0	INT_O0
SRC_CANINT1	INT_O1
SRC_CANINT2	INT_O2
SRC_CANINT3	INT_O3
SRC_CANINT4	INT_O4
SRC_CANINT5	INT_O5
SRC_CANINT6	INT_O6
SRC_CANINT7	INT_O7
SRC_CANINT8	INT_O8
SRC_CANINT9	INT_O9
SRC_CANINT10	INT_O10
SRC_CANINT11	INT_O11
SRC_CANINT12	INT_O12
SRC_CANINT13	INT_O13
SRC_CANINT14	INT_O14
SRC_CANINT15	INT_O15

23.7.5.5 Connections to General Timer Module (GTM) Inputs

The interrupt output line INT_O12-15 of MultiCAN+ is connected to the General Timer Module, see [Table 23-24](#).

Table 23-24 General Timer Module Inputs

GTM Input	Connected to CAN Interrupt Output
Timer input (gtm.tim_0_muxin_1_i(13))	INT_O12 (mcan0.int_req_o(12))
Timer input (gtm.tim_1_muxin_1_i(13))	
Timer input (gtm.tim_2_muxin_1_i(13))	
Timer input (gtm.tim_0_muxin_2_i(13))	INT_O13 (mcan0.int_req_o(13))
Timer input (gtm.tim_1_muxin_2_i(13))	
Timer input (gtm.tim_2_muxin_2_i(13))	

Controller Area Network Controller (MultiCAN+)**Table 23-24 General Timer Module Inputs**

GTM Input	Connected to CAN Interrupt Output
Timer input (gtm.tim_0_muxin_3_i(13))	INT_O14 (mcan0.int_req_o(14))
Timer input (gtm.tim_1_muxin_3_i(13))	
Timer input (gtm.tim_2_muxin_3_i(13))	
Timer input (gtm.tim_0_muxin_4_i(13))	INT_O15 (mcan0.int_req_o(15))
Timer input (gtm.tim_1_muxin_4_i(13))	
Timer input (gtm.tim_2_muxin_4_i(13))	

23.7.6 Interrupt Control

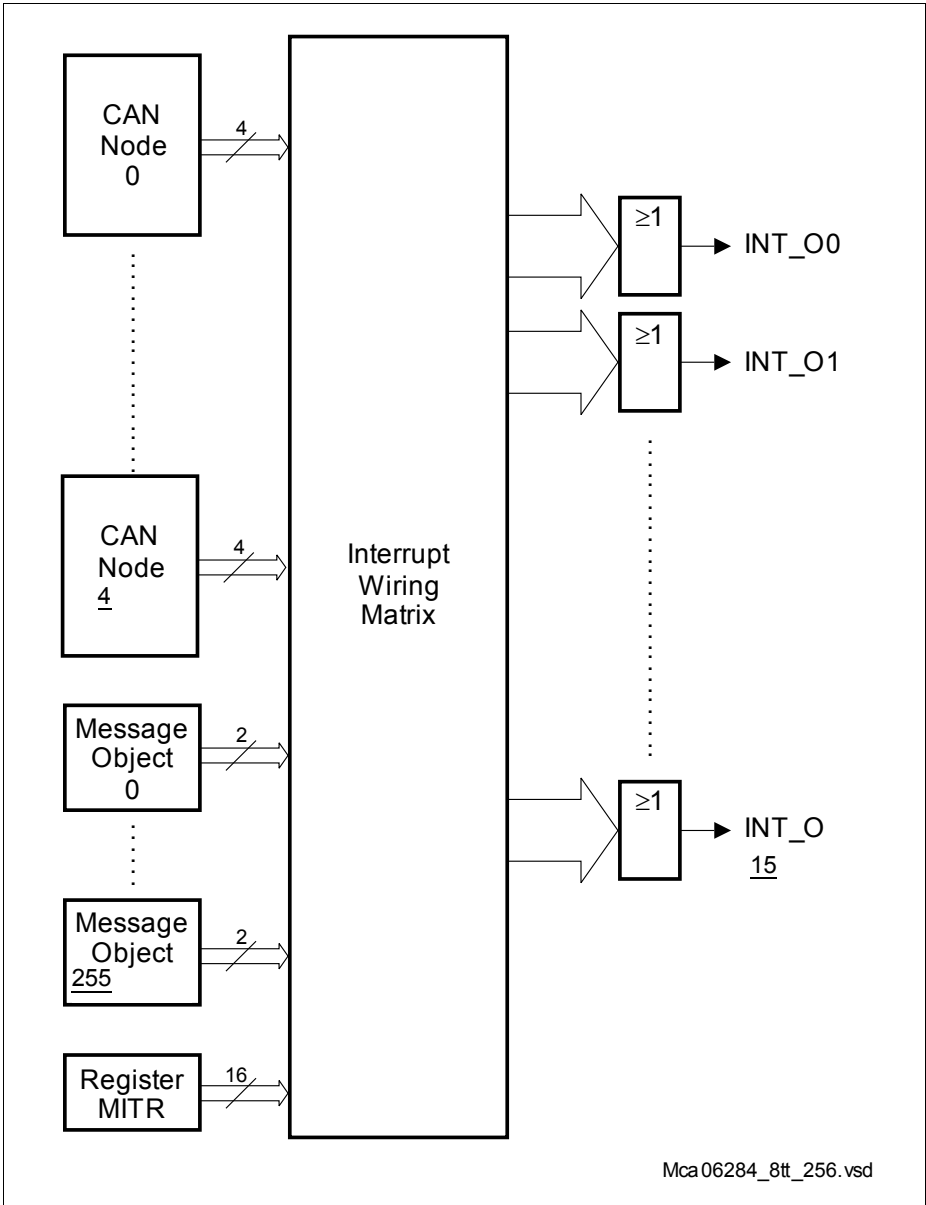
The interrupt control logic in the MultiCAN+ module uses an interrupt compressing scheme that allows high flexibility in interrupt processing. There are hardware and software interrupt sources available:

- CAN node interrupts:
 - Five different interrupt sources for each of the 5 CAN nodes = $5 * \underline{5}$ interrupt sources
- Message object interrupts:
 - Two interrupt source for each message object = $2 * \underline{256}$ interrupt sources
- One register (MITR) to initiate 16 interrupts via software

Each of the hardware initiated interrupt sources is controlled by a 4-bit interrupt pointer that directs the interrupt source to one of the 16 interrupt outputs INT_Om ($m = 0\text{-}15$). This makes it possible to connect more than one interrupt source (between one and all) to one interrupt output line. The interrupt wiring matrix shown in **Figure 23-31** is built up according to the following rules:

- Each output of the 4-bit interrupt pointer demultiplexer is connected to exactly one OR-gate input of the INT_Om line. The number “m” of the corresponding selected INT_Om interrupt output line is defined by the interrupt pointer value.
- Each INT_Om output line has an input OR gate which is connected to all interrupt pointer demultiplexer outputs which are selected by an identical 4-bit pointer value.

Controller Area Network Controller (MultiCAN+)



Mca06284_8tt_256.vsd

Figure 23-31 Interrupt Compressor

Controller Area Network Controller (MultiCAN+)**23.8 MultiCAN+ Soft Configuration**

There is one property of the MultiCAN+ module that can be configured via external signals. The MultiCAN+ module reads the configuration of these properties from PERCFG register.

- CAN FD on/off for the device is configured in PERCFG.CANFDEN (which is bit 8 of the corresponding register).

MultiCAN+ Soft Configuration can only restrict the module. That means that only the numbers equal or smaller than the number of physically available resources in the module may be used. Otherwise, an unpredictable behavior occurs.

23.9 Revision history

External revision history only shows major changes of MultiCAN+ module versions.

External Revision History (from MultiCAN+ v3.0 to MultiCAN+ v3.1)

- As per review meeting on 4Sep12 with designer, Removed for below:
 - Section 1.1.3.5 on Intermission removed as same as CAN V2.0B Bosch specs, under Section 9.1 Protocol Modifications, point 2.
 - Section 1.1.3.5 on Suspend transmission removed as same as CAN V2.0B Bosch specs, under Section 3.1.5 Interframe Spacing, Suspend Transmission.
 - Section 1.1.4.1 on Resynchronization removed as same as CAN V2.0B Bosch specs, under Section 8 Bit timing requirements, Resynchronization.
 - Section 1.1.4.1 on Synchronization rules, removed point 1,2 and 4. Point 3 and 5 are kept as new requirements in CAN FD.
 - Section 1.5.2 on Integrating mode, removed as same as CAN V2.0B Bosch specs, under Section 2 Sleep Mode / Wake-up.
- As per review meeting on 4Sep12 with designer, Added for below:
 - Additional text under MOFCRn.DLC, “Also if DLC>8 is programmed for message transmission, message transmit will be disabled.”
- Added at Section 1.6.4 OCDS suspend, Note on “During suspend mode, kernel registers may be accessed only when the kernel clock is running”

External Revision History (from MultiCAN+ v3.1 to MultiCAN+ v3.2)

- As per AURIX JF on 1oct12, after bosch meeting on discussion of baudrates, request for below:
 - Modified on Section 1.1.3.1 on Fig1-2 CAN FD data frames changed to 64 data bytes and removed footnote on “Current design supports up to 8 data bytes payload only”
 - Removed on Section 1.3.1 footnote on “supports message transfer of data length code 0 to 8 data bytes with no Transmitter Delay compensation”
 - Removed on Section 1.5 “8 data bytes, no Transmitter Delay compensation “
 - Modified on Section 1.5.1, table 1-3, point 4 on Transmitter Delay compensation from “not implemented” to “implemented”
 - Removed on Section 1.6.8.1 on CAN Restricted operation mode.
 - Modified on MOFCR.DLC bit to support for more than 8 data bytes, have included 12, 16, 20, 24, 32, 48, 64 data bytes option.
 - Added Section 1.5.2 Transmitter Delay Compensation and FNBTRx.TDC, FNBTRx.TDCO, FNBSTATx.TDCV.
-

External Revision History (from MultiCAN+ v3.2 to MultiCAN+ v3.3)

- Modified MultiCAN+ clocking from SCU for below

Controller Area Network Controller (MultiCAN+)

- Various diagrams and text that refers “ $f_{MULTICAN}$ ” to “ f_{ASYN_CAN} ” and “ f_{SYN_CAN} ”. Also added Section 1.13.3 MultiCAN+ Clock Interconnects with SCU
- Section 1.6.2 Table 1-5 footnote changed from “To guarantee the minimum operating frequencies when low power mode is enabled, fCLC is switched in fMULTICAN to a operating clock source, when SCU register CCUCON.” to “To guarantee the minimum operating frequencies when low power mode (pretended networking mode) is enabled (i.e when frequency of fSPB is reduced), fSYN_CAN and fCLC are switched from fSPB to fCAN when SCU register CCUCON0.LPDIV>0.”
- Renamed Fast Node x Bit Timing Status Register to Node x Transmitter Delay Compensation Register and moved FNBTRx.TDC, FNBTRx.TDCO to NTDCRx.TDC, NTDCRx.TDCO.

External Revision History (from MultiCAN+ v3.3 to MultiCAN+ v3.4)

- As per mail on “Widening of NBTR register field while maintaining compatibility and more” modified for below:
- - Rearranged FNBTRx register bits to use same layout as NBTRx for lowest 16bits and empty for upper 32bits
 - Changed address of FNBTRx to 0x0238 and NTDCRx to 0x23C
 - Swapped on NTDCRx register bits on TDC =Bit 15 and TDCO= Bits 11-8
 - Added NBTEVRx, Node Bit Timing Extended View Register. NBTEVRx view applies when NCRx.FDEN=1 and NBTRx view applies when NCRx.FDEN=0.
 - NBTEVRx. TSEG1 is 6 bits, valid values from 2 to 63
 - NBTEVRx. TSEG2 is 4 bits, valid values are 1 to 15
 - NBTEVRx.SJW is 4 bits, valid values are 1 to 15

External Revision History (from MultiCAN+ v3.4 to MultiCAN+ v3.5)

- As per review feedback on V3.3 CANFD specs review modified for below:
 - Removed on Section 1.1.3.1 text on point 2e “In current CAN FD implementation, up to 8 data bytes payload are supported only”
 - Added on Table 1-4 Minimum Operating Frequencies, footnote “To guarantee the minimum operating frequencies when low power mode.”
 - Removed on Section 1.4.12.4 “vice versa”
 - Update of Table 1-24 MultiCAN+ Clock Interconnects
- Additional modifications for below:
 - Added on NCRx.FDEN footnote on “Message transmission in Long & Fast CAN FD Frames. Message Reception according to CAN FD Protocol Specification, (i.e Able to Receive Classical CAN Format Frames ISO11898-1 and Long + Fast CAN FD Frames)”
 - Added on Table 1-8 Coding of DLC in CAN FD and MOFCRx.DLC

Controller Area Network Controller (MultiCAN+)

- Added MOFCR.FDF, MOFCR.BRS bit to indicate message format reception when MOFCRx.MOFM =1 (CAN FD enabled) (i.e received frame in classical CAN or CAN FD format)
- Added on NSRx.FLEC, fast last error code to indicate the error code during the fast bit rate.
- Added note on Table 1-13 Encoding of LEC bit field, “When a frame in CAN FD format reached data phase with BRS flag set, the next CAN event (error/valid frame) will be shown in FLEC instead of LEC. An error in a fixed stuff bit of CAN FD CRC will be shown as Form and not Stuff Error.”

External Revision History (from MultiCAN+ v3.5 to MultiCAN+ v3.6)

-
-

External Revision History (from MultiCAN+ v3.6 to MultiCAN+ v3.7)

- As per mail on “Message Mode proposal for CAN FD 64 data bytes” modified for below.
 - Added [Section 23.4.12.10](#) on CAN FD 64 byte Message Mode
 - Added [CAN_EMOzDATA0 \(z = 0-255\)](#) Extended Message Object Data Register and on Table 1-11
 - Modified on MOFCR.MMC bit 101 to CAN FD 64 bytes message mode.
- As per TC24x_tag_coverage.xls added tags AURIX_GEN1:5246, 5396, 5373, 5374:
- As per CAN FD weekly meeting on 21Nov12, modified for MOFCR.BRS and MOFCR.FDF bit from rh to rwh.

External Revision History (from MultiCAN+ v3.7 to MultiCAN+ v3.8)

- Added note on [Section 23.4.6.5](#) “The transmit request bits (i.e NTATTx.TXMO / NTBTTx.TXMO / NTCTTx.TXMO) in the timer control register of a node is able to trigger transmit request (MOSTATn.TXRQ) of message object in a list belonging to any other CAN nodes”.
- As per MultiCAN+ V3.7 feedback review
 - Modified on CAN_NBTEVRx text to “NBTEVRx contains all parameters to setup CAN bit timing for Nominal Bit Rate”.
 - Modified on CAN_FNBTRx text to “FNBTRx contains all parameters to setup CAN bit timing for Data Bit Rate”.
 - Modified on CAN_FNBTRx register bits name to FTSEG1, FTSEG2, FSJW and FBRP.
 - Modified on CAN_NBTEVRx.TSEG2 to 5bits, bit20 to bit16.

Controller Area Network Controller (MultiCAN+)
External Revision History (from MultiCAN+ v3.8 to MultiCAN+ v3.9)

- As per MultiCAN+ V3.7 feedback review.
 - As per mail “CCE protection of register tdcrc” added on CAN_NTDCR register “NTDCRx register can be written only if bit NCRx.CCE is set.”. Removed on NTDCR.TDC text on “this bit is CCE and INIT protected”.
 - Added on soft configuration section a footnote on CAN FD “When CAN FD is disabled by PRDCFG, access to CAN FD registers do not generate a bus error. To verify CAN FD is disabled, try to set NCRx.FDEN bit and read back the NCRx register values, the NCRx.FDEN bits remain cleared.”
 - Modified on NTDCR.TDCV text from fcan/fclc to time quanta “Valid values for TDCV are 0 to 31 times of time quanta t_Q ”
 - Modified on NTDCR.TDCO text from fcan/fclc to time quanta “Valid values for TDCO are 0 to 15 times of time quanta t_Q .”
 - Added text under Section CAN FD 64 bytes Message Mode “When MSGVAL of the base object is cleared, an ongoing data transfer to TOP and BOT objects are aborted, thereby freeing objects BOT and TOP. Clearing MSGVAL bits of BOT, TOP objects and the extra message objects do not stop an ongoing data transfer and are not considered at all.”
 - Added text on NCRx.FDEN, MOFCR.MOFM, MOFCR.BRS and MOFCR.FDF bits “Please see Table 1-19 CAN FD Transmit and Receive Behavior.”

External Revision History (from MultiCAN+ v3.9 to MultiCAN+ v3.10)

- As per MultiCAN+ V3.7 feedback review
 - Updated Table 1-19 CAN FD Transmit and Receive Behavior, corrected for transmit behavior “Classical CAN Frames” FDEN from 0_B to X_B .

External Revision History (from MultiCAN+ v3.10 to MultiCAN+ v3.11)

- As per MultiCAN+ V3.7 feedback review
 - Updated Table 1-30 MultiCAN+ I/O control Selection and Setup and Table 1-36 Receive Input Selection
 - Added Table 1-9 “Minimum Operating Frequencies for CAN FD [MHz]”
- Updated on Section 1.6.2 Clock Control on example of equation 1.1 to “As an example, when $f_{CLC} = 10\text{MHz}$, $f_{CAN} = 20\text{MHz}$, No of active CAN nodes =2, $\text{Baudratemax} = [(8 \times 50\text{ns}) + (8 \times 100\text{ns}) + (4 \times 2 \times 100\text{ns})] = 2000\text{ns} = 500 \text{KBaud}$ ”

External Revision History (from MultiCAN+ v3.11 to MultiCAN+ v3.12)

- Modified on Section 1.1.3.2 Remote Frames, paragraph from “Remote frames is only defined in CAN format, Remote frames and RTR bit do not exist for CAN FD format.”

Controller Area Network Controller (MultiCAN+)

In a CAN FD node, Remote frames request are always replied back in CAN format and will never receive replies in CAN FD format.” to “Remote frames is only defined in CAN format, Remote frames and RTR bit do not exist for CAN FD format. Remote frame requests do not change the format of the preconfigured reply data frames (i.e FDF and BRS bits of preconfigured data frames are not changed on remote frame requests, reply data frames may be in CAN base/extended or CAN FD base/extended).”

- Modified on Table 1-20 CAN FD Transmit and Receive Behavior on receive behavior to FDEN=0, MOFM=X, BRS=0/1, FDF=0/1- Classical CAN Frames, FDEN=1, MOFM=X, BRS=0/1, FDF=0/1- Classical CAN Frames, Long Frames, Long + Fast Frames.
- Added on Table 1-20 CAN FD Transmit and Receive Behavior, footnote on receive behavior “Please note that Remote Request Frames do not change BRS and FDF bits.”

External Revision History (from MultiCAN+ v3.12 to MultiCAN+ v3.13)

- Modified for below
 - Added on MCR.CLKSEL bit option 0100_B E-Ray clock “(Feray)”.
 - Added Fig1-52 MultiCAN+ Interconnects with SCU on section 1.13.3 MultiCAN+ Clock Interconnects With SCU.
-

External Revision History (from MultiCAN+ v3.13 to MultiCAN+ v3.14)

- Removed NSRx.RESI bit would generate interrupt on NSRx.ALERT and NCRx.ALIE. Also modified NSRx.RESI bit to ‘rh’ changed description to “This bit is an error flag that is set when the ESI flag in a received CAN FD frame is set.” and “0_b-Last received CAN FD message did not have its ESI flag set. 1_b-Last received CAN FD message had its ESI flag set.”
- Addition to note on Table 1-13 Encoding of LEC bit field, “Correspondingly CAN event (error/valid frame) will be shown back at LEC after the first bit of the CRC delimiter when CAN FD switches from Data bit rate to Nominal bit rate”.

External Revision History (from MultiCAN+ v3.14 to MultiCAN+ v3.15)

- Removal of MOFCR.MOFM bit. Update of Table 1-20 CAN FD Transmit and Receive Behaviour.
- Update of Table 1-5 Minimum Operating Frequencies for CAN FD [MHz].
 - Addition of Acceleration Factor rows for values from 1 to 8
 - Modified on Footnote 3) with addition of Base Nominal Bit Rate is 1Mbit/s and addition “when several nodes operate at different acceleration factor A.F, only the node operating the highest acceleration factor need to be considered for the required minimum operating frequency requirement.”

Controller Area Network Controller (MultiCAN+)**External Revision History (from MultiCAN+ v3.15 to MultiCAN+ v3.16)**

- Section 1.6.12.4 Message Object Format updated with removal of MOFCR.MOFM and replaced with description of NCRx.FDEN and MOFCRn.FDF, MOFCRn.BRS bits.
- CAN FD feature added for TC27x C-step, TC26x B-step, TC29x B-step.
- Updated on features list, with bullet item “Supports an acceleration factor of 8 when operating in CAN FD mode per Bosch CAN FD V1.0 April17th 2012”

External Revision History (from MultiCAN+ v3.16 to MultiCAN+ v3.17)

-
- Updated Table 1-5 Minimum Operating Frequencies for CAN FD [MHz], with corrected values from multican_performance_canfd_16apr13.xls
 - Modified Acceleration Factor rows for values from 1 to 5 and at various places that mentions AF to 1 to 5.

External Revision History (from MultiCAN+ v3.17 to MultiCAN+ v3.18)

- Added additional text on Section 1.5.2 Transmitter Delay Compensation to describe more details on secondary sample point implementation.
- Added on footnote of Table 1-5 Minimum Operating Frequencies for CAN FD “When message objects are configured as transmit or receive FIFO structures i.e MOFCRn.MMC = 0001/0010, only the base object of 1 is counted towards the minimum frequency requirement, all other slave objects on the FIFO do not count towards the minimum frequency requirement. See Section 1.6.12.5.
- Relabel tags CAN_NUMBER_NODES to CAN1_NUMBER_NODES on Table1-2.

External Revision History (from MultiCAN+ v3.18 to MultiCAN+ v3.19)

- Text enhancements below:
 - Under Section 1.1.3.1, abbreviation added for FDF, BRS, ESI bit
 - Under Section 1.1.3.3, amended from “shall switch back” to “will switch back”
 - Under Section 1.3, amended from “on every nodes” to “on all nodes”
 - Under Section 1.3.1, amended from “measured Transmitter Delay” to measured loop delay”
 - Under CAN_MOFCR on CAN FD Transmit and Receive Behavior, added text “Message transmission and reception behavior of CAN data frames is summarized at Table 1-20 below.”
 - Under CAN_MOFGPR.TOP/BOT added text MOFGPR.BOT points to data byte 8-35 and MOFGPR.TOP points to 36-63.

External Revision History (from MultiCAN+ v3.19 to MultiCAN+ v3.20)

- All reference to EDL is renamed to FDF, register bit MOFCR.EDL is renamed to MOFCR.FDF, functionality remains the same.

Controller Area Network Controller (MultiCAN+)**External Revision History (from MultiCAN+ v3.20 to MultiCAN+ v3.21)**

-

External Revision History (from MultiCAN+ v3.21 to MultiCAN+ v3.22)

- Changed CAN FD descriptions to customer relevant information.

External Revision History (from MultiCAN+ v3.22 to MultiCAN+ v3.23)

- Corrected spelling bugs

External Revision History (from MultiCAN+ v3.23 to MultiCAN+ v3.24)

- Changes for XMC device frequencies, not applying for AURIX devices

External Revision History (from MultiCAN+ v3.24 to MultiCAN+ v3.25)

- Changes for XMC port naming. Correcting XMC specification by removing AURIX relevant issues.

External Revision History (from MultiCAN+ v3.25 to MultiCAN+ v3.26)

- Corrected memory space for XCM1000.
- Fixed pad naming for all devices.
- CLC and FDR are existing on all modules, which are existent in one microcontroller.

External Revision History (from MultiCAN+ v3.26 to MultiCAN+ v3.27)

- Corrected interrupt mapping for XCM1000.

External Revision History (from MultiCAN+ v3.27 to MultiCAN+ v3.28)

- CRC issue of CAN FD documented.

External Revision History (from MultiCAN+ v3.28 to MultiCAN+ v3.29)

- MCR register, formula for clock switching mathematically not correct. As no clock cycles were calculated. Exchanged to clock cycles..

24 Single Edge Nibble Transmission (SENT)

This chapter describes the SENT Interface of the TC26x B-Step. It contains the following sections:

- Functional description of the SENT kernel (see **“Overview” on Page 24-2**)
- SENT kernel register descriptions (see **“SENT Kernel Registers” on Page 24-22**)
- TC26x B-Step implementation-specific details and registers of the SENT module (port connections and control, interrupt control, address decoding, and clock control, see **“SENT Module Implementation” on Page 24-70**)

24.1 SENT Kernel Description

Figure 24-1 shows a global view of the SENT interface.

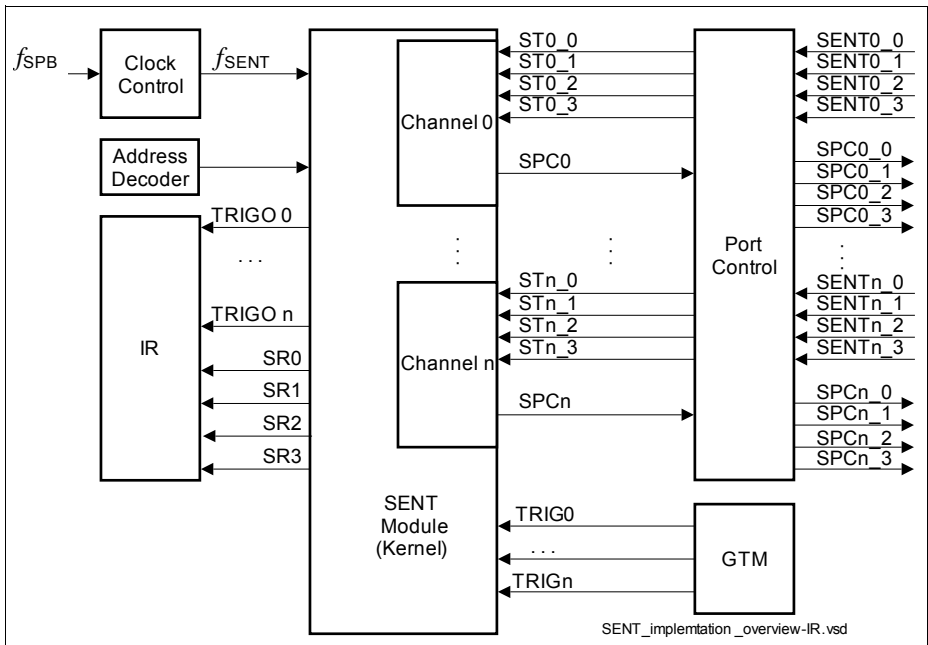


Figure 24-1 General Block Diagram of the SENT Interface

The SENT module communicates with the external world via one I/O line for each channel. The STx lines are the receive data input signals. They can overlay ADC inputs. If the optional SPC mode is used, they can be used on a port configured with an open drain transistor. This way the optional SPC data can be transmitted and the line is used bidirectionally. In case of an external transceiver, receive and transmit path can be routed to two different ports.

Single Edge Nibble Transmission (SENT)

24.1.1 Overview

The SENT interface provides a serial communication link typically used to connect sensors or other peripheral devices.

Clock control, address decoding, and service request control are managed by the SENT module kernel.

The SENT IP-module performs communication according to the SENT specification J2716 JAN2010.

While staying compliant to this standard, it is able to cover as well the Short PWM Code (SPC) protocol extensions. This enhances the standardized SENT protocol defined by J2716 JAN2010. SPC enables the use of enhanced protocol functionality like “synchronous”, “range selection” and “ID selection” protocol mode.

Receive data on a SENT channel can be set up according to the underlying application. In particular the number of nibbles forming one value is configurable.

The message storage consists of two 32-bit registers for each channel, representing a flexible double buffer system.

In SPC mode, maintaining the sample and transmission schedule as well as providing message status information is support.

The register set of the SENT module can be accessed directly by the CPU for configuration, data read out and status query.

The SENT IP-module supports the following features:

Single Edge Nibble Transmission (SENT)

Features

- Conformance with SENT protocol specification J2716 JAN2010
- Data rates of up to 65,8 kbit/s at 3 μ s tick length and 6 data nibbles on each channel
- Support of standard tick times (3 μ s through 90 μ s) and
- Message tick time programmable between 0.2 μ s and 90 μ s
- 6 SENT channels working independently in parallel
- Status nibble optionally included in the checksum (default not included)
- Sticky interrupt flags, error interrupt optional (default disabled)
- Configurable frame length (default is 24 bit), max data size is 32 bits
- Serial data processing optional (default: disabled)
- Option for bigger frame lengths (must still be fix for each application)
- transparent mode (nibble CRCs are written to the receive control register for SW processing)
- Support of SPC
- Support of trailing Pause Nibble of any length (even longer than 70 ticks)
- Indication of system status: STOP, INITIALIZED, RUNNING, SYNCHRONIZED
- The receiver module will monitor the message for the following error conditions:
 - Calibration pulse length deviates more than +/-25% from the nominal 56 ticks
 - Too many or too few nibbles between calibration pulses.
 - Checksum error.
 - Successive calibration pulse differ by more than 1.5625%
 - Any nibble data values measured as < 0 or >15.
- When any of those errors is detected, the receiver module shall declare that a message error has occurred and ignore the entire message.
- Any of those errors shall cause the receiver to begin searching for a valid calibration pulse to re synchronize.
- Option to enable/disable the check of the next calibration pulse before validation of received data
- Digital Glitch filter suppressing noise
- Buffer overrun detection
- Optional output inversion for use of external open drain transistor
- Optional input inversion for use of external open transistor for level shifting
- Interrupt on status nibble violation
- Programmable Nibble sorting to support LSN or HSN first and relief CPU
- Time stamp generation
- Watch Dog on incoming frames

Single Edge Nibble Transmission (SENT)

24.1.2 General Operation

The Single Edge Nibble Transmission encoding scheme (SENT) is intended for use in applications where high resolution sensor data needs to be communicated from a sensor to an Engine Control Unit (ECU). It is intended as a replacement for the lower resolution methods of 10 bit A/D converters and PWM and as a simpler low cost alternative to CAN or LIN. The implementation assumes that the sensor is a smart sensor containing a microprocessor or dedicated logic device (ASIC) to create the signal.

Figure 24-3 shows a typical TC26x B-Step application in which a SENT interface reads a sensor device.

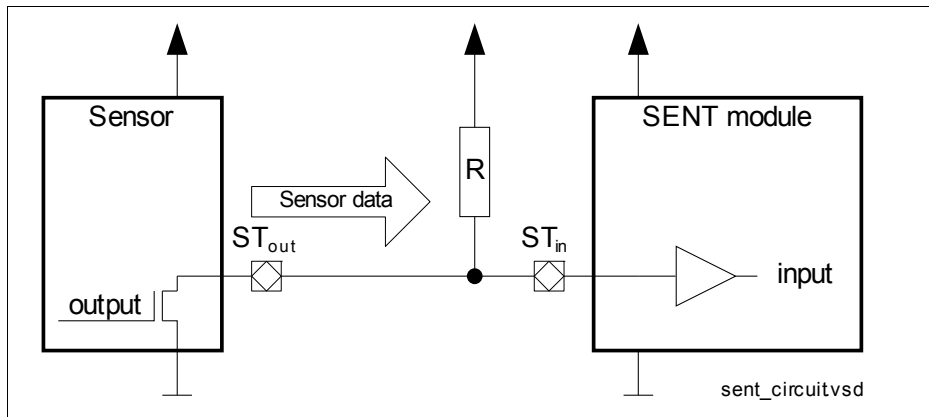


Figure 24-2 SENT to External Device Connection

SENT communication is unidirectional from sensor to controller without any synchronization. The sensor signal is transmitted as a series of PWM blocks measured as falling to falling edge times.

The Short PWM Code (SPC) extension overcomes the drawback of unidirectionality as said above while keeping conformance to the standard.

Figure 24-3 shows a typical TC26x B-Step application in which an SPC enabled SENT ECU reads an SPC enabled SENT sensor device.

Single Edge Nibble Transmission (SENT)

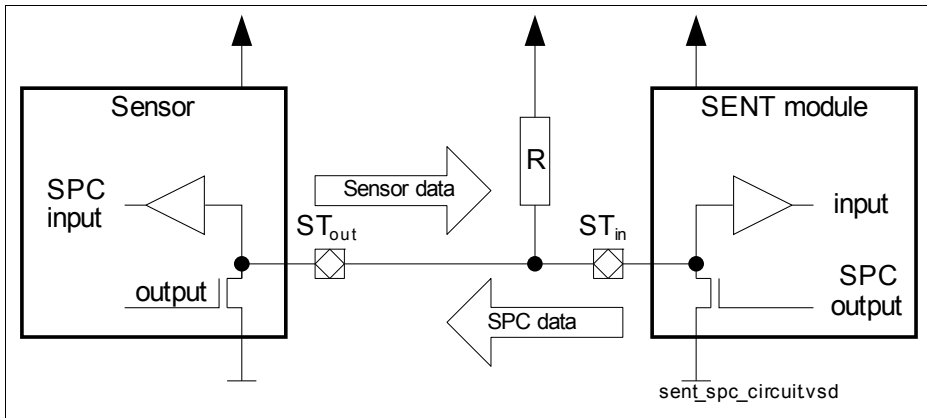


Figure 24-3 SENT SPC to External Device Connection

Some applications are:

- Read out of the external throttle position sensor (e.g. SENT enabled linear hall sensor)
- Receiving pedal position
- Synchronize sampling and read out of up to four sensors on one single line (SPC)
- Selection of 1 out of 4 sensors connected to a single SENT channel.
- Serial connections of the TC26x B-Step to other peripheral devices

24.1.2.1 Definitions

SENT: Single Edge Nibble Transmission

Nibble: Four Bit value between 0 and 15 = half a Byte = one character in hex (0 to F)

SPC: Short PWM Code

PWM: Pulse Width Modulation

ASIC: Application Specific Integrated Circuit

CAN: Controller Area Network

LIN: Local Interconnect Network

ISO: International Organization for Standardization

ECU: Electronic Control Unit

FSM: Finite State Machine

Single Edge Nibble Transmission (SENT)

24.1.3 Standard SENT Operation

Standard SENT Mode supports communication fully compliant to J2716 JAN2010, in which only the external device is sending and the ECU is receiving. In this unidirectional communication, both transmitter and receiver use the same data frame format and have the same baud rate. These settings are defined at system integration time and do not change for a given application. Data is received on pin ST. **Figure 24-4** shows the block diagram of the SENT Module when operating in Standard SENT Mode.

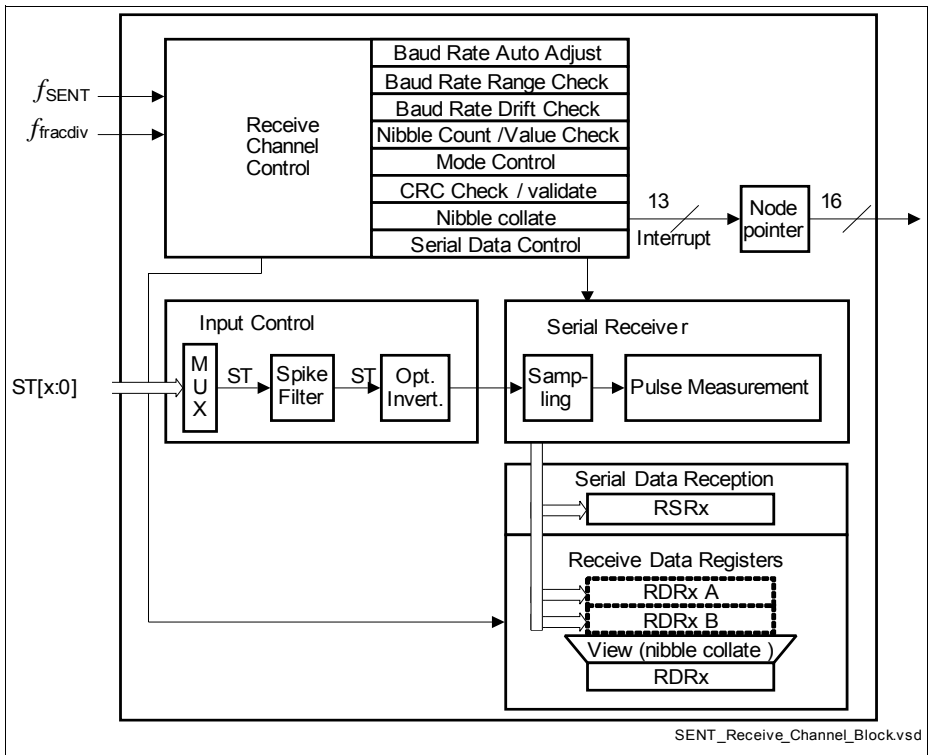


Figure 24-4 Standard SENT Mode Operation

Single Edge Nibble Transmission (SENT)

24.1.3.1 Frame Formats and Definitions

This section describes the frame formats and definitions of the SENT protocol.

Basic Definitions

Figure 24-5 shows the layout and definitions of a standard SENT frame. Note that the SENT standard does not specify whether the most significant nibble or the least significant nibble is sent out first.

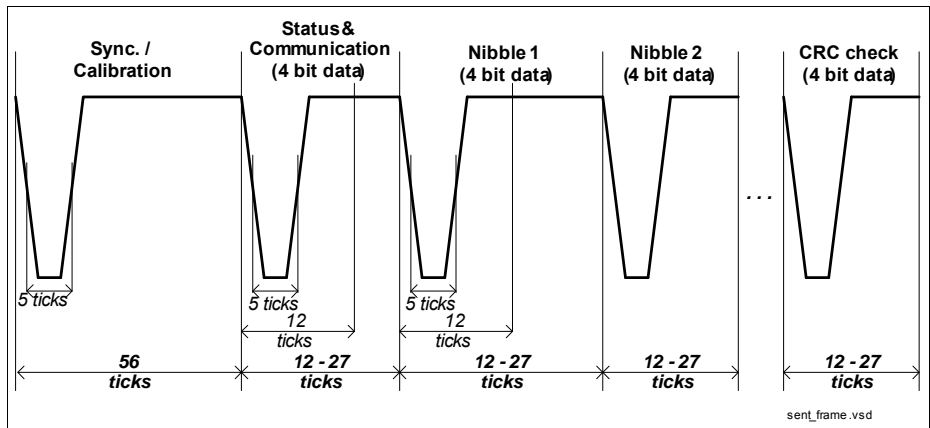


Figure 24-5 Standard Encoding Frame

Serial Data Encoding in Status and Communication Nibble

The Status and Communication nibble is used by sensors that transmit extra serial data such as part numbers or diagnostic information. Data bits from this nibble are constructed across multiple SENT messages to form a Short Serial Message or an Enhanced Serial Message. These messages are alternately referred to as “Slow Channel” data in the J2716 specification.

Table 24-1 Short Serial Data Encoding

Bit Number	Bit Function
0	Reserved for specific application
1	Reserved for specific application
2	Serial Data message bits
3	Message start = 1, otherwise = 0

Single Edge Nibble Transmission (SENT)

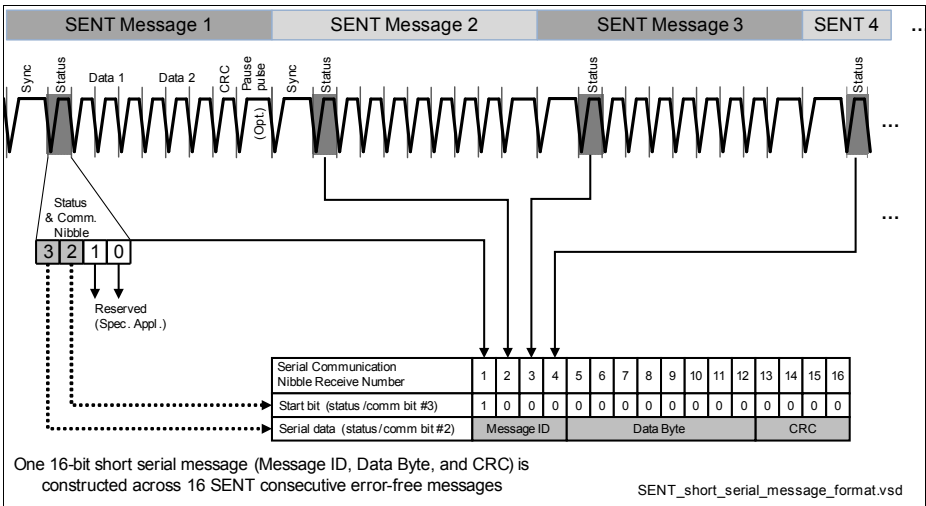


Figure 24-6 Short Serial Message, Serial Data Encoding over 16 messages

Short Serial Message Format

The SENT kernel supports the Short Serial Message Format defined in the J2716 specification. The Short Serial Message Format provides 16 bits of data, constructed across 16 SENT messages. Each of the 16 messages contains one bit of Short Serial Message data in Status and Communication nibble bit 2.

Serial data is communicated in a 16-bit sequence as shown in Table 24-2. The start of a Short Serial Message is indicated by a 1 in bit 3 of the Status and Communication nibble. That SENT message and the next 15 must be successfully received (no errors) for the Short Serial Message data to be received. The CRC generation is identical to the CRC generation on the data nibbles. Short Serial Message data is provided in the SDSx register.

Table 24-2 Short Serial Message Format

Serial Communication Nibble Receive No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Startbit (bit # 3)	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Serial Data (bit # 2)	Message ID			Data Byte									CRC			

CRC Calculation

SENT signal data and Short Serial Message data is secured with a 4 bit CRC. The standard CRC calculation method is defined in the J2716 JAN2010 specification.

Single Edge Nibble Transmission (SENT)

RCR.CRZ defines, if a zero nibble is added for calculation or not.

The alternative checksum nibble is a 4-bit CRC of the data nibbles (including the status nibble if RCR.SNI is set, as used in sensor TLE4998). The CRC is calculated using a polynomial $x^4 + x^3 + x^2 + 1$ with a seed value of 0101.

In order to facilitate CRC implementations and to avoid ambiguities, an example implementation of the alternative 4-bit CRC is given below. This is used e.g. in the sensor TLE4998 for the signal data. See [Figure 24-7](#) for details.

```
// Fast way for any µC with low memory and compute capabilities
// contains input data (status nibble, 6 data nibble, CRC)
char Data[8] = {...};
// required variables and LUT
char CheckSum, i;
char CrcLookup[16] = {0, 13, 7, 10, 14, 3, 9, 4, 1, 12, 6, 11, 15,
2, 8, 5};
CheckSum= 5; // initialize checksum with seed "0101"
for (i=0; i<7; i++) {
    CheckSum = CheckSum ^ Data[i];
    CheckSum = CrcLookup[CheckSum];
};
// finally check if Data[7] is equal to CheckSum
```

Note: For the “Enhanced Serial Message Frame Format”, an own 6-bit CRC calculation method is defined by the standard.

Single Edge Nibble Transmission (SENT)

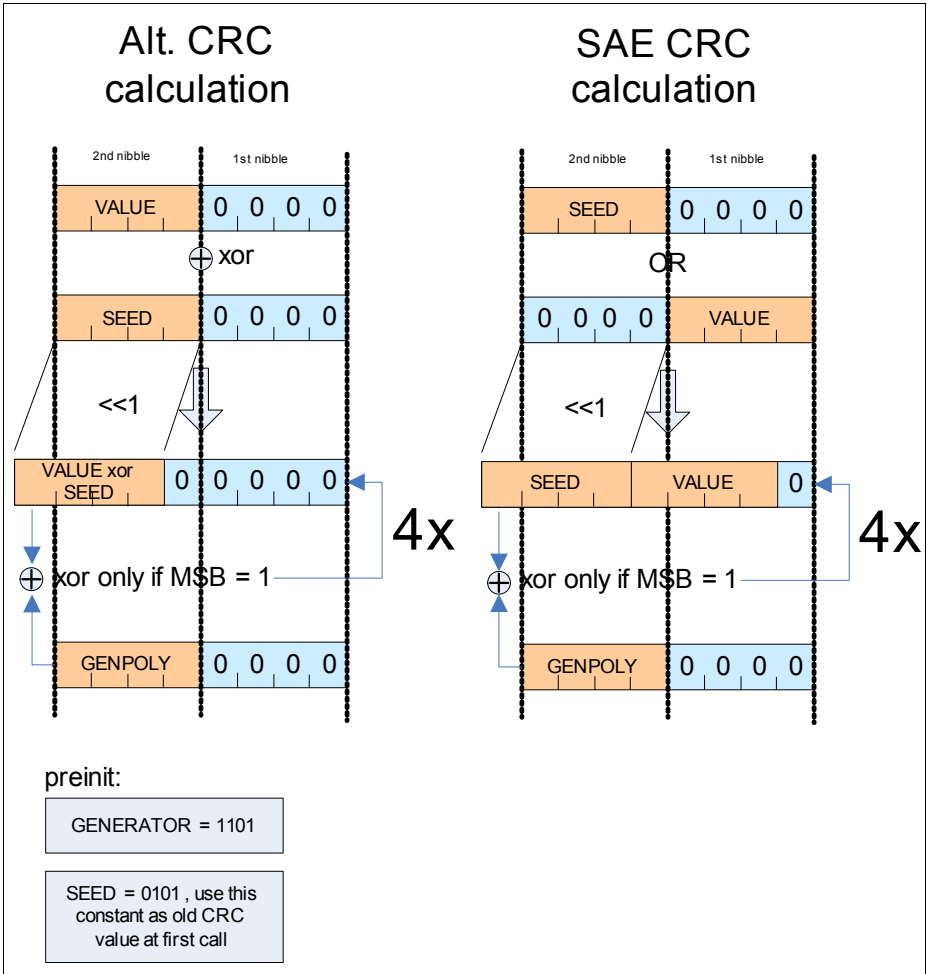


Figure 24-7 Alternate vs. SAE CRC Calculation

Single Edge Nibble Transmission (SENT)

Enhanced Serial Message Format

The SENT kernel supports the Enhanced Serial Message Format defined in the J2716 specification. The Enhanced Serial Message Format provides 21 bits of message data and a 6-bit CRC, constructed across 18 SENT messages. The serial data is transmitted bit wise per frame in bits [3:2] of the Status and Communication nibbles of 18 consecutive messages from the transmitter.

Table 24-3 Enhanced Serial Data Encoding

Bit Number	Bit Function
0	Reserved for specific application
1	Reserved for specific application
2	Serial Data message bits
3	Message start = 1 1 1 1 1 1 0

Enhanced Serial Message Frame

Serial data is communicated in an 18 frame sequence as shown in [Table 24-4](#). The start of an Enhanced Serial Message is indicated by the unique pattern “1111110” in bit #3 of the status and communication nibble, constructed across 7 SENT messages. In message 1 - 6 they are set to “1”. In message 7, 13 and 18 they are set to “0”. 18 consecutive SENT messages must be successfully received (no errors) for the Enhanced Serial Message data to be received. The 6-bit Enhanced Serial Message CRC is different from the 4-bit CRC on SENT (fast channel) messages. (See SENT Standard)

An Enhanced Serial Message contains 21 bits of message data and a 6-bit frame-check sequence. A configuration bit (serial data bit #3 of the 8th SENT message's Serial Communication Nibble) determines how the SENT module interprets the serial data.

- Configuration bit = 0: 12-bit data and 8-bit message ID, see [Table 24-5](#)
- Configuration bit = 1: 16-bit data and 4-bit message ID, see [Table 24-6](#)

Enhanced Serial Message data is provided in the SDSx register.

Table 24-4 Enhanced Serial Data Frame

Serial Communication Nibble Receive No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Serial Data (bit # 3)	1	1	1	1	1	1	0	C	8-bit ID (7-4)				0	8-bit ID (3-0) or 4-bit data			0	
Serial Data (bit # 2)	6-bit CRC						12-bit data field											

Note:

Single Edge Nibble Transmission (SENT)

13. Message 8 Status Nibble bit # 3 is Configuration Bit C that defines the Enhanced Serial Data format:

0: 12-bit data, 8-bit ID, see [Table 24-5](#)

1: 16-bit data, 4-bit ID, see [Table 24-6](#)

14. Messages 17-14 Status Nibble bit # 3 contain either the lower 4 bits (Bit 3-0) of 8bit ID or 4 additional data bits.

[Table 24-5](#) shows in detail the frame structure for Enhanced Serial Data Frames with Configuration Bit = 0.

Table 24-5 Configuration Bit = 0

Serial Communication Nibble Receive No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Serial Data (bit # 3)	1	1	1	1	1	1	0	0	8-bit ID (7-4)				0	8-bit ID (3-0)			0	
Serial Data (bit # 2)	6-bit CRC						12-bit data field											

[Table 24-6](#) shows in detail the frame structure for Enhanced Serial Data Frames with Configuration Bit = 1.

Table 24-6 Configuration Bit = 1

Serial Communication Nibble Receive No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Serial Data (bit # 3)	1	1	1	1	1	1	0	1	4-bit ID (3-0)				0	4-bit data (15-12)			0	
Serial Data (bit # 2)	6-bit CRC						12-bit data field											

Single Edge Nibble Transmission (SENT)

24.1.4 SPC Operation

The module supports a SPC (Short PWM Code) protocol, which enhances the standardized SENT protocol (Single Edge Nibble Transmission) defined by J2716 JAN2010. SPC enables the use of enhanced protocol functionality due to the ability to select between “synchronous”, “range selection” and “ID selection” protocol mode or even “bidirectional transmit mode”.

24.1.4.1 Synchronous Transmission

In the “synchronous” mode, the sensor (slave) starts to transfer a complete data frame only after a low pulse is forced by the master on the OUT pin. This means that the data line is bidirectional - an open drain output of the micro controller (master) sends the trigger pulse. The sensor then initiates a sync pulse and starts to calculate the new output data value. After the synchronization period, the data follows in form of a standard SENT frame, starting with the status, data and CRC nibbles. At the end, an end pulse allows the CRC nibble decoding and indicates that the data line is idle again. The timing diagram in [Figure 24-8](#) visualizes a synchronous transmission

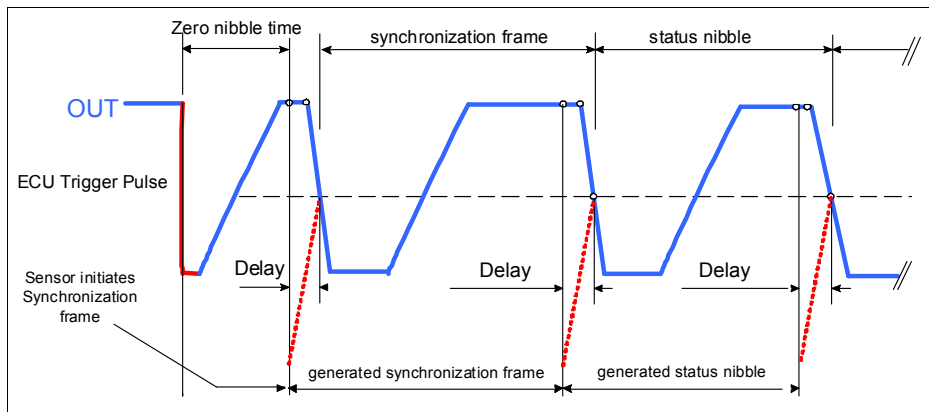


Figure 24-8 Synchronous Transmission (SPC)

24.1.4.2 Range Selection

The low time duration of the master can be used to select the range of the sensor in SPC dynamic range selection mode.

Single Edge Nibble Transmission (SENT)

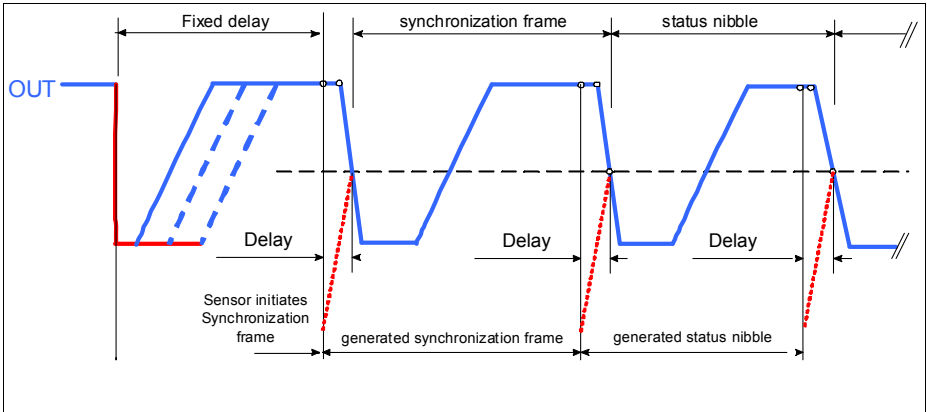


Figure 24-9 Range (SPC)

24.1.4.3 ID Selection

This functionality is similar to the previous mode, but instead of switching the range of one sensor, one of up to four sensors are selectable on a bus (bus mode, 1 master with up to 4 slaves). This allows parallel connection of up to 4 sensors using only three lines (VDD, GND, OUT), as illustrated in [Figure 24-10](#).

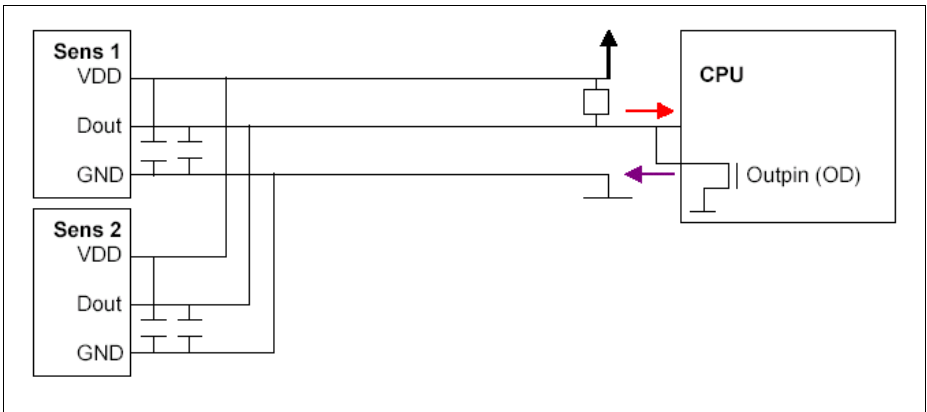


Figure 24-10 ID Selection (SPC)

Single Edge Nibble Transmission (SENT)

24.1.4.4 Bidirectional Transmit Mode

After addressing the sensor the ECU interrupts the sensor sync pulse by pulling down the line. The ECU starts to transmit own nibbles on the time base of the selected sensor. The ECU has to adapt the sensors timing from an earlier received cycle.

The sensor detects the falling edge that infringes the protocol and switches to receive mode as illustrated in [Figure 24-11](#).

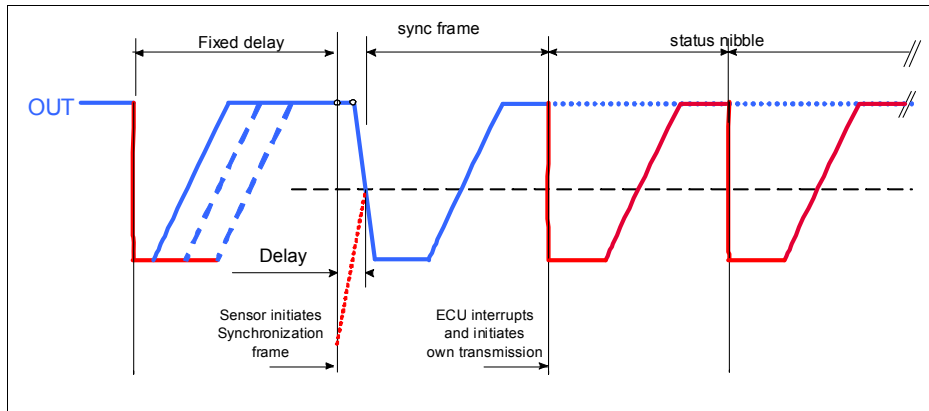


Figure 24-11 Bidirectional Transmit Mode (SPC)

24.1.4.5 SPC Timing

An SPC transmission is initiated by a Master pulse on the OUT pin. To detect a low level on the OUT pin, the voltage must be below a threshold V_{thf} . The sensor detects that the OUT line has been released as soon as V_{thr} is crossed. [Figure 24-12](#) shows the timing definitions for the master pulse. The master low time t_{mlow} as well as the total trigger time t_{mtr} are individual for the different SPC modes and are given in the sensors specifications. It is recommended to choose the typical master low time exactly between the minimum and the maximum possible time given by the connected sensor: $t_{mlow,typ} = (t_{mlow,min} + t_{mlow,max}) / 2$.

Single Edge Nibble Transmission (SENT)
24.1.5 Baud Rate Generation

The nominal baud rate of each channel is individually adjustable. This is required as it is depending from the connected sensor and its current deviation from nominal frequency.

The actual baud rate of the channel follows automatically the baud rate of the connected device and its current deviation from nominal frequency. The adaptation range is +- 25% as specified in J2716 JAN2010.

Chapter 24.1.5 shows in detail how the baud rate for each channel is adjusted. $1 / f_{\text{tick}_x}$ defines the resulting tick length.

In the first stage, a global fractional divider serves as pre divider. Its intermediate frequency f_{fracdiv} can be set up so that it is handy as input frequency for the different SENT channels. The clock signal f_{pdiv} of a channel must always be at least 20 times the nominal tick frequency of the channel. This is to allow for the highly flexible and big tolerance of +-25% versus the sending device. In addition the module must be able to detect a deviation in the length of 2 consecutive Synchronization / Calibration pulses of 1.5625% (1/64) and adapt the own baud rate. The tick time is typically 3 μs but can vary by standard and take values up to 10 μs . Future application might require even shorter tick times for higher repetition rates. This is why this implementation allows for an even extended range of 0.2 ... 90 μs . To achieve this each channel has its own fractional divider. This allows to downscale the frequency of f_{tick} precisely to the required tick frequency. For details on the principles of a fractional divider, please refer to the SCU chapter of TC26x B-Step section "Clock Control".

The SENT module provides two clock signals to the channels (**Figure 24-13**):

- f_{SENT}
This is the module clock that is used inside the SENT kernel for control purposes such as clocking of control logic and register operations. The frequency of f_{SENT} is always identical to the system clock frequency f_{SYS} . The clock control register SENT_CLC makes it possible to enable/disable f_{SENT} under certain conditions.
- f_{fracdiv}
This clock is the module clock that is used inside the SENT kernel for baud rate generation of the serial channels. The fractional divider register SENT_FDR controls the frequency of f_{fracdiv} . Usually not required and set to 1.

The channels generate two local clock signals:

- f_{pdiv_x}
This clock is the channel clock that is used inside the SENT channel x for baud rate generation of the serial channel. The divider register SENT_CPDRx controls the frequency of f_{pdiv_x} .
- f_{tick_x}
This clock is the channel clock that is used inside the SENT channel as the baud rate frequency. The fractional divider register SENT_CFDRx controls the frequency of

Single Edge Nibble Transmission (SENT)

f_{tick_x} . The higher the value of DIV the higher the precision with which the Synchronization / Calibration pulse and the deviation in the length of 2 consecutive Synchronization / Calibration pulses (drift) can be measured.

DIV must be set in an interval of [560, 3276].

Each time a Synchronization / Calibration pulse starts DIVM is measured. DIVM is the result of measuring the calibration pulse duration with f_{pdiv_x} . At the end of the Synchronization / Calibration pulse this value is taken as new divider of the CFDR.

Each time a pulse starts, the internal accumulator of the CFDR is preset with $\text{DIVM} / 2$. This is required to center the data eye and to make the margin symmetrical.

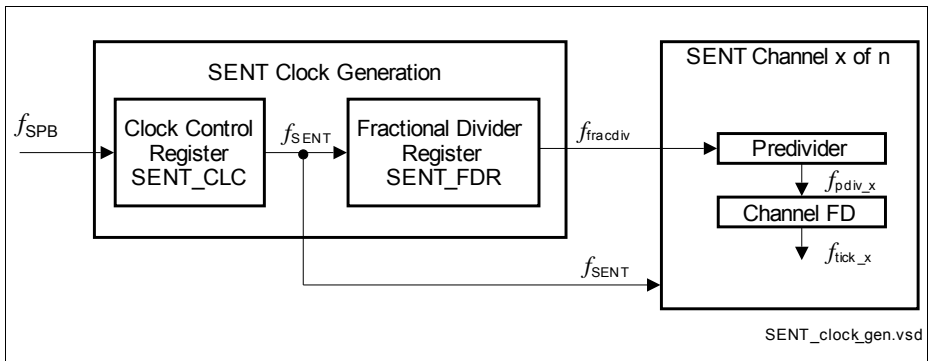


Figure 24-13 SENT Module Clock Generation

The following two formulas define the frequency of f_{fractdiv} :

$$f_{\text{fractdiv}} = f_{\text{SENT}} / (1024 - \text{SENT_FDR.STEP}); \text{FDR.DM} = 01_{\text{B}} \quad (24.1)$$

$$f_{\text{fractdiv}} = f_{\text{SENT}} \times \text{SENT_FDR.STEP} / 1024 \text{ with STEP} = 0 \dots 1023; \text{FDR.DM} = 10_{\text{B}} \quad (24.2)$$

The following formula defines the frequency of f_{pdiv_x} :

$$f_{\text{pdiv}_x} = f_{\text{fractdiv}} / (\text{SENT_CPDRx.PDIV} + 1) \quad (24.3)$$

The following formula defines the nominal frequency of f_{tick_x} . For the actual frequency of f_{tick_x} DIV is replaced by DIVM after the current sensor frequency was validly measured.

$$f_{\text{tick}_x} = f_{\text{pdiv}_x} \times 56 / \text{SENT_CFDRx.DIV} \text{ with DIV} = 560 \dots 3276 \quad (24.4)$$

Single Edge Nibble Transmission (SENT)

24.1.6 Error Detection Capabilities

Each SENT channel can detect and signal the following error conditions:

Protocol Level:

- Calibration pulse length deviates more than +/-25% from the nominal 56 ticks
- Too many or too few nibbles between calibration pulses
- Checksum error
- Successive calibration pulse differ by more than 1.5625%
- Any nibble data values measured as < 0 or > 15
- Wrong Status and Communication nibbles
- Serial Communication CRC error

Transfer Management Level:

- Receive Data Buffer Overrun
- SPC Data Buffer Underrun

Single Edge Nibble Transmission (SENT)

24.1.7 Digital Glitch Filter

Very slow slopes and signal noise can lead to fast transitions of the input signal. These unwanted transitions are suppressed by a digital glitch filter similar to a Filter and Prescaler Cell (FPC) in Delayed Debounce Filter Mode with up and down (no reset).

It is built for filtering very fast transitions only. The filter calculates the integral of the signal. If the integral reaches a programmable saturation point, the signal change is notified to the pulse measurement unit. Thus it helps to find the exact pulse length for said slow slopes in noisy environment.

The glitch filter is clocked with f_{pdiv} . If the state of the input sample differs from the current output signal value, the internal counter is incremented by one. When the state of the input sample matches the current output signal value and the timer is not in idle, the timer is decremented by one. When the timer matches the compare value stored in IOCRx.DEPTH, the level of the output signal line is inverted. The timer will be reset and set to idle state again.

The depth of the filter can be programmed to a value between 1 and 15. Typically a depth of 3 to 5 T_{pdiv} is sufficient. By default it is cleared. If IOCRx.DEPTH is cleared the filter is inactive. Nevertheless, the input signal is sampled with f_{pdiv} .

The internal signal after the filter will change value not before the new value was sampled DEPTH times. If during this period a spike occurs, it takes $2 \times T_{pdiv}$ times longer for the internal signal to change value. The filters principal implies a delay of the signal by $(DEPTH \times T_{pdiv})$.

Upon detection of glitch during rising or falling edge, IOCRx.REG or IOCRx.FEG is set. The rising / falling edge glitch flags must be reset by software.

Figure 24-14 shows the digital glitch filter:

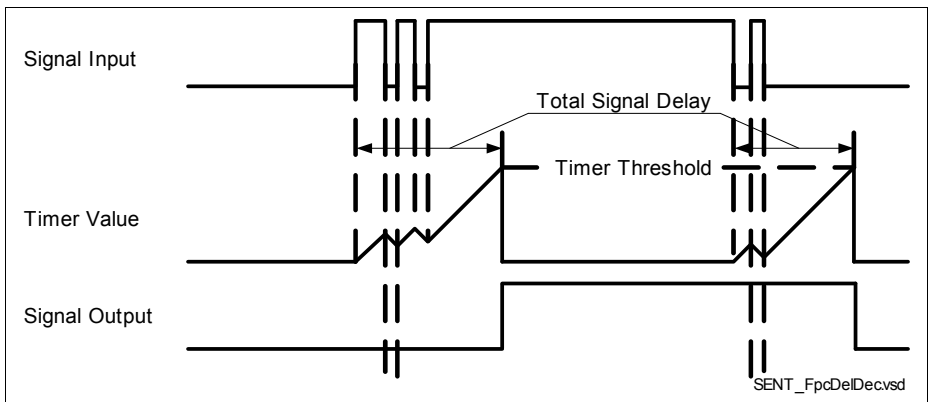


Figure 24-14 Digital Glitch Filter

Single Edge Nibble Transmission (SENT)

24.1.8 Interrupts

4 Interrupt request lines are available for the SENT module.

RDI indicates a receive data interrupt. It is activated when a received frame is moved to a Receive Data Register RDR. RSI indicates a receive frame success interrupt, i.e. the CRC was successful. Both RDI and RSI will be issued together in normal use cases where the frame size is not bigger than 8 nibbles and CRC is correct. RBI indicates a receive buffer overrun interrupt. It is activated when a new frame is transferred to a Receive Data Register RDR while the old value was still not read by the host ("overwrite"), i.e. the kernel wants to set any of the two interrupts RSI and RDI and finds any of these two interrupts already set. TDI indicates a transmit interrupt. It is activated when data is moved from a SCR to a transmit shift register. TBI indicates a transfer buffer under run interrupt. It is set after data has been completely transferred (PLEN exceeded) and no new data was written to SCRx. In addition the protocol error interrupts are available: FRI, FDI, NNI, NVI, CRCI. If one of the protocol interrupts is activated, data is to be treated as invalid according to J2716 JAN2010. WSI, SDI SCRI treat the interrupts referring to the Status and Communication nibble. WDI is the Watch Dog Error Interrupt. It is issued if the time between two frames is too long.

For acceleration of the interrupt service routine, a Register INTOV is implemented that holds a flag for each channel. This flag is automatically set if there is an interrupt pending for the channel which is enabled. It is automatically reset, if no more enabled interrupt is pending for this channel.

The interrupt request or the corresponding interrupt set bit (in register INTSET) can trigger the interrupt generation at the selected interrupt node. The service request pulse is generated independently from the interrupt flag in register INTSTATx. The interrupt flag can be cleared by software by writing to the corresponding bit in register INTCLR. If more than one interrupt source is connected to the same interrupt node pointer (in register INPx), the requests are combined to one common line.

24.1.9 Trigger Outputs

Any interrupt source can be used as trigger output TRIGO outside the module. Each TRIGO is connected to a IR input. See [Table 24-9 "Service Request Lines of SENT" on Page 24-71](#).

Single Edge Nibble Transmission (SENT)

24.2 SENT Kernel Registers

This section describes the kernel registers of the SENT module. All SENT kernel register names described in this section will be referenced in other parts of the TC26x B-Step User’s Manual by the module name prefix “SENT_” for the SENT interface.

All registers in the SENT address spaces are reset with the application reset (definition see SCU section “Reset Operation”).

SENT Kernel Register Overview

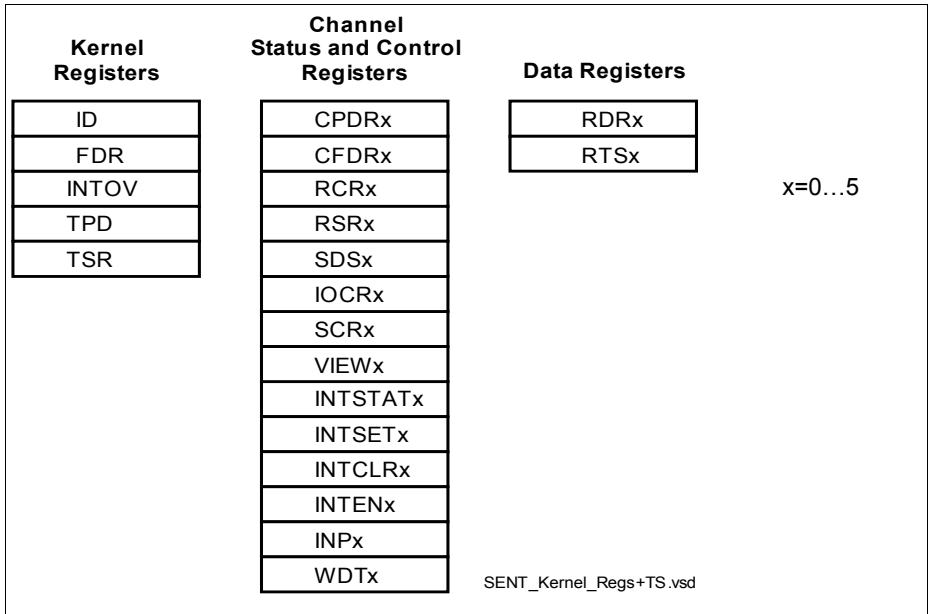


Figure 24-15 SENT Kernel Registers

The complete and detailed address map of the SENT module is described in [Table 24-7](#) on [Page 24-22](#).

Note: x can take the values 0 ... 5

Table 24-7 Registers Address Space - SENT Kernel Registers

Module	Base Address	End Address	Note
SENT	F0003000 _H	F0003AFF _H	–

Single Edge Nibble Transmission (SENT)
Table 24-8 Registers Overview - SENT Kernel Registers

Register Short Name	Register Long Name	Offset Address ¹⁾	Access Mode		Description see
			Read	Write	
SENT_CLC	SENT Clock Control Register	00 _H	SV, U	SV, E, P	Page 24-77
-	reserved	04 _H	BE	BE	
SENT_ID	Module Identification Register	08 _H	SV, U	BE	Page 24-26
SENT_FDR	Module Fractional Divider	0C _H	SV, U	E, P	Page 24-27
-	reserved	10 _H	BE	BE	
SENT_INTOV	Interrupt Overview Register	14 _H	SV, U	BE	Page 24-53
SENT_TSR	Module Time Stamp Register	18 _H	SV, U	BE	Page 24-28
SENT_TPD	Module Time Stamp Predivider Register	1C _H	SV, U	E, P	Page 24-29
-	reserved	20 _H - 7C _H	BE	BE	
SENT_RDRx	Receive Data Register x	80 _H - 80 _H +x*4 _H	SV, U	BE	Page 24-46
-	reserved	84 _H +5*4 _H - E4 _H	BE	BE	
OCS	OCDS Control and Status Register	E8 _H	U, SV	SV, E, P	Page 24-78
ACCEN0	Access Enable Register 0	FC _H	U, SV	SV, SE, P	Page 24-79
ACCEN1	Access Enable Register 1	F8 _H	U, SV	SV, SE, P	Page 24-80
KRST0	Reset Control Register 0	F4 _H	U, SV	SV, E, P	Page 24-81
KRST1	Reset Control Register 1	F0 _H	U, SV	SV, E, P	Page 24-82
KRSTCLR	Reset Status Clear Register	EC _H	U, SV	SV, E, P	Page 24-83

Single Edge Nibble Transmission (SENT)
Table 24-8 Registers Overview - SENT Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset Address 1)	Access Mode		Description see
			Read	Write	
SENT_CPDRx	Channel Pre Divider Register x	$100_H + x$ $* 40_H$	SV, U	SV, U, P	Page 24-30
SENT_CFDRx	Channel Fractional Divider Register x	$104_H + x$ $* 40_H$	SV, U	SV, U, P	Page 24-31
SENT_RCRx	Receiver Control Register x	$108_H + x$ $* 40_H$	SV, U	SV, U, P	Page 24-32
SENT_RSRx	Receive Status Register x	$10C_H + x$ $* 40_H$	SV, U	BE	Page 24-40
SENT_SDSx	Serial Data and Status Register x	$110_H + x$ $* 40_H$	SV, U	BE	Page 24-41
SENT_IOCRRx	Input and Output Control Register x	$114_H + x$ $* 40_H$	SV, U	SV, U, P	Page 24-42
SENT_SCRx	SPC Control Register x	$118_H + x$ $* 40_H$	SV, U	SV, U, P	Page 24-51
SENT_VIEWx	Receive Data View Register x	$11C_H + x$ $* 40_H$	SV, U	SV, U, P	Page 24-48
SENT_INTSTATx	Interrupt Status Register x	$120_H + x$ $* 40_H$	SV, U	BE	Page 24-54
SENT_INTSETx	Interrupt Set Register x	$124_H + x$ $* 40_H$	SV, U	SV, U, P	Page 24-60
SENT_INTCLRx	Interrupt Clear Register x	$128_H + x$ $* 40_H$	SV, U	SV, U, P	Page 24-62
SENT_INTENx	Interrupt Enable Register x	$12C_H + x$ $* 40_H$	SV, U	SV, U, P	Page 24-64
SENT_INPx	Interrupt Node Pointer Register x	$130_H + x$ $* 40_H$	SV, U	SV, U, P	Page 24-67
SENT_WDTx	Watch Dog Timer Register x	$134_H + x$ $* 40_H$	SV, U	SV, U, P	Page 24-39
-	reserved	$138_H + x$ $* 40_H$ - $13F_H + x$ $* 40_H$	BE	BE	

Single Edge Nibble Transmission (SENT)

Table 24-8 Registers Overview - SENT Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset Address ¹⁾	Access Mode		Description see
			Read	Write	
-	reserved	9F0 _H - A7C _H	BE	BE	
SENT_RTsx	Receive Time Stamp x	A80 + x * 4 _H	SV, U	BE	Page 24-47
	Reserved	AA8 _H - AFC _H	BE	BE	-

1) The absolute register address is calculated as follows:

Module Base Address ([Table 24-7](#)) + Offset Address (shown in this column)

List of Access Protection Abbreviations

U - User Mode

SV - Supervisor Mode

BE - Bus Error

nBE - no Bus Error

P - Access Protection, as defined by the ACCEN Register

E - ENDINIT

SE - Safety ENDINIT

Single Edge Nibble Transmission (SENT)

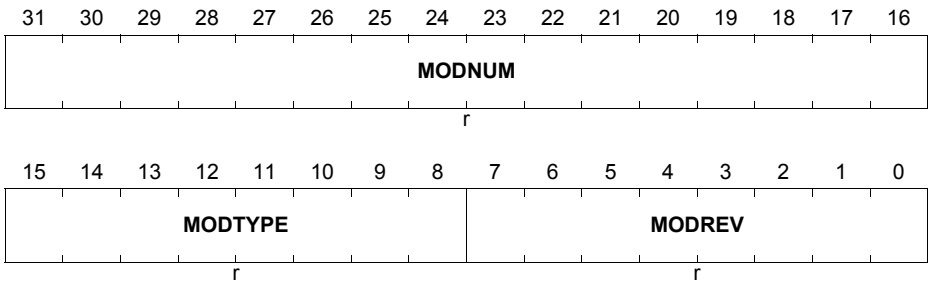
24.2.1 Module Control

Module Identification Register

The SENT Module Identification Register ID contains read-only information about the module version.

ID

Module Identification Register (08_H) Reset Value: 0080 C0XX_H

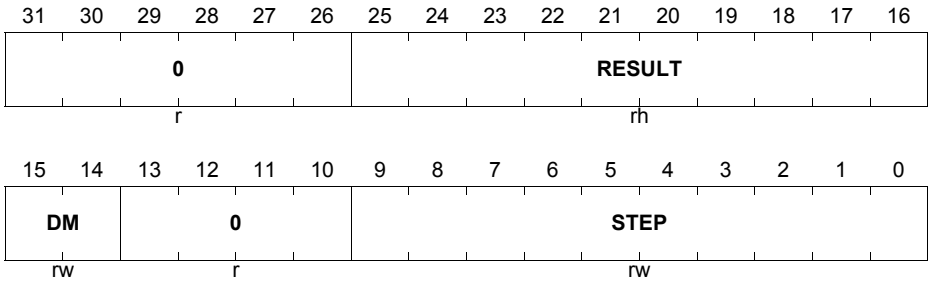


Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MODTYPE	[15:8]	r	Module Type This bit field defines the module as a 32-bit module: C0 _H
MODNUM	[31:16]	r	Module Number Value This bit field defines the module identification number for the SENT: 0080 _H

Fractional Divider Register

The Fractional Divider Register controls the input clock f_{fracdiv} of all SENT channels.

Single Edge Nibble Transmission (SENT)

SENT_FDR
SENT Fractional Divider Register
(0C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
STEP	[9:0]	rw	Step Value Reload or addition value for RESULT.
DM	[15:14]	rw	Divider Mode DM selects normal or fractional divider mode. 00 _B Fractional divider is switched off; no output clock is generated. The Reset External Divider signal is 1. RESULT is not updated (default after System Reset). 01 _B Normal Divider Mode selected. 10 _B Fractional Divider Mode selected. 11 _B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	[25:16]	rh	Result Value Bit field for the addition result.
0	[13:10], [31:26]	r	Reserved Read as 0; should be written with 0.

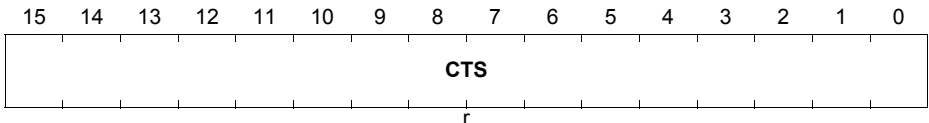
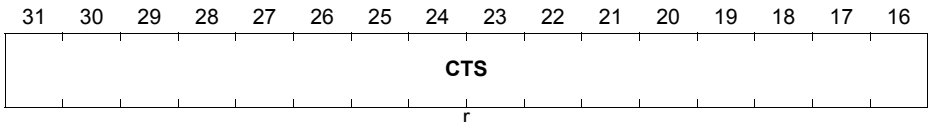
Single Edge Nibble Transmission (SENT)

Module Time Stamp Register

The SENT Module Time Stamp Register contains read-only information about the current time given in clock cycles generated from TPD since last reset while module was clocked. Writing to TPD clears this register.

TSR

Time Stamp Register (18_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
CTS	[31:0]	r	Current Time Stamp for the Module This bit field shows the current time stamp.

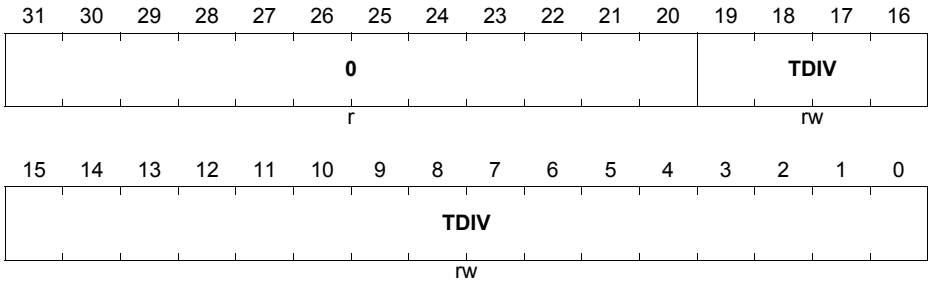
Single Edge Nibble Transmission (SENT)

Module Time Stamp Pre Divider Register

The SENT Module Time Stamp Predivider Register contains the pre divider that defines the time resolution of TSR. Writing to TPD clears register TSR.

TPD

Time Stamp Predivider Register (1C_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
TDIV	[19:0]	rw	Divider Factor of Pre Divider for TSR Divides $f_{fracdiv}$ by (TDIV + 1) and drives TSR.
0	[31:20]	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)

24.2.2 Channel Baud Rate Registers

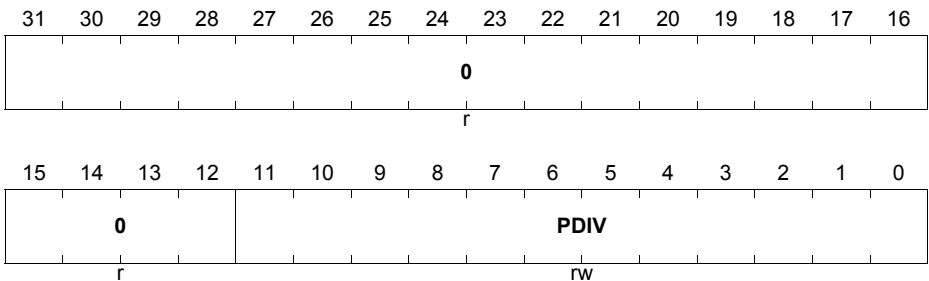
Channel Pre Divider Register

The Channel Pre Divider Register CPDRx contains the pre divider that is related to the SENT channel baud rate.

See [Equation \(24.3\)](#)

CPDRx (x = 0-5)

Channel Pre Divider Register x (100_H+40_H*x) Reset Value: 0000 0000_H



Field	Bits	Type	Description
PDIV	[11:0]	rw	Divider Factor of Pre Divider for Channel x Divides $f_{fracdiv}$ by (PDIV + 1) and delivers f_{pdiv_x} to the Channel Fractional Divider. RCR.CEN must be cleared before changing CPDR.PDIV or CFDR.DIV.
0	[31:12]	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)

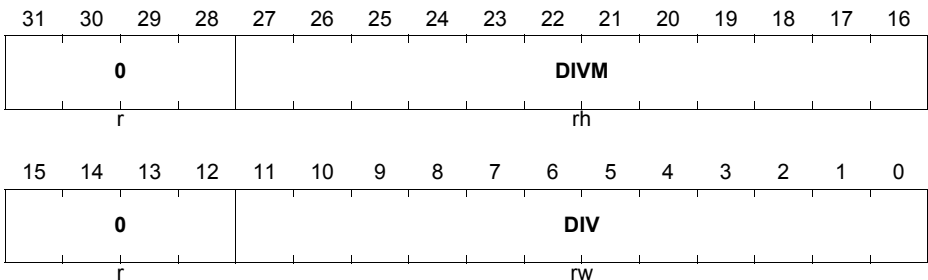
Channel Fractional Divider Register

The Channel Fractional Divider Register CFDRx contains control bits/bit fields that are related to the SENT channel baud rate.

See [Equation \(24.4\)](#) in [Chapter 24.1.5](#) for a detailed description.

CFDRx (x = 0-5)

Channel Fractional Divider Register x(104_H+40_H*x) Reset Value: 0000 0000_H



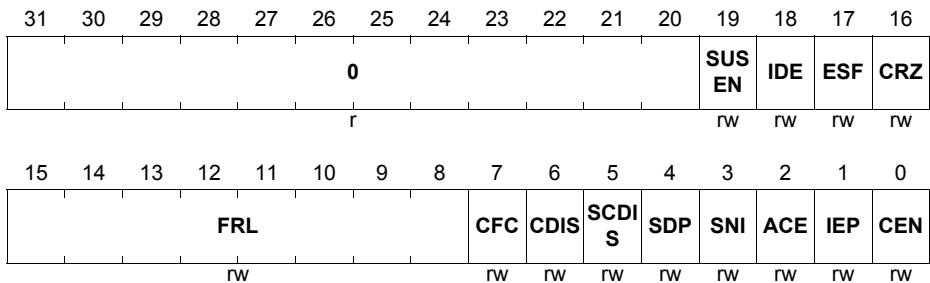
Field	Bits	Type	Description
DIV	[11:0]	rw	Divider Value Initial and reference divider value for the CFDR. DIV must be programmed > 0. If cleared, DIV becomes 1. If written, DIVM is updated automatically with the same value. RCR.CEN must be cleared before changing CPDR.PDIV or CFDR.DIV.
DIVM	[27:16]	rh	Measured Divider Value DIVM is automatically updated by HW to adjust the receiver frequency to the current sender frequency. This value is kept automatically in the range of 75% DIV < DIVM < 125% DIV Write data is ignored.
0	[15:12], [31:28]	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)
24.2.3 Receiver Control and Status Registers
Receiver Control Register

The Receiver Control Register RCRx contains control bits/bit fields that are related to the SENT receiver operation.

RCRx (x = 0-5)

Receiver Control Register x (108_H+40_H*x) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
CEN	0	rw	Channel Enable When CEN is set, the receiver of channel x is enabled. The internal receiver state machine can be initialized by switching the channel off and on. This does not change the current register content. 0 _B channel x disabled (default) 1 _B channel x enabled
IEP	1	rw	Ignore End Pulse When IEP is set, an end pulse is ignored. An end pulse can be generated in SPC mode or as pause pulse. 0 _B End Pulse not ignored (default) 1 _B End Pulse ignored For systems with an end pulse, during synchronize or re-synchronize of reception, if calibration pulses are detected one immediately following the other, the first calibration pulse shall be ignored as it may be a pause pulse with duration matching the calibration pulse range.

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
ACE	2	rw	<p>Alternative CRC Mode Enable</p> <p>When ACE is set, the CRC is calculated in an alternative way for both: fast (signal) and slow (serial message) data path.</p> <p><i>Note: If ESF is set, the standard 6 bit CRC is always used for the serial message and ACE is ignored.</i></p> <p>0_B Serial CRC calculation as specified in J2716 JAN2010 (default)</p> <p>1_B Alternative: 4 bit in parallel CRC calculation as used e.g. in hall sensor TLE4998C.</p>
SNI	3	rw	<p>Status Nibble Included in CRC</p> <p>When SNI is set, the status Nibble is included in (signal data) CRC.</p> <p>0_B Status Nibble not included in CRC (default)</p> <p>1_B Status Nibble included in CRC (as used e.g. in hall sensor TLE4998C).</p>
SDP	4	rw	<p>Serial Data Processing Mode</p> <p>This bit switches automatic serial data processing on.</p> <p>0_B Automatic Serial Data Processing is disabled. Status and Communication nibble can be read from RSRx for SW processing. (default)</p> <p>1_B Automatic Serial Data Processing is enabled. Status and Communication nibble can be read from RSRx; Message ID, Serial Data and SCRC can be read from SDSx after serial data interrupt SDI is activated.</p>
SCDIS	5	rw	<p>CRC for Serial Data Disabled Mode</p> <p>This bit selects the CRC disabled mode.</p> <p>00_B CRC is enabled (default)</p> <p>01_B CRC is disabled CRC nibble can be read from SDSx. The CPU must perform the CRC on the current data by SW.</p>

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
CDIS	6	rw	CRC Disabled Mode This bit selects the CRC disabled mode. 00 _B CRC is enabled (default) 01 _B CRC is disabled CRC nibble can be read from RSRx. The CPU must perform the CRC on the current data by SW.

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
CFC	7	rw	<p>Consecutive Frame Check This bit determines the way the most recently received frame buffer is indicated as valid.</p> <p>0_B Check against Past Sync Pulse The current Synchronization / Calibration Pulse is compared to the Synchronization / Calibration Pulse received immediately before. The whole frame is invalid if the Synchronization / Calibration Pulse length differs from the length of the Synchronization / Calibration Pulse before by more than 1.5625%. In this case of error, its length is not used as new reference. In case the check passes and no other error occurs the Frame Buffer is indicated valid immediately after CRC calculation result is correct. Resynchronization: On the third successive calibration pulse error, the current calibration pulse value is considered as valid and the message accepted unless the message frame contains other errors. In both cases a receive data interrupt (RDI), or a referring error interrupt is issued.</p> <p>1_B Check against Future Sync Pulse The current Synchronization / Calibration Pulse is compared with the Synchronization / Calibration Pulse received immediately after the current frame. The whole frame is invalid if the Synchronization / Calibration Pulse length differs from the length of the following Synchronization / Calibration Pulse by more than 1.5625%. Resynchronization: In this case of error, the current length is used as new reference. Note: The whole frame can be indicated valid only after additionally measuring the Synchronization / Calibration Pulse of the successive frame.</p>

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
FRL	[15:8]	rw	<p>Frame Length</p> <p>FRL determines the number of data nibbles per frame that the SENT channel x is setup for. Note that FRL does not include the Synchronization / Calibration Pulse, the Status and Communication nibble, the CRC nibble nor the additional zero length nibble that might be introduced by use of SPC.</p> <p>00000000_B No data nibble 00000001_B 1 data nibble 00000010_B 2 data nibbles 00000011_B 3 nibbles 00001000_B 8 nibbles Maximum in normal length mode 11111111_B 255 nibbles</p> <p>If more than 8 nibbles are configured, please note: In addition to the receive success interrupt RSI at the successfully received end of a frame, a receive data interrupt RDI is issued each time 8 nibbles have been transferred to the Receive Data Register RDRx. At the end of a frame, RDI is issued if RSI is issued. If an error occurred, RDI is not set at the end of a frame. If no CRC has been received at the point in time where RDI is issued, the receive data interrupt is no indication whether or not the transfer was successful so far. A CRC Error Interrupt is issued at the end of the frame if Automatic CRC check is enabled and the CRC is wrong.</p>

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
CRZ	16	rw	<p>CRC with Zero Nibble for Serial Data</p> <p>This bit selects the CRC method. If CRZ is cleared, augmentation is selected, (i.e a ZERO NIBBLE is added at the end of CRC calculation (only in calculation)). E.g. as 7th nibble (in case of 6 data nibbles)</p> <p>00_B Augmentation is selected for both 4-bit message CRC and the 4-bit CRC of the serial messages (legacy, 16 frames) (default)</p> <p>01_B Augmentation is switched off for both 4-bit message CRC and the 4-bit CRC of the serial messages (legacy, 16 frames).</p> <p>The enhanced serial message (18 Frames, 6-bit CRC) is not controlled by CRZ but the 6-bit CRC is always augmented according to standard.</p>
ESF	17	rw	<p>Enhanced Serial Frame Mode</p> <p>This bit selects the serial frame structure.</p> <p>0_B short (16 frames, 4 bit ID, 8 bit data, 4 bit CRC)</p> <p>1_B enhanced (18 frames, 4 or 8 bit ID, 12 or 16 bit data, 6 bit CRC)</p>
IDE	18	rw	<p>Ignore Drift Error Mode</p> <p>This bit selects if drift errors lead to frame rejection and if an interrupt (INTSTAT.FDI) is generated.</p> <p>Used, if sensors are triggered by SPC. During a long pause period the accumulated drift could be more than 1.5625%. In this special case setting IDE is useful.</p> <p>0_B Drift Errors enabled (default)</p> <p>1_B Drift Errors disabled</p>
SUSEN	19	rw	<p>Suspend Enable</p> <p>This bit makes it possible to set the SENT channel into Suspend Mode via OCDS (on chip debug support):</p> <p>0_B An OCDS suspend trigger is ignored by this SENT channel.</p> <p>1_B An OCDS suspend trigger disables the SENT channel: As soon as the SPC sender logic of the SENT channel becomes idle, the module is stopped while all registers of the channel stay readable. The receiver is stopped unconditionally. A partly received frame is discarded.</p> <p>Bit SUSEN is reset via OCDS Reset.</p>

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
0	[31:20]	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)

Watch Dog Timer Register

The Watch Dog Timer Register WDTx contains the time out value for channel x. The internal Watch Dog timer is cleared and started automatically each time an RDI (Receive Data Interrupt Request Flag) is set in the INTSTAT of the referring channel and also on any write to WDL that sets WDL>0. The value entered here defines the time in multiples of T_{tick_x} (defined in CFDR) from the last RDI on channel x.

The internal Watch Dog Timer is compared to WDLx.

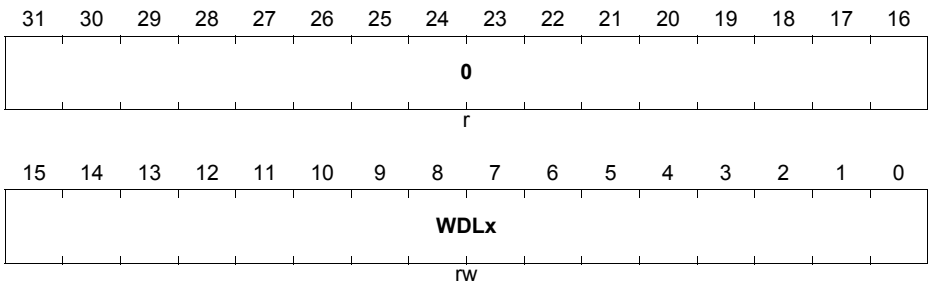
A match triggers interrupt WDI and stops the internal Watch Dog Counter x.

After such a match, the Watch Dog Timer will be cleared and started automatically with clearing bit WDI (by setting INTCLR.WDI). I.e. the Watch Dog Counter is running only if interrupt flag WDI is cleared.

If WDL is cleared, the WDTx is stopped/disabled.

WDTx (x = 0-5)

Watch Dog Timer Register x $(134_H + x * 40_H)$ **Reset Value: 0000 0000_H**

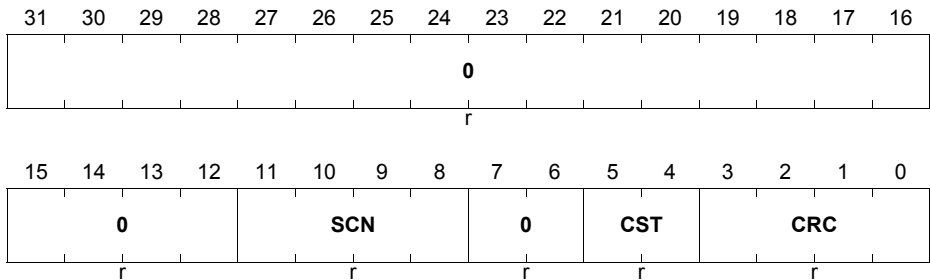


Field	Bits	Type	Description
WDLx	[15:0]	rw	Watch Dog Timer Limit for channel x .
0	[31:16]	r	Reserved Read as 0; should be written with 0.

Receiver Status Register

The Receive Status Register provides the status information of channel x.

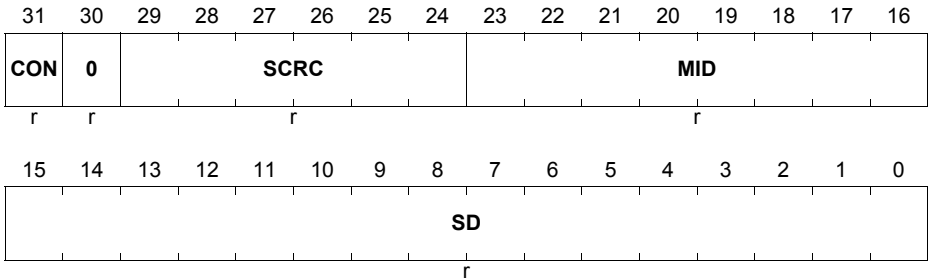
Single Edge Nibble Transmission (SENT)

RSRx (x = 0-5)
Receive Status Register x ($10C_H + 40_H * x$) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
CRC	[3:0]	r	CRC of last frame. CRC ₀ is on bit position 0.
CST	[5:4]	r	Channel Status CST shows the current status of channel x. 00 _B STOP Channel is disabled and can be configured 01 _B INITIALIZED Channel is configured and enabled and no Synchronization / Calibration Pulse was received since last enable. 10 _B RUNNING one or more Synchronization / Calibration Pulses were received and Frequency Range or Frequency Drift not or no longer in range. Fallback status from SYNCHRONIZED. 11 _B SYNCHRONIZED Frequency Range and Frequency Drift in range
SCN	[11:8]	r	Status and Communication Nibble of last frame. SCN ₀ is on bit position 8.
0	[7:6], [31:12]	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)
Serial Data and Status Register

The Serial (Receive) Data and Status Register provides the data and status information of channel x.

SDSx (x = 0-5)
Serial Data and Status Register x (110_H+40_H*x)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SD	[15:0]	r	Serial Data of last serial data frame. SD ₀ is on bit position 0. If RCR.ESF is cleared 8 bits of data are available and bits [15:8] are zero. If RCR.ESF is set and if SDS.CON is cleared 12 bits of data are available and bits [15:12] are zero.
MID	[23:16]	r	Message ID of last serial data frame. ID ₀ is on bit position 16. If RCR.ESF is cleared, or if SDS.CON is set, bits [23:20] are zero.
SCRC	[29:24]	r	SCRC CRC of last serial data frame. CRC ₀ is on position 24. If RCR.ESF is cleared, bits [29:28] are always zero.
CON	31	r	Configuration bit of last serial frame. 0 _B 12-bit data and 8-bit message ID 1 _B 16-bit data and 4-bit message ID
0	30	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)

24.2.4 Input and Output Control

Input and Output Control Register Functions

The Input and Output Control Register IOCRx determines for the SENT channel x:

for the receiver:

- the alternate input
- the filter depth
- the input signal polarity

for the SPC Unit

- the trigger source
- the output signal polarity

IOCRx (x = 0-5)

Input and Output Control Register x(114_H+40_H*x)

Reset Value: X000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXM	RXM	TRM	CTR	EC								ETS	ETS		
rh	rh	rh	rw	rh								rw	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFE G	CRE G	FEG	REG	0	CEC	IIE	OIE	DEPTH				0	ALTI		
rw	rw	rh	rh	r	w	rw	rw	rw				r	rw		

Field	Bits	Type	Description
ALTI	[1:0]	rw	Alternate Input Select Selects the alternate input for channel y: 0000 _B Alternate Input 0 selected 0001 _B Alternate Input 1 selected ... _B ... 0011 _B Alternate Input 3 selected

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
DEPTH	[7:4]	rw	Digital Glitch Filter Depth DEPTH determines the number of port input samples clocked with f_{pdiv} that are taken into account for the calculation of the floating average. The higher DEPTH is chosen to be, the longer the glitches that are suppressed and the longer the delay of the input signal introduced by this filter. 0000 _B off, default 0001 _B 1 T_{pdiv} 0010 _B 2 0011 _B 3 ... _B ... 1111 _B 15
OIE	8	rw	Output Inverter Enable Channel x Selects the Pulse Polarity of the output of channel x 0 _B Pulse polarity is active low 1 _B Pulse polarity is active high
IIE	9	rw	Input Inverter Enable Channel x Selects the Pulse Polarity of the input of channel x 0 _B Pulse polarity is active low 1 _B Pulse polarity is active high
CEC	10	w	Clear Edge Counter If this bit is set, IOCR.EC is cleared. Always reads back as '0'.
REG	12	rh	Rising Edge Glitch Flag for Channel x Shows the status of the glitch detection of channel x 0 _B No Glitch detected on rising edge 1 _B Glitch detected on rising edge REG is cleared by setting CREG.
FEG	13	rh	Falling Edge Glitch Flag for Channel x Shows the status of the glitch detection of channel x 0 _B No Glitch detected on falling edge 1 _B Glitch detected on falling edge FEG is cleared by setting CFEG.
CREG	14	rw	Clear Rising Edge Glitch Flag for Channel x Clears the status flag REG 0 _B REG is not cleared 1 _B REG is cleared CREG always read zero.

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
CFEG	15	rw	Clear Falling Edge Glitch Flag for Channel x Clears the status flag FEG 0 _B FEG is not cleared 1 _B FEG is cleared CFEG always read zero.
ETS	[19:16]	rw	External Trigger Select Selects the external trigger line if SCR _x .TRIG is programmed to 11 _B . 0000 _B TRIG0 0001 _B TRIG1 5 _D TRIG5 1111 _B reserved
EC	[27:20]	rh	Edge Counter This bit field contains a counter with saturation (stops at 0xFF). It is incremented with any falling edge that appears on the input pin selected by IOCR.ALTI. Note that this holds true in all states (STOP, INITIALIZED, RUNNING, SYNCHRONIZED). It is intended for debugging, in particular to find a bubbling idiot that sends before enabling the module.
CTR	28	rw	Clear Trigger Monitor Flag for Channel x Clears the status flag TRM 0 _B TRM is not cleared 1 _B TRM is cleared CTR always read zero. Reset value of CTR is 0.
TRM	29	rh	Trigger Monitor Flag for Channel x Shows the status of the trigger detection of channel x 0 _B No Trigger detected 1 _B Trigger detected (one or several) TRM is cleared by setting CTR. Reset value of TRM is 0.
RXM	30	rh	Receive Monitor for Channel x Shows the status of the receive signal of channel x after glitch filtering and inverted as specified by IIE. 0 _B Current signal is low. 1 _B Current signal is high. Reset value of RXM is X.

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
TXM	31	rh	Transmit Monitor for Channel x Shows the status of the transmit signal of channel x inverted as specified by OIE. 0_B Current signal is low. 1_B Current signal is high. Reset value of TXM is X.
0	[3:2], 11	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)

24.2.5 Receive Data Registers

Receive Data Registers RDRx

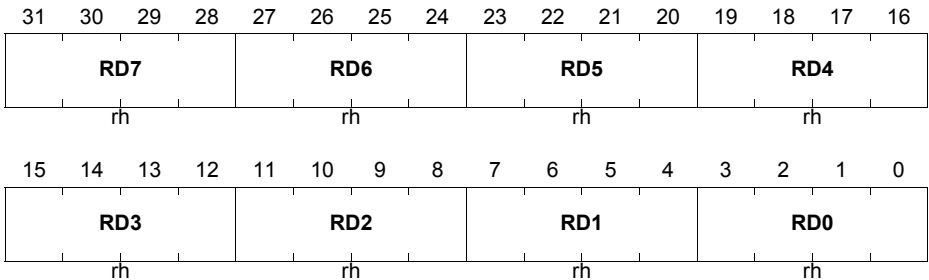
The Receive Data Registers RDRx for channel x shows the data content of a received data frame. Register VIEWx is used to sort the nibbles.

*Note: Register VIEW must be set up correctly to see all data nibbles of the frame!
By default the application software set VIEW to 7654 3210_H.*

*Note: The internal receive buffer is always cleared (0x0000 0000_H) at each frame start.
Thus unused nibbles are always read as zero.*

RDRx (x = 0-5)

Receive Data Register x (80_H+x*4_H) Reset Value: 0000 0000_H



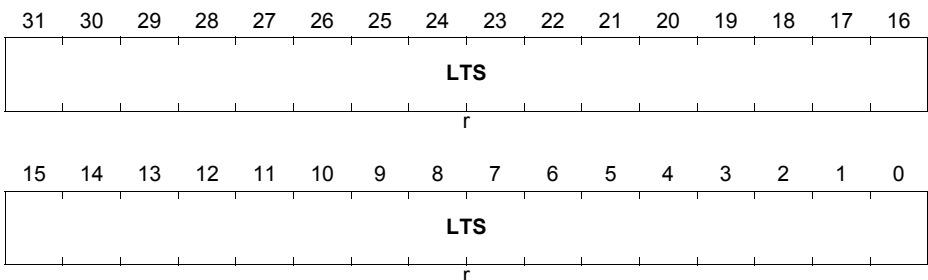
Field	Bits	Type	Description
RDRy (y = 0-7)	[4*y+3 : 4*y]	rh	Receive Data Nibble y RDRy shows the nibble from the received frame that is sorted to this position. It can be selected by any of VIEWx.RDNPy (y = 0-7). By default all nibbles are sorted to RD0 as the reset value of VIEW is 0x0000 0000 _H . I.e. at the end of frame reception RD0 contains the last data nibble of the frame.

Single Edge Nibble Transmission (SENT)
Channel Receive Time Stamp Register

The SENT Channel x Receive Time Stamp Register contains read-only information about the time the last frame for channel x was received. Time is captured with the second falling edge, i.e. at the Status/Communication data pulse.

RTSx (x=0-5)

Receive Time Stamp Register x ($A80_H + x * 4_H$) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
LTS	[31:0]	r	Last Receive Time Stamp for Channel x This bit field shows the time stamp of the last frame on channel x.

Receive Data View Register VIEWx

The Receive Data View Registers VIEWx stores the nibble pointers. They determine the sequence in which the received data nibbles are presented to the host. This reduces the SW effort to sort the nibbles.

The data nibble that is received first in the received frame is moved to the location given in register VIEW.RDNP0, the second to VIEW.RDNP1 and so on until VIEW.RDNP7.

If more than one VIEW.RDNPx point to a certain location in RDR, the last one will overwrite the previous ones.

Example: two 12 bit values are transmitted. One with highest significant nibble first and one with lowest significant nibble first. The frame looks like this: 456321_H. Note that the 1 is received as first data nibble and the 4 comes in as last data nibble.

The actual signal values are 0x123_H and 0x456_H. By using VIEWx this can be sorted out into two 16bit values by HW. In the example VIEWx would be set to: 73 654 012_H. 73 is a dummy value and is not regarded if not more than 6 nibbles are received in a frame. The Register RDRx looks like this: 0x0456 0123_H to the host.

In the example RDR nibbles 3 and 7 contain 0x0000_B as the Receive Buffer is always cleared (0x0000 0000_H) before new data is received.

Single Edge Nibble Transmission (SENT)

If a frame contains more than eight nibbles and the sorting can not be specified statically, VIEW can be set to e.g. 7654 3210_H.

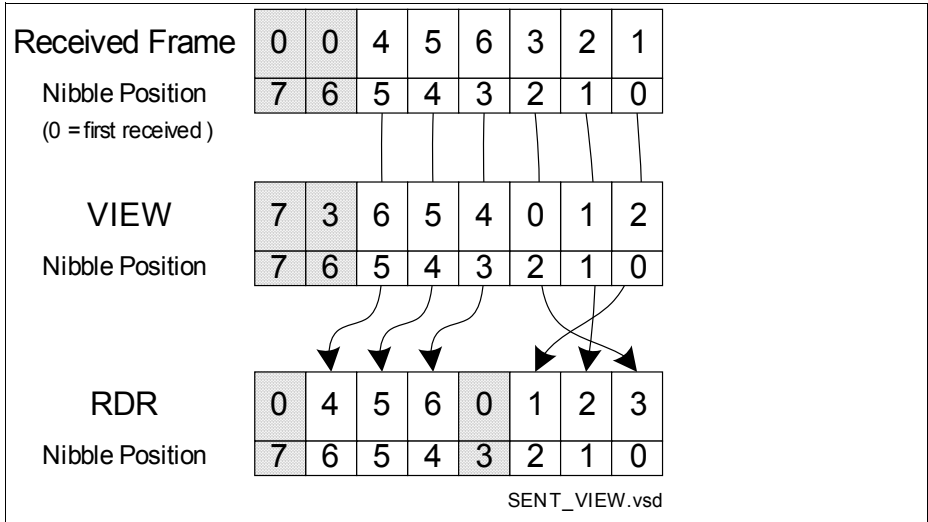
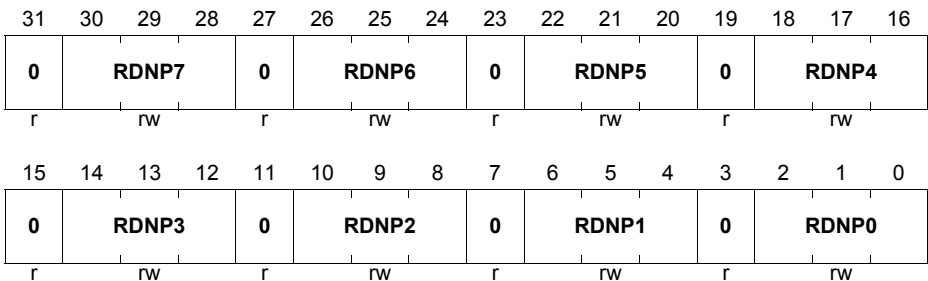


Figure 24-16 Functionality of VIEW Register

VIEW_x (x = 0-5)

Receive Data View Register x (11C_H+40_H*x) Reset Value: 0000 0000_H



Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
RDNP_y (y = 0-7)	[4*y+2 :4*y]	rw	<p>Receive Data Target Nibble Pointer y</p> <p>RDNP_y points to the Nibble in Receive Data Register RDR_x where the nibble y from the received frame is sorted to. Nibble 0 is the first data nibble in the frame. It gets moved to the position defined in RDNP0. And on.</p> <p>000_B Nibble 0 selected 001_B Nibble 1 selected ..._B ... 111_B Nibble 7 selected</p> <p><i>Note: RDNP_y must be written before first frame reception. All RDNP_y must have different values. (Higher RDNP_y overwrite lower RDNP_y.)</i></p>
0	3, 7, 11, 15, 19, 23, 27, 31	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Single Edge Nibble Transmission (SENT)

24.2.6 SPC Control

SPC Control Registers SCR_x

The SPC Control Register SCR contains data to be transmitted during the sync pulse of the data frames. It contains as well the trigger control bits required for sending nibbles from the SENT module to the sensor / external SENT device.

The SPC Control Register is used to control the trigger mode and time base of the SPC channel transmission.

Data and the control bits are collected in this single register to ease transfer of multiple pulses in cases where dynamic switching of the trigger condition is required.

The SPC Control Register is build to control single pulse transfers only.

Thus it is possible to change the control settings of an individual channel from pulse to pulse as it is required in SPC mode "Bidirectional transmit". Here it might be considered useful to change trigger mode between Mode 1 (immediately) and Mode 2 (falling edge of next Synchronization / Calibration Pulse with programmable delay).

In addition repeating transfers with the same control settings are supported. For a pulse transfer to be initiated a synchronization signal can be sufficient and no further SW intervention is required. Here the data can be changed ("ID-Selection" Mode) or simply left constant ("Sync" Mode). Only a HW trigger needs to be set up.

In Mode 0, SPC is deactivated for this channel. A write access to SCR_x does not initiate an SPC pulse transmission. All state transmitter machines are initialized.

In Mode 1, an SPC pulse is sent, each time SPC Control Register SCR_x is written to. If a transfer is ongoing, the channel waits automatically until the internal transmission register is ready. Transmit Buffer Underflow bit TB_{ix} is set after data has been completely transferred (PLEN exceeded) and no new data was written to SCR_x. After the data was transferred to the internal transmission register, interrupt TD_{ix} signals that a new value can be written. INTSTAT_x.TD_{ix} must be cleared by SW. Independently from this interrupt pending bit, a new interrupt pulse is generated on each transfer of an SPC pulse.

This mode is important for back to back transfers of several nibbles as in bidirectional SPC mode.

In Mode 2, an SPC pulse is sent, each time the first falling edge of any Synchronization / Calibration Pulse is received. In this mode, the programmable delay DEL is most useful. In SPC mode "Bidirectional Transmit" this mode is useful to synchronize the transmission. TD_{ix} and TB_{ix} work as in Mode 1.

In Mode 3, an SPC pulse is sent (with current DEL and PLEN) after each external trigger event (defined by IOCR_x.ETS). This is most useful in SPC "sync" mode. TD_{ix} and TB_{ix} work as in Mode 1.

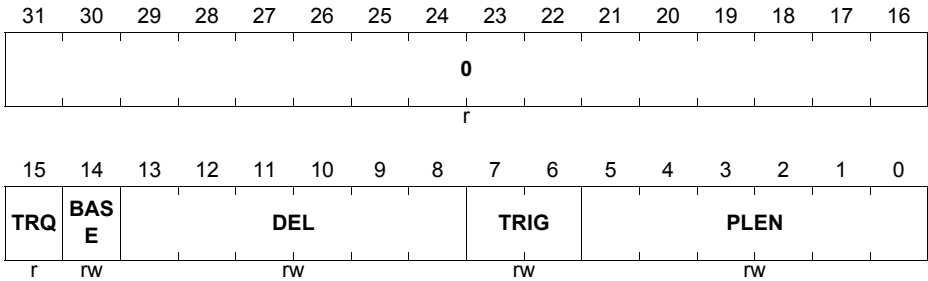
Single Edge Nibble Transmission (SENT)

SCRx (x = 0-5)

SPC Control Register x

(118_H+40_H*x)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
PLEN	[5:0]	rw	<p>Pulse Length</p> <p>Defines the length of the pulse in tick times. The time base is the measured tick time of the latest received frame if selected so by BASE. In case this measured tick time was invalid or not already available after enable of the channel, the nominal time base of the module is used.</p> <p>000000_B Pulse length is 0 ticks 000001_B Pulse length is 1 tick 111111_B Pulse length is 63 ticks</p>

Single Edge Nibble Transmission (SENT)

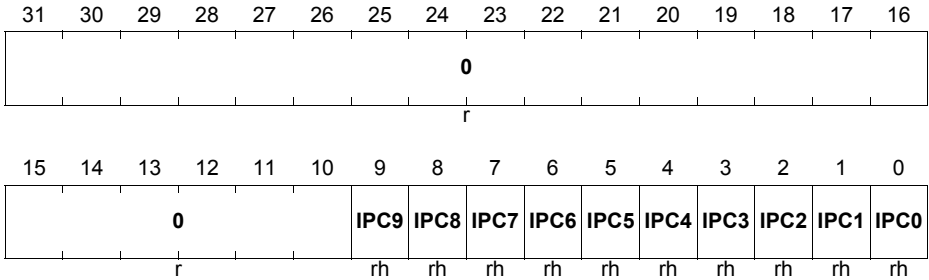
Field	Bits	Type	Description
TRIG	[7:6]	rw	Trigger Source and Mode Selection Selects the Trigger Source and Mode. The internal sender state machine can be initialized by switching the channel off (TRIG is cleared) and on. This does not change the current register content. 00 _B No Pulse is generated, OFF When cleared, an ongoing transfer is stopped immediately and the transmit output is driven recessive. 01 _B Pulse starts immediately (no auto repetition) 10 _B Pulse starts each time the first falling edge of any Synchronization / Calibration Pulse is received (auto repetition on next Sync. / Cal. Pulses) 11 _B Pulse starts after each external trigger event. (auto repetition on next trigger) IOCRx.ETS selects the source of this event.
DEL	[13:8]	rw	Delay Length Selects how long the SPC pulse is delayed after the trigger condition. The time base is the measured tick time of the latest received frame if selected so by BASE. In case this measured tick time was invalid or not already available after enable of the channel, the nominal time base of the module is used. 000000 _B Pulse is not delayed 000001 _B Pulse is delayed by 1 tick ... 111111 _B Pulse is delayed by 63 ticks
BASE	14	rw	Time Base Selects the Pulse Time Base 0 _B Pulse is based on measured frequency of last Synchronization/Calibration Pulse 1 _B Pulse is based on nominal frequency
TRQ	15	r	Transfer Request in Progress While an SPC Pulse is being sent this bit is set. Write access is ignored.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)

24.2.7 Interrupt Control Registers

INTOV

Interrupt Overview Register (14_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
IPC_y (y = 0-5)	y	rh	<p>Interrupt Pending on Channel y</p> <p>If any interrupt requested flag is set for channel y in register INTSTAT_y AND the referring interrupt is enabled in INTEN_x then IPC_y is set. It is automatically reset if all flags in INTSTAT_y are cleared for which the referring interrupt is enabled in INTEN_x.</p> <p><i>Note: Not all IPC0-9 are available, the number of Interrupt Pending on Channel is equivalent to the SENT channels available, e.g 4 SENT channels has 4 Interrupt Pending IPC0-3 and vice versa.</i></p>
0	[31:6]	r	<p>Reserved</p> <p>Read as 0.</p>

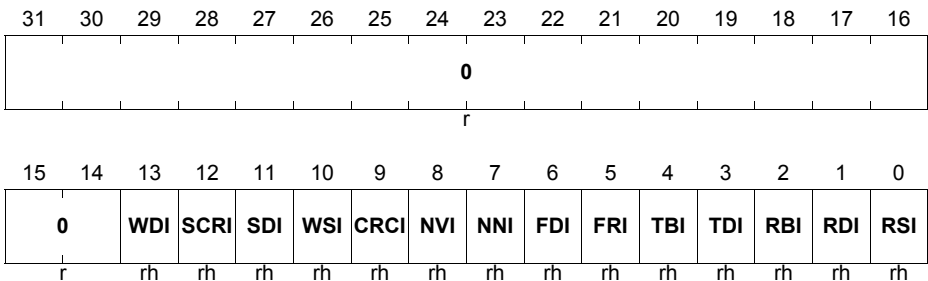
Single Edge Nibble Transmission (SENT)
Interrupt Status Register

The Interrupt Status Register INTSTATx contains status bits that show the status of any interrupt of SENT channel x.

Note: The bits are set independently from the referring Interrupt Enable in Register INTENx. Thus they can be used as status bits as well e.g. by a SW based on polling.

INTSTATx (x = 0-5)

Interrupt Status Register x (120_H+40_H*x) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RSI	0	rh	<p>Receive Success Interrupt Request Flag</p> <p>This bit is set at the successfully received end of a frame. Depending on bit RCRx.CDIS this indicates a successful check of the CRC.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.RSI. This bit can be set by bit INTSET_x.RSI. This bit is set independently from INTEN_x.</p>

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
RDI	1	rh	<p>Receive Data Interrupt Request Flag</p> <p>RDI is activated when a received frame is moved to a Receive Data Register RDR. Both RDI and RSI will be issued together in normal use cases where the frame size is not bigger than 8 nibbles and CRC is correct or not checked (if RCRx.CDIS is cleared).</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.RDI. This bit can be set by bit INTSET_x.RDI. This bit is set independently from INTEN_x.</p>
RBI	2	rh	<p>Receive Buffer Overflow Interrupt Request Flag</p> <p>This bit is set after a frame has been received while the old one was not read from RDR_x. I.e. the kernel wants to set any of the two interrupts RSI and RDI and finds any of these two interrupts already set. The old data is overwritten by the new data.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit is NOT cleared by reading RDR_x. This bit can be cleared by bit INTCLR_x.RBI. This bit can be set by bit INTSET_x.RBI. This bit is set independently from INTEN_x.</p>
TDI	3	rh	<p>Transfer Data Interrupt Request Flag</p> <p>This bit is set after the trigger condition was detected. Data to be transferred has been moved internally. Thus a new value can be written to SCR_x. This can be used for back to back transfers.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit is automatically cleared by writing SCR_x. This bit can be cleared by bit INTCLR_x.TDI. This bit can be set by bit INTSET_x.TDI. This bit is set independently from INTEN_x.</p>

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
TBI	4	rh	<p>Transmit Buffer Underflow Interrupt Request Flag</p> <p>This bit is set after data has been completely transferred (PLEN exceeded) and no new data was written to SCR_x.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit is NOT cleared by writing SCR_x. This bit can be cleared by bit INTCLR_x.TBI. This bit can be set by bit INTSET_x.TBI. This bit is set independently from INTEN_x.</p>
FRI	5	rh	<p>Frequency Range Interrupt Request Flag</p> <p>This bit is set after a Synchronization / Calibration pulse was received that deviates more than +- 25% from the nominal value. The referring data is ignored.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.FRI. This bit can be set by bit INTSET_x.FRI. This bit is set independently from INTEN_x.</p>
FDI	6	rh	<p>Frequency Drift Interrupt Request Flag</p> <p>This bit is set after a subsequent Synchronization / Calibration pulse was received that deviates more than 1.5625% (1/64) from its predecessor. (See RCR.CFC)</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.FDI. This bit can be set by bit INTSET_x.FDI. This bit is set independently from INTEN_x.</p>

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
NNI	7	rh	<p>Number of Nibbles Wrong Request Flag</p> <p>This bit is set after a more nibbles have been received than expected or a Synchronization / Calibration Pulse is received too early thus too few nibbles have been received.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.NNI. This bit can be set by bit INTSET_x.NNI. This bit is set independently from INTEN_x.</p>
NVI	8	rh	<p>Nibbles Value out of Range Request Flag</p> <p>This bit is set after a too long or too short nibble pulse has been received. I.e. value < 0 or value > 15.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.NVI. This bit can be set by bit INTSET_x.NVI. This bit is set independently from INTEN_x.</p>
CRCI	9	rh	<p>CRC Error Request Flag</p> <p>This bit is set if the CRC fails.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.CRCI. This bit can be set by bit INTSET_x.CRCI. This bit is set independently from INTEN_x.</p>

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
WSI	10	rh	<p>Wrong Status and Communication Nibble Error Request Flag</p> <p>In Short Serial Frame Mode (RCR.ESF is cleared), this bit is set if the Status and Communication nibble shows a start bit in a frame other than frame number $n \times 16$.</p> <p>In Enhanced Serial Frame Mode this bit is without function.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.WSI. This bit can be set by bit INTSET_x.WSI. This bit is set independently from INTEN_x.</p>
SDI	11	rh	<p>Serial Data Receive Interrupt Request Flag</p> <p>This bit is set after all serial data bits have been received via the Status and Communication nibble. Depending on bit RCR_x.SCDIS this indicates a successful check of the CRC.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.SDI. This bit can be set by bit INTSET_x.SDI. This bit is set independently from INTEN_x.</p>
SCRI	12	rh	<p>Serial Data CRC Error Request Flag</p> <p>This bit is set if the CRC of the serial message fails. In Enhanced Serial Message Format, this includes a check of the Serial Communication Nibble for correct 0 values of bit 3 in frames 7, 13 and 18.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.SCRI. This bit can be set by bit INTSET_x.SCRI. This bit is set independently from INTEN_x.</p>

Single Edge Nibble Transmission (SENT)

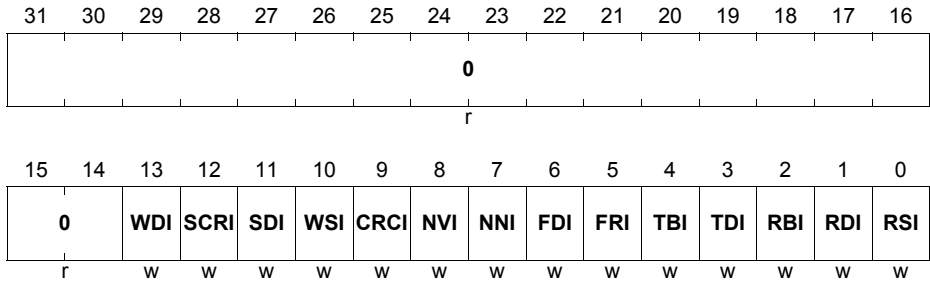
Field	Bits	Type	Description
WDI	13	rh	<p>Watch Dog Error Request Flag</p> <p>This bit is set if the Watch Dog Timer of the channel x expires.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.WDI. This bit can be set by bit INTSET_x.WDI. This bit is set independently from INTEN_x.</p>
0	[31:14]	r	<p>Reserved</p> <p>Read as 0.</p>

Single Edge Nibble Transmission (SENT)
Interrupt Set Register

The Interrupt Set Register INTSETx contains control bits that trigger an interrupt pulse for any interrupt of SENT channel x.

INTSETx (x = 0-5)

Interrupt Set Register x (124_H+40_H*x) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RSI	0	w	Set Interrupt Request Flag RSI Setting this bit set bit INTSTATx.RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	Set Interrupt Request Flag RDI Setting this bit set bit INTSTATx.RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	Set Interrupt Request Flag RBI Setting this bit set bit INTSTATx.RBI. Clearing this bit has no effect. Reading this bit returns always zero.
TDI	3	w	Set Interrupt Request Flag TDI Setting this bit set bit INTSTATx.TDI. Clearing this bit has no effect. Reading this bit returns always zero.
TBI	4	w	Set Interrupt Request Flag TBI Setting this bit set bit INTSTATx.TBI. Clearing this bit has no effect. Reading this bit returns always zero.

Single Edge Nibble Transmission (SENT)

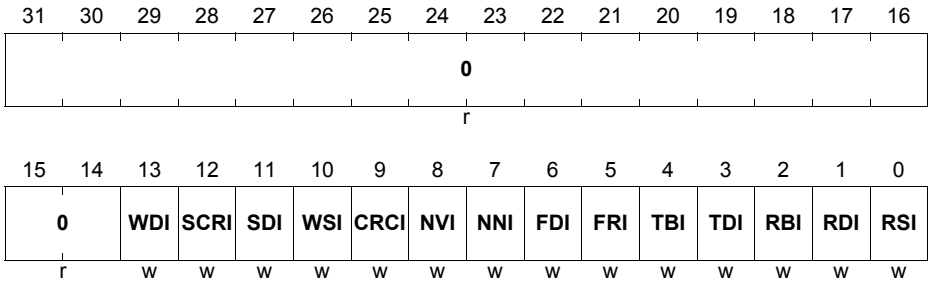
Field	Bits	Type	Description
FRI	5	w	Set Interrupt Request Flag FRI Setting this bit set bit INTSTATx.FRI. Clearing this bit has no effect. Reading this bit returns always zero.
FDI	6	w	Set Interrupt Request Flag FDI Setting this bit set bit INTSTATx.FDI. Clearing this bit has no effect. Reading this bit returns always zero.
NNI	7	w	Set Interrupt Request Flag NNI Setting this bit set bit INTSTATx.NNI. Clearing this bit has no effect. Reading this bit returns always zero.
NVI	8	w	Set Interrupt Request Flag NVI Setting this bit set bit INTSTATx.NVI. Clearing this bit has no effect. Reading this bit returns always zero.
CRCI	9	w	Set Interrupt Request Flag CRCI Setting this bit set bit INTSTATx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.
WSI	10	w	Set Interrupt Request Flag WSI Setting this bit set bit INTSTATx.WSI. Clearing this bit has no effect. Reading this bit returns always zero.
SDI	11	w	Set Interrupt Request Flag SDI Setting this bit set bit INTSTATx.SDI. Clearing this bit has no effect. Reading this bit returns always zero.
SCRI	12	w	Set Interrupt Request Flag SCRI Setting this bit set bit INTSTATx.SCRI. Clearing this bit has no effect. Reading this bit returns always zero.
WDI	13	w	Set Interrupt Request Flag WDI Setting this bit set bit INTSTATx.WDI. Clearing this bit has no effect. Reading this bit returns always zero.
0	[31:14]	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)
Interrupt Clear Register

The Interrupt Clear Register INTCLR_x contains control bits that clear the status of any interrupt of SENT channel x.

INTCLR_x (x = 0-5)

Interrupt Clear Register x (128_H+40_H*x) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RSI	0	w	Clear Interrupt Request Flag RSI Setting this bit clears bit INTSTATx.RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	Clear Interrupt Request Flag RDI Setting this bit clears bit INTSTATx.RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	Clear Interrupt Request Flag RBI Setting this bit clears bit INTSTATx.RBI. Clearing this bit has no effect. Reading this bit returns always zero.
TDI	3	w	Clear Interrupt Request Flag TDI Setting this bit clears bit INTSTATx.TDI. Clearing this bit has no effect. Reading this bit returns always zero.
TBI	4	w	Clear Interrupt Request Flag TBI Setting this bit clears bit INTSTATx.TBI. Clearing this bit has no effect. Reading this bit returns always zero.

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
FRI	5	w	Clear Interrupt Request Flag FRI Setting this bit clears bit INTSTATx.FRI. Clearing this bit has no effect. Reading this bit returns always zero.
FDI	6	w	Clear Interrupt Request Flag FDI Setting this bit clears bit INTSTATx.FDI. Clearing this bit has no effect. Reading this bit returns always zero.
NNI	7	w	Clear Interrupt Request Flag NNI Setting this bit clears bit INTSTATx.NNI. Clearing this bit has no effect. Reading this bit returns always zero.
NVI	8	w	Clear Interrupt Request Flag NVI Setting this bit clears bit INTSTATx.NVI. Clearing this bit has no effect. Reading this bit returns always zero.
CRCI	9	w	Clear Interrupt Request Flag CRCI Setting this bit clears bit INTSTATx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.
WSI	10	w	Clear Interrupt Request Flag WSI Setting this bit clears bit INTSTATx.WSI. Clearing this bit has no effect. Reading this bit returns always zero.
SDI	11	w	Clear Interrupt Request Flag SDI Setting this bit clears bit INTSTATx.SDI. Clearing this bit has no effect. Reading this bit returns always zero.
SCRI	12	w	Clear Interrupt Request Flag SCRI Setting this bit clears bit INTSTATx.SCRI. Clearing this bit has no effect. Reading this bit returns always zero.
WDI	13	w	Clear Interrupt Request Flag WDI Setting this bit clears bit INTSTATx.WDI. Clearing this bit has no effect. Reading this bit returns always zero.
0	[31:14]	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)

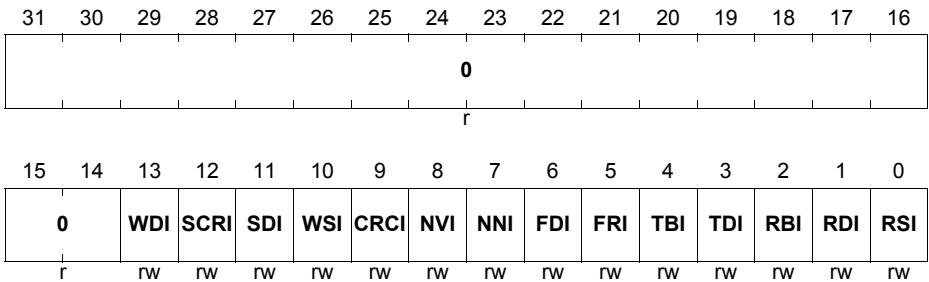
Interrupt Enable Register

The Interrupt Enable Register INTENx contains control bits that enable the interrupt source of any interrupt of SENT channel x.

Note: The Interrupt Status bits in register INTSTATx are set independently from the Interrupt Enable in Register INTENx.

INTENx (x = 0-5)

Interrupt Enable Register x (12C_H+40_H*x) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RSI	0	rw	<p>Enable Interrupt Request RSI</p> <p>0_B No interrupt request can be generated for this source</p> <p>1_B An interrupt request can be generated for this source</p>
RDI	1	rw	<p>Enable Interrupt Request RDI</p> <p>0_B No interrupt request can be generated for this source</p> <p>1_B An interrupt request can be generated for this source</p>
RBI	2	rw	<p>Enable Interrupt Request RBI</p> <p>0_B No interrupt request can be generated for this source</p> <p>1_B An interrupt request can be generated for this source</p>

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
TDI	3	rw	Enable Interrupt Request TDI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
TBI	4	rw	Enable Interrupt Request TBI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
FRI	5	rw	Enable Interrupt Request FRI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
FDI	6	rw	Enable Interrupt Request FDI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
NNI	7	rw	Enable Interrupt Request NNI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
NVI	8	rw	Enable Interrupt Request NVI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
CRCI	9	rw	Enable Interrupt Request CRCI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
WSI	10	rw	Enable Interrupt Request WSI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
SDI	11	rw	Enable Interrupt Request SDI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
SCRI	12	rw	Enable Interrupt Request SCRI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
WDI	13	rw	Enable Interrupt Request WDI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
0	[31:14]	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)

Interrupt Node Pointer Register

The Interrupt Node Pointer Register INP_x contains the node pointers of SENT channel x.

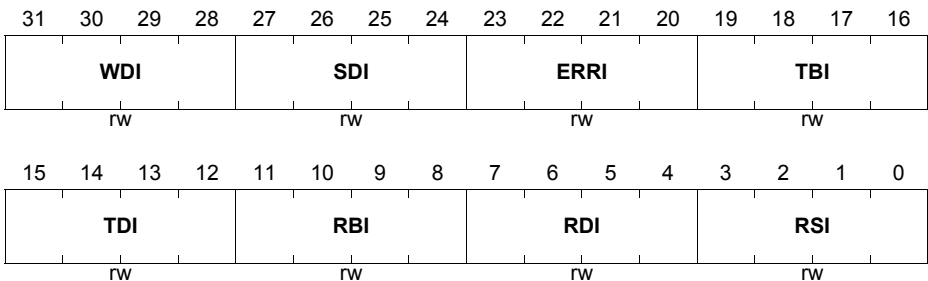
Note: Node Pointer ERRI is one single node pointer for the following error interrupts:

- FRI
- FDI
- NNI
- NVI
- CRCI
- WSI
- SCRI

INP_x (x = 0-5)

Interrupt Node Pointer Register x (130_H+40_H*x)

Reset Value: 0000 0000_H



Field	Bits	Type	Description																				
RSI	[3:0]	rw	<p>Interrupt Node Pointer for Interrupt RSI</p> <p>This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.RSI (if enabled by bit INTENx.RSI).</p> <table style="width: 100%; border: none;"> <tr> <td style="width: 100px;">0000_B</td><td>Interrupt node 0 is selected</td></tr> <tr> <td>0001_B</td><td>Interrupt node 1 is selected</td></tr> <tr> <td>0010_B</td><td>Interrupt node 2 is selected</td></tr> <tr> <td>0011_B</td><td>Interrupt node 3 is selected</td></tr> <tr> <td>0100_B</td><td>Trigger Output TRIGO 0 is selected</td></tr> <tr> <td>0101_B</td><td>Trigger Output TRIGO 1 is selected</td></tr> <tr> <td>0110_B</td><td>Reserved, do not use</td></tr> <tr> <td>...</td><td>...</td></tr> <tr> <td>...</td><td>...</td></tr> <tr> <td>1111_B</td><td>Reserved, do not use</td></tr> </table>	0000 _B	Interrupt node 0 is selected	0001 _B	Interrupt node 1 is selected	0010 _B	Interrupt node 2 is selected	0011 _B	Interrupt node 3 is selected	0100 _B	Trigger Output TRIGO 0 is selected	0101 _B	Trigger Output TRIGO 1 is selected	0110 _B	Reserved, do not use	1111 _B	Reserved, do not use
0000 _B	Interrupt node 0 is selected																						
0001 _B	Interrupt node 1 is selected																						
0010 _B	Interrupt node 2 is selected																						
0011 _B	Interrupt node 3 is selected																						
0100 _B	Trigger Output TRIGO 0 is selected																						
0101 _B	Trigger Output TRIGO 1 is selected																						
0110 _B	Reserved, do not use																						
...	...																						
...	...																						
1111 _B	Reserved, do not use																						

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
RDI	[7:4]	rw	Interrupt Node Pointer for Interrupt RDI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.RDI (if enabled by bit INTENx.RDI). For bit field definition, see RSI.
RBI	[11:8]	rw	Interrupt Node Pointer for Interrupt RBI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.RBI (if enabled by bit INTENx.RBI). For bit field definition, see RSI.
TDI	[15:12]	rw	Interrupt Node Pointer for Interrupt TDI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.TDI (if enabled by bit INTENx.TDI). For bit field definition, see RSI.
TBI	[19:16]	rw	Interrupt Node Pointer for Interrupt TBI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.TBI (if enabled by bit INTENx.TBI). For bit field definition, see RSI.
ERRI	[23:20]	rw	Interrupt Node Pointer for Interrupt FRI, FDI, NNI, NVI, CRCI, WSI, SCRI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.FRI (if enabled by bit INTENx.FRI) or INTSTATx.FDI (if enabled by bit INTENx.FDI) or INTSTATx.NNI (if enabled by bit INTENx.NNI) or INTSTATx.NVI (if enabled by bit INTENx.NVI) or INTSTATx.CRCI (if enabled by bit INTENx.CRCI) or INTSTATx.WSI (if enabled by bit INTENx.WSI) or INTSTATx.SCRI (if enabled by bit INTENx.SCRI). For bit field definition, see RSI.
SDI	[27:24]	rw	Interrupt Node Pointer for Interrupt SDI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.SDI (if enabled by bit INTENx.SDI). For bit field definition, see RSI.

Single Edge Nibble Transmission (SENT)

Field	Bits	Type	Description
WDI	[31:28]	rw	Interrupt Node Pointer for Interrupt WDI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.WDI (if enabled by bit INTENx.WDI). For bit field definition, see RSI.

Single Edge Nibble Transmission (SENT)

24.3 SENT Module Implementation

This section describes the SENT module interface as it is implemented in the TC26x B-Step. It especially covers clock control, port and on-chip connections, interrupt control, and address decoding.

24.3.1 Interface Connections of the SENT Module

Figure 24-17 shows the TC26x B-Step-specific implementation details and interconnections of the SENT module.

The SENT module is supplied with a separate clock control, address decoding, and interrupt control logic. Outputs of the GTM module are connected to the 6 timer inputs.

The serial data inputs of the receive channels of the SENT module as well as the SPC outputs (SPCn) are connected to GPIO lines. If SPC outputs are used, they are usually mapped to the same port pin like the referring SENT data input line as this minimizes the pin count requirement.

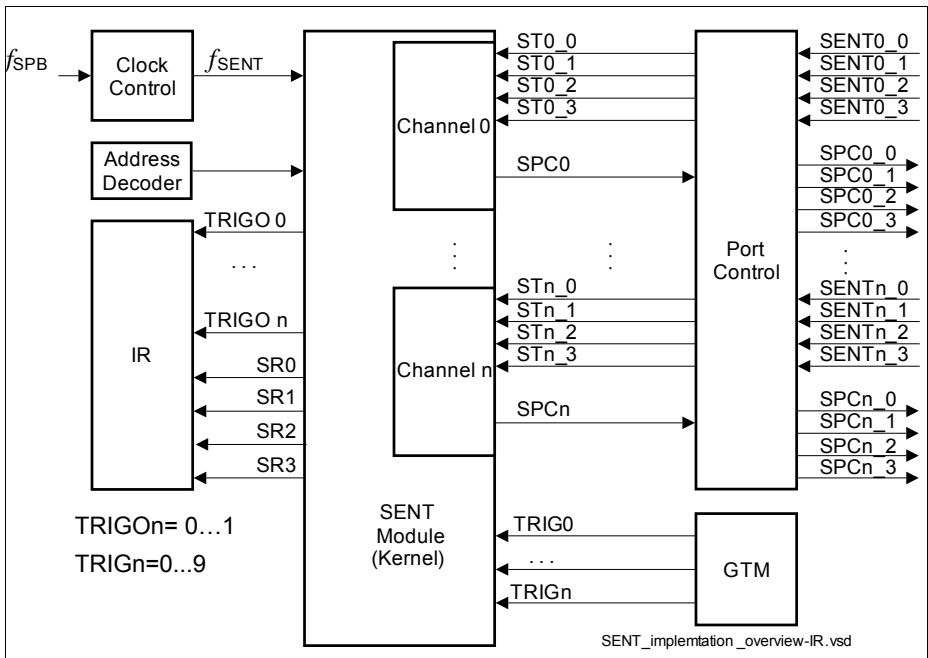


Figure 24-17 SENT Module Implementation and Interconnections

Single Edge Nibble Transmission (SENT)
24.3.1.1 On-Chip Connections

This section describes the on-chip connections of the SENT module.

24.3.1.2 Interrupt and DMA Controller Service Requests

The trigger outputs of the SENT module are connected via the Interrupt router. The request lines are connected as shown in [Table 24-9](#).

Table 24-9 Service Request Lines of SENT

INP value	Request Line	Connected to	Description
0000 _b	SR0	SRC_SENT0	Interrupt Router SENT Request 0
0001 _b	SR1	SRC_SENT1	Interrupt Router SENT Request 1
0010 _b	SR2	SRC_SENT2	Interrupt Router SENT Request 2
0011 _b	SR3	SRC_SENT3	Interrupt Router SENT Request 3
0100 _b	TRIGO0	SRC_SENT4	Interrupt Router SENT Request 4
0101 _b	TRIGO1	SRC_SENT5	Interrupt Router SENT Request 4
0110 _b	TRIGO2	SRC_SENT6	Not Connected
0111 _b	TRIGO3	SRC_SENT7	Not Connected
1000 _b	TRIGO4	SRC_SENT8	Not Connected
1001 _b	TRIGO5	SRC_SENT9	Not Connected
1010 _b	TRIGO6	SRC_SENT10	Not Connected
...
1111 _b	TRIGO11	SRC_SENT15	Not Connected

Trigger Inputs

The module has 6 Sent Channels and the same number of trigger inputs which can be randomly chosen by programming IOCRx.ETS. The trigger inputs (TRIG[5:0]) of the SENT module are connected to the GTM as shown in [Table 24-10](#). n = [0 .. 5]

Table 24-10 Trigger Input Lines of SENT

Request Line	Connected to	Description
TRIG0	TRIG0	GTM.ADC_0_TRIG_0
TRIG1	TRIG1	GTM.ADC_1_TRIG_0
...

Single Edge Nibble Transmission (SENT)**Table 24-10 Trigger Input Lines of SENT**

Request Line	Connected to	Description
TRIG4	TRIG4	GTM.ADC_4_TRIG_0
TRIG5	TRIG5	GTM.DSADC_0_TRIG_0

Single Edge Nibble Transmission (SENT)

24.3.2 SENT Module-Related External Registers

The registers listed in [Figure 24-18](#) are external of the SENT module kernel but must be programmed for proper operation of the SENT module.

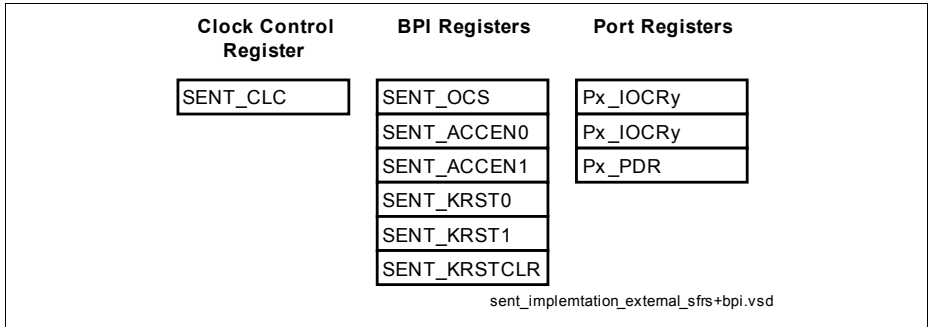


Figure 24-18 SENT Implementation-specific Special Function Registers

24.3.2.1 Port Control

The SENT input channels are overlaid with standard ADC channels as they are replacing former analog signals. Each channel is connected to two ADC channels that can be chosen alternatively. In addition each channel is connected to general purpose I/O lines as well. Each channel is connected to two different general purpose I/O lines that can be chosen alternatively. The selection from on of these 4 alternatives is done by application SW by programming SENT_IOCRx.ALTi respectively.

The SPC output lines are connected to the same I/O ports as the referring input channel. The ADC channels do not provide output functionality. The SPC I/O ports are controlled in the port logic (see also [Figure 24-17](#)). The following port control operations and selections must be executed for these I/O lines:

- Input/output function selection (Port IOCR registers)
- Pad driver characteristics selection for the outputs (Port PDR registers)

Input/Output Function Selection

SENT can be used in three different ways:

- ADC input (dedicated for ADC) overlaid with SENT digital input and no output option on this pin.
- SENT configured digital I/O lines (open drain, no push/pull, uses SDIR, single pin hardware direction control HW_DIR, SDIR is controlled by the data value: 0 = active out, 1 = passive/open drain, input)

Single Edge Nibble Transmission (SENT)

- Standard general purpose input/output function on two lines for each channel (input on ADC alternative 1 or 2, or on I/O port alternative 1 while the output is chosen to be on I/O ports alternative 2).

The SENT module overlays dedicated analog to digital converter (ADC) pins as digital input port.

SENT physical layer uses 5V signal voltage levels. All SENT inputs are mapped to 5V compatible ADC pads. V_{ddm} must be supplied with 5V for SENT operation.

SENT physical layer uses 5V signal voltage levels. All SENT inputs are mapped to 5V compatible ADC pads. V_{ddm} must be supplied with 5V for SENT operation.

The SENT module can be configured to use input/output ports configured for use with SENT. These control settings for the port pins differ from the standard general purpose I/O lines in so far as

- they are controlled by the SENT module output data via their HW_DIR line
- they are configured to use no push/pull devices and to work in open drain mode if the output function is selected by the HW_DIR line of the port

The SENT module can be configured to use standard (push pull) general purpose I/O lines as well. Different port pins can be selected for input and for output. This allows the use of external transceiver devices.

The port input/output control registers contain the bit fields that select the digital output and input driver characteristics such as port direction (input/output) with alternate output selection, pull-up/down devices, and open-drain selections. The I/O lines for the SENT module are controlled by the Port input/output control registers shown below.

Table 24-11 shows an overview how bits and bit fields must be programmed for the required I/O functionality of the SENT I/O lines.

Table 24-11 SENT I/O Control Selection and Setup

SENT Channel	Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
0	SENT0A / P40.0	SENT_IOCRR0.ALTI = 0000 _B	P40_PDISC.PDIS0 = 0 _B	I
	SENT0B / P00.1	SENT_IOCRR0.ALTI = 0001 _B	P00_IOCRR0.PC1 = 0XXXX _B	I
	SENT0C / P02.8	SENT_IOCRR0.ALTI = 0010 _B	P02_IOCRR8.PC8 = 0XXXX _B	I
		SENT_IOCRR0.ALTI = 0001 _B		O

Single Edge Nibble Transmission (SENT)
Table 24-11 SENT I/O Control Selection and Setup (cont'd)

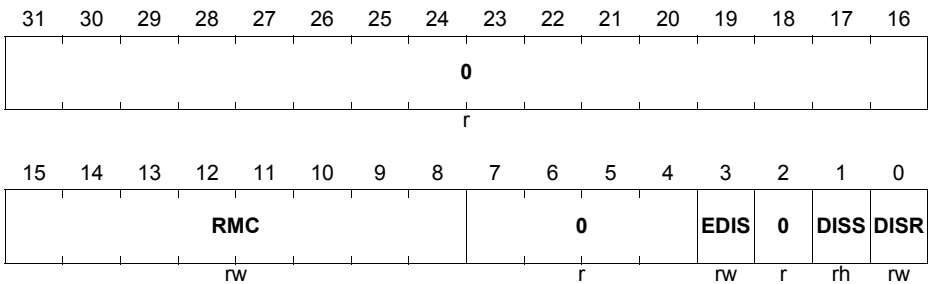
SENT Channel	Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
1	SENT1A / P40.1	SENT_IOC1.ALT1 = 0000 _B	P40_PDISC.PDIS1 = 0 _B	I
	SENT1B / P00.2	SENT_IOC1.ALT1 = 0001 _B	P00_IOC0.PC2 = 0XXXX _B	I
	SENT1C / P02.7	SENT_IOC1.ALT1 = 0010 _B	P02_IOC4.PC7 = 0XXXX _B	I
		SENT_IOC1.ALT1 = 0010 _B		O
2	SENT2A / P40.2 / AN26	SENT_IOC2.ALT1 = 0000 _B	P40_PDISC.PDIS2 = 0 _B	I
	SENT2B / P00.3	SENT_IOC2.ALT1 = 0001 _B	P00_IOC0.PC3 = 0XXXX _B	I
	SENT2C / P02.6	SENT_IOC2.ALT1 = 0010 _B	P02_IOC4.PC6 = 0XXXX _B	I
	SENT2D / P40.6 / AN36	SENT_IOC2.ALT1 = 0011 _B	P40_PDISC.PDIS6 = 0 _B	I
		SENT_IOC2.ALT1 = 0001 _B		O
3	SENT3A / P40.3 / AN27	SENT_IOC3.ALT1 = 0000 _B	P40_PDISC.PDIS3 = 0 _B	I
	SENT3B / P00.4	SENT_IOC3.ALT1 = 0001 _B	P00_IOC4.PC4 = 0XXXX _B	I
	SENT3C / P02.5	SENT_IOC3.ALT1 = 0010 _B	P02_IOC4.PC5 = 0XXXX _B	I
	SENT3D / P40.7 / AN37	SENT_IOC3.ALT1 = 0011 _B	P40_PDISC.PDIS7 = 0 _B	I
		SENT_IOC3.ALT1 = 0001 _B		O

Single Edge Nibble Transmission (SENT)
Table 24-11 SENT I/O Control Selection and Setup (cont'd)

SENT Channel	Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
4	SENT4A / P40.8	SENT_IOCR4.ALTI = 0000 _B	P40_PDISC.PDIS8 = 0 _B	I
	SENT4B / P00.5	SENT_IOCR4.ALTI = 0001 _B	P00_IOCR4.PC5 = 0XXXX _B	I
	SENT4C / P33.6	SENT_IOCR4.ALTI = 0010 _B	P33_IOCR4.PC6 = 0XXXX _B	I
		SENT_IOCR4.ALTI = 0001 _B		O
5	SENT5A / P40.9	SENT_IOCR5.ALTI = 0000 _B	P40_PDISC.PDIS9 = 0 _B	I
	SENT5B / P00.6	SENT_IOCR5.ALTI = 0001 _B	P00_IOCR4.PC6 = 0XXXX _B	I
	SENT5C / P33.5	SENT_IOCR5.ALTI = 0010 _B	P33_IOCR4.PC5 = 0XXXX _B	I
		SENT_IOCR5.ALTI = 0001 _B		O

Single Edge Nibble Transmission (SENT)
24.3.3 BPI_FPI Module Registers
Clock Control Register (CLC)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI_FPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the module clock, sleep mode and disable mode for the module.

SENT_CLC
Clock Control Register
(00_H)
Reset Value: 0000 0003_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
EDIS	3	rw	Sleep Mode Enable Control Used to control module's sleep mode.
RMC	[15:8]	rw	8-bit Clock Divider Value in RUN Mode
0	[31:16], [7:4], 2	r	Reserved Read as 0; should be written with 0.

Note: The number of module clock cycles (wait states) which are required by the kernel to execute a read or write access depends on the selected CLC clock frequency, which is selected via bit field RMC in the CLC register. Therefore, increasing CLC.RMC may result in a longer FPI Bus read cycle access time for kernel registers and can also slow down the write throughput to the kernel registers.

Single Edge Nibble Transmission (SENT)

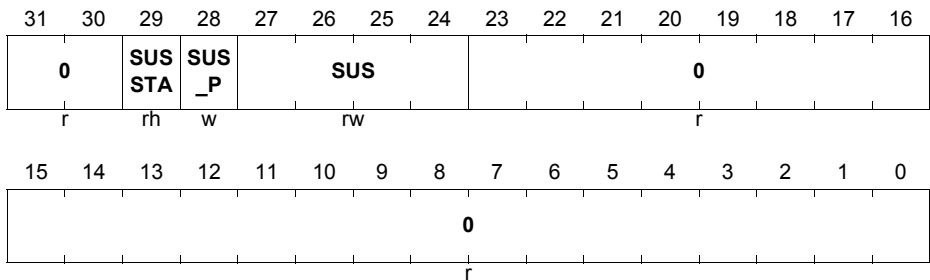
15. After a hardware reset operation, the f_{SENT} and $f_{fracdiv}$ clocks are switched off and the SENT module is disabled (DISS set).

OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset. The OCS register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS register value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32bit wide only and requires Supervisor Mode.

OCS

OCDS Control and Status (E8_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately. (SPC low pulse breaks immediately, Port Pin keeps the last value if suspend state is entered) 2 _H Soft suspend (SPC low pulse will be finished before suspend state is entered) others , reserved
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:0], [31:30]	r	Reserved Read as 0; must be written with 0.

Single Edge Nibble Transmission (SENT)
Access Enable Register (ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ... , EN31 -> TAG ID 011111_B.

ACCEN0
Access Enable Register 0
(FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN3	EN3	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN1	EN1	EN1	EN1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN1	EN1	EN1	EN1	EN1	EN1	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
5	4	3	2	1	0										
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
ENn (n = 0-31)	n	rW	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

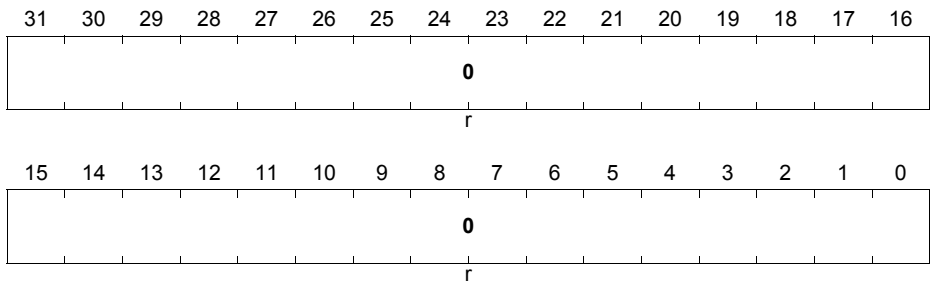
Access Enable Register (ACCEN1)

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

Single Edge Nibble Transmission (SENT)

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 10000_B, EN1 -> TAG ID 100001_B, ... , EN31 -> TAG ID 111111_B.

ACCEN1
Access Enable Register 1
(F8_H)
Reset Value: 0000 0000_H


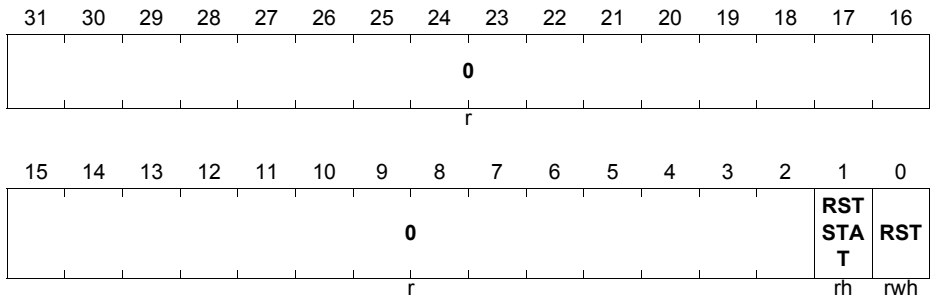
Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLE.CLR register bit.

Note: During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

Single Edge Nibble Transmission (SENT)
KRST0
Kernel Reset Register 0
(F4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
RSTSTAT	1	rh	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits. 0 _B No kernel reset was executed 1 _B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

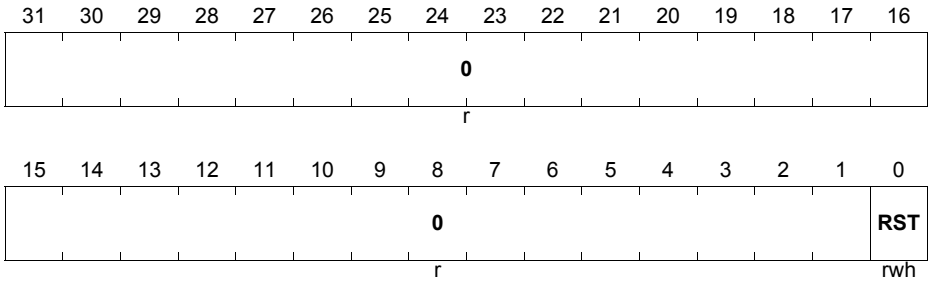
Single Edge Nibble Transmission (SENT)

KRST1

Kernel Reset Register 1

(F0_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set.</p> <p>0_B No kernel reset was requested</p> <p>1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p>
0	[31:1]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Kernel Reset Status Clear Register (KRSTCLR)

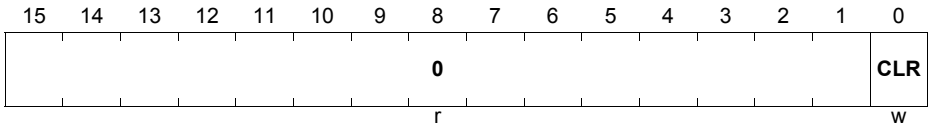
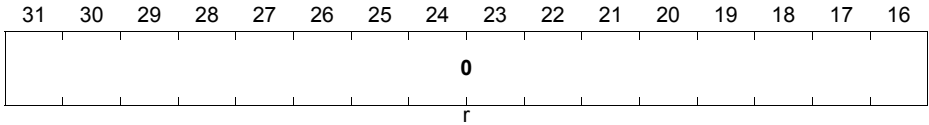
The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

Single Edge Nibble Transmission (SENT)

KRSTCLR

Kernel Reset Status Clear Register (EC_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Single Edge Nibble Transmission (SENT)
24.4 Revision History

This User's Manual is based on: SAE Standard "SENT Revision J2716 JAN2010".

Table 24-12 Revision History

Version Number	Changes to Previous Version
Rev_0.1D	First Draft (Bullet Points ++)
Rev_0.2D	Second Draft (Bullet Points ++)
Rev_0.3D	First Review Version
Rev_0.4D	<ul style="list-style-type: none"> • RCRx.NINT removed as unnecessary FIFO support • Spelling Checked • Renamed INTNP to INP (request DE) • One common error node pointer in INP for FRI, FDI, NNI, NVI, CRCI, WSI, SCRI (reduce DE effort) • Interrupt Node Pointer in INPx increased from 2 bit to 4 bit • Renamed PVAL to PDIV in register CPDRx • RCRx.ASP "Additional Sync/Cal Pulse expected" deleted, not required • RCRx.ACE "Alternate CRC Mode" added, request customer • cleanup Register SCR • ISR0 (Input select register) deleted • AltIn and Filter DEPTH moved to IOCRx • SCRx.DEL moved to SDRx • SCRx deleted, BASE and TRQ moved to SDR • Restructured Chapters (interrupts after functional registers, implementation chapter cleaned up) • Simplified Baud rate generation (Fdtick removed)
Rev_0.5D	<ul style="list-style-type: none"> • Clean up Glitch filter Chapter • Detailed Baud Rate Generation • Added Address overview table

Single Edge Nibble Transmission (SENT)

Table 24-12 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_0.6D	<ul style="list-style-type: none"> • Channel Fractional Divider with variable divider instead of variable dividend • Glitch filter based on F_pdiv and no longer on F_sent • Glitch detection added • Pre divider = PDIV and no longer PDIV + 1 • Compressed address overview table • Interrupt generation updated with figure • Trigger outputs renamed (RSI and SDI, no more TRIGO) and completed • RDI simplified - now independent from RSI • ETS wording corrected • INTOV now only updated for enabled interrupts (before: any change) • Renamed "SPC Data Register SDR" to "SPC Control Register SCR", it contains both data and control information • TDI wording optimized • External Registers Overview corrected • KSCCFG removed
Rev_1.0D	First complete revision (To be done)
Rev_1.1D	<ul style="list-style-type: none"> • Moved Base Address to F032_1000 - F032_19FF (10x256 bytes) • Serial message ID added. SDSx.SCN moved to [19:16], SDSx.SCRC moved to [15:12], SDSx.MID placed at [11:8]. • RSI and TDI as trigger outputs replaced by 8 Trigger Outputs programmable in the INPs. • implemented first port mapping proposal from product
Rev_1.2	<ul style="list-style-type: none"> • Added hint to RCR.CEN that FSMs are initialized during enable • Port mapping updated to latest changes of CW08/47

Single Edge Nibble Transmission (SENT)
Table 24-12 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.3	<ul style="list-style-type: none"> • Fixed INTENx Bit descriptions: all bits are rw and high active (typo) • added to feature list: “Programmable Nibble sorting to support LSN or HSN first and relief CPU” (was missing) • SENT_FDR.DM[1:0] added description of all 4 states of this bit field. (same as system level) • IOCR.DEPTH corrected f_{SENT} sent to f_{pdiv} (longer glitch tolerance) • Add Freeze-Disable Bit [10] to SENT_FDR (system requirement) • Pre divider factor is (PDIV + 1) and no longer PDIV (ease design) • Resulting Channel Fractional Divider is (DIV + 1) and no longer DIV • Add IOCR.TXM, RXM and TRM monitor bits (ease of validation) • RDRx added explanation: unused nibbles are shown as '0' (for clarity) • RDI and RSI: Read clears these bits NOT (request from AE) • Renamed Transmit Buffer Overflow to UNDER-Flow with change to respective functionality (data completely transmitted without new write) • RCRx.CFC: Wording improved and details added
Rev_1.4	<ul style="list-style-type: none"> • Added column Access Modes to Register Table • Updated Figure 24-2, Figure 24-3, Figure 24-4 (beautification) • The clock signal f_{pdiv} of a channel must always be at least 20 (old value 10) times the nominal tick frequency, to cater for worst case. • INPx: Added reserved values • Corrected interrupt structure figure, no more separate DMA TRIGO lines. • RCRx, VIEWx and CFDRx Reset value corrected to 0x0000 0000 hex • Added explanations to RDRx and VIEWx • AI00050230 - SENT_FDR register located at “unusual” address: Moved CLC to offset 0x0Ch for consistency with other modules • Reduced ETS to 3 bit and ALTI to 2 bit width • “Zero Nibble Insertion in CRC” covered. See RCR.CRZ. • UTP AI00050205 - “Extended Serial Frame” covered. Added description of extended serial frame. • SDS.SCN moved to RSR.SCN (Allows for an aligned representation of ID and Data as well as a fixed position for SCN) • SDS.SD, SDS.MID, SDS.SCRC enlarged (to 16, 8 and 6 bit width) • SDS.CON and RCR.ESF added • Resulting Channel Fractional Divider is DIV and no longer (DIV + 1)

Single Edge Nibble Transmission (SENT)
Table 24-12 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.5	<ul style="list-style-type: none"> • Corrected TBix description on Page 50. • SCR.TRQ corrected (deleted “Reads back zero”) • Added hint to definition of Alternate CRC Mode (ACE) • Detailed RBI behavior. It is set if kernel wants to set RSI or RDI and finds RSI or RDI already set. • Detailed that RCR.CEN resets the receiver state machines only while SCR.TRIG resets the sender state machines. • RCR.WSI updated wrt. the special case “extended serial frame” (2009-06-24) • INSTAT.SCRI detailed wrt.: CRC check must include check for correct 0 values in Extended Serial Frame Format (2009-06-24) • Message tick time prolonged to 90 μs according to new standard (2009-06-24) • RCR.CFC (successive calibration pulse detection) adopted to new standard. (2009-06-24) • Updated INTSTAT.FDI wrt.: new additional error diagnostics (total frame length variation and ration calibration pulse/total frame length) in new standard. (2009-06-24)
Rev_1.6	<ul style="list-style-type: none"> • Added bit RCR.IDE (Ignore Drift Error) to cater for rare triggers by SPC. • Removed SV protection from SENT_FDR. • VIEW.RDNP must be set before reception. • SCR.TRQ spec' ed not active from request to pulse start. • Added “alternative CRC” spec from TLE4998 . • Changed wording for RBI. • New additional error diagnostics (total frame length variation and ration calibration pulse/total frame length) removed.
Rev_1.7	<ul style="list-style-type: none"> • Added note: RCR.CEN must be cleared before changing CPDR.PDIV or CFDR.DIV. • SCRI typo: frame 8, not 7 contains a zero value. • Chapter 24.1.5 typo corrected: replaced f_{SENT} with $f_{fracdiv}$ • Chapter 24.1.3.1 note added on <i>RCR.CRZ</i>. • Added hint to Chapter “Module Clock Generation” for details on CLC. • Updated Port connections and Trigger Output connections
Rev_1.8	<ul style="list-style-type: none"> • Changed functionality of VIEW. • Changed functionality of WSI.
Rev_1.9	<ul style="list-style-type: none"> • Added Time Stamping (RTSx, TSR, TPD). • SRCs removed for TC27x family

Single Edge Nibble Transmission (SENT)
Table 24-12 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.10	<ul style="list-style-type: none"> Added Tagging
Rev_1.11	<ul style="list-style-type: none"> Added new BPI Registers
Rev_1.12	<ul style="list-style-type: none"> Configured for 10 Channels, document automatization improved
Rev_1.13	<ul style="list-style-type: none"> Writing to TPD clears TSR. 10 interrupt sources available for TC27x Added port mapping
Rev_1.14	<ul style="list-style-type: none"> added Watch Dog Timer (Registers WDTx) added new interrupt source for WDT: WDI added new interrupt node pointer in INPx for WDI added Edge Counter and Edge Counter Clear bit in IOCR changed Time Stamp Trigger Qualifier (see register RTS)
Rev_1.15	<ul style="list-style-type: none"> Modified RCRx (x = 0-5).CRZ bit definitions: CRC with Zero Nibble for Serial Data This bit selects the CRC method. If CRZ is cleared, augmentation is selected, (i.e a ZERO NIBBLE is added at the end of CRC calculation (only in calculation)). E.g. as 7th nibble (in case of 6 data nibbles) 00-Augmentation is selected for both 4-bit message CRC and the 4-bit CRC of the serial messages (legacy, 16 frames) 01-Augmentation is switched off for both 4-bit message CRC and the 4-bit CRC of the serial messages (legacy, 16 frames) The extended serial message (18 Frames, 6-bit CRC) is not controlled by CRZ but the 6-bit CRC is always augmented according to standard.
Rev_1.16	<ul style="list-style-type: none"> Modified IOCRx (x = 0-5).CEC bit moved from bit position 2 to bit position 10.
Rev_1.17	<ul style="list-style-type: none"> Modified IOCRx (x = 0-5).EC bit from rw to rh. Corrected on Watch dog Timer Register, statement from “The internal Watch Dog timer is cleared and started automatically each time an RDI (Receive Data Interrupt) is issued on the referring channel” to The internal Watch Dog timer is cleared and started automatically each time an RDI (Receive Data Interrupt Request Flag) is set in the INTSTAT of the referring channel.
Rev_1.18	<ul style="list-style-type: none"> Modified IOCRx (x = 0-5) reset value to X000 0000 with additional text on IOCRx.CTR, IOCRx.TRM reset value is 0 and IOCRx.TXM, IOCRx.RXM reset value is X.

Single Edge Nibble Transmission (SENT)
Table 24-12 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.19	<ul style="list-style-type: none"> • Removed on Figure 1-1, Figure 1-26 obsolete signal HW_DIR. • Added List of Access Protection Abbreviations on Table 24-8 • Modified on Table 24-8, SENT_INTSTAT, SENT_RTS, SENT_INTOV, SENT_RSR, write access to BE (Bus Error). • Modified on Table 24-8, for reserved locations, read/write accesses cause BE (bus error), SENT_ID write access causes BE (bus error).
Rev_1.20	<ul style="list-style-type: none"> • Modified on Table 24-8, for SENT_SDS write access causes BE (bus error). • Modified on Table 24-8, for SENT_RDR write access causes BE (bus error)
Rev_1.21	<ul style="list-style-type: none"> • Corrected on Table 24-8 for SENT_RDRx register offset address to $80_H - 80_H + x * 4_H$. • Updated Table 24-11 SENT I/O Control Selection and Setup
Rev_1.22	<ul style="list-style-type: none"> • Modified text on OCS (OCDS Control and Status Register) to “The OCDS Control and Status (OCS) register is cleared by Debug Reset. The OCS register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS register value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32bit wide only and requires Supervisor Mode.” • Modified for OCS.SUS for bit F_H removed and added others reserved (will not suspend)
Rev_1.23	<ul style="list-style-type: none"> • Updated Table 24-11 SENT I/O Control Selection and Setup
Rev_1.24	<ul style="list-style-type: none"> • Corrected on Table 1-3 Service Request Lines for SENT, for last option 1111_B to TRIGO11, SRC_SENT15 • Modified on Table 1-3 Service Request Lines for SENT, for TC26x to 6 interrupts, also on INPx.RSI bits definition.
Rev_1.25	<ul style="list-style-type: none"> • Corrected on Table 1-7 Sent I/O Control Selection & Setup SENT3D mapped to AN37, SENT2D mapped to AN36, SENT2A mapped to AN26, SENT 3A mapped to AN27.
Rev_1.26	<ul style="list-style-type: none"> • This bulleted point is not applicable to TC24x, TC26x, TC29x.

Single Edge Nibble Transmission (SENT)
Table 24-12 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.27	<ul style="list-style-type: none"> • Modified under Table1-6 SENT I/O Control Selection and Setup, for SENT channel1 in SPC mode (SPC1/P02.7) Input/Output control register bits from P00_IOCR0.PC1 = 1X110_B to P02.IOCR4.PC7 = 1X110_B. • Added under Table 1-2 Registers Overview - SENT Kernel Registers, Write Access Mode with 'P' for SENT_CLC, SENT_FDR, SENT_TPD, SENT_OCS, SENT_ACCEN0, SENT_ACCEN1, SENT_KRST0, SENT_KRST1, SENT_KRSTCLR, SENT_CPDRx, SENT_CFDRx, SENT_RCRx, SENT_IOCRx, SENT_SCRx, SENT_VIEWx, SENT_INTSETx, SENT_INTCLR, SENT_INTENx, SENT_INPx, SENT_WDTx, SENT_SRC3, SENT_SRC2, SENT_SRC1, SENT_SRC0 registers.
Rev_1.28	<ul style="list-style-type: none"> • This bulleted point is not applicable for TC1798, TC24x, TC26x, TC29x. • Changes done for this revision is not applicable for TC1798, TC24x, TC26x, TC29x.
Rev_1.29	<ul style="list-style-type: none"> • Updated on Watch Dog Timer Register with addition of statement - and also on any write to WDL that sets WDL>0 to "The internal Watch Dog timer is cleared and started automatically each time an RDI (Receive Data Interrupt Request Flag) is set in the INTSTAT of the referring channel and also on any write to WDL that sets WDL>0."
Rev_1.30	<ul style="list-style-type: none"> • This bulleted point is not applicable for TC27x, TX29x, TC26x. • Changes done for this revision is not applicable for TC27x, TX29x, TC26x.
Rev_1.31	<ul style="list-style-type: none"> • Added note on Section under Port Control "<i>SENT physical layer uses 5V signal voltage levels. All SENT inputs are mapped to 5V compatible ADC pads. Vddm must be supplied with 5V for SENT operation.</i>"
Rev_1.32	<ul style="list-style-type: none"> •

Single Edge Nibble Transmission (SENT)
Table 24-12 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.33	<ul style="list-style-type: none"> • Modified for below: <ul style="list-style-type: none"> – Added Note on INTOV, Interrupt Overview Register “Not all IPC0-9 are available, the number of Interrupt Pending on Channel is equivalent to the SENT channels available, e.g 4 SENT channels has 4 Interrupt Pending IPC0-3 and vice versa.” – Replaced “Fig 1-7 Standard Serial Data Encoding” in .emf with same “Table 1-1 Standard Serial Data Encoding” as framemaker table. – Replaced “Fig 1-8 Serial Data Frame” in .emf with same “Table 1-2 “Serial Data Frame” as framemaker table. – Replaced “Fig 1-9 Extended Serial Data Encoding” in .emf with same “Table 1-3 Extended Serial Data Encoding” as framemaker table. – Replaced “Fig1-11 Extended Serial Data Frame” in .emf with same “Table 1-4 Extended Serial Data Frame” – Replaced “Fig 1-12 Configuration Bit = 0” in .emf with same “Table 1-5 Configuration Bit = 0” – Replaced “Fig 1-13 Configuration Bit = 1” in .emf with same “Table 1-6 Configuration Bit = 1”
Rev_1.34	Changed minimum tick timer to 0.2 μ s For TC29x only: <ul style="list-style-type: none"> • Removed SENT2D from P40.6 • Moved SENT5A from P40.9 to P40.5 and SENT4A from P40.8 to P40.4 where available in product
Rev_1.34a	Updated Serial Message Frame Chapter

Single Edge Nibble Transmission (SENT)

25 FlexRay™ Protocol Controller (E-Ray)

The E-Ray IP-module performs communication according to the FlexRay™ ¹⁾ protocol specification v2.1, developed for automotive applications. With maximum specified clock the bitrate can be programmed to values up to 10 Mbit/s. Additional bus driver (BD) hardware is required for connection to the physical layer.

25.1 E-Ray Kernel Description

Figure 25-1 shows a global view of the E-Ray interface.

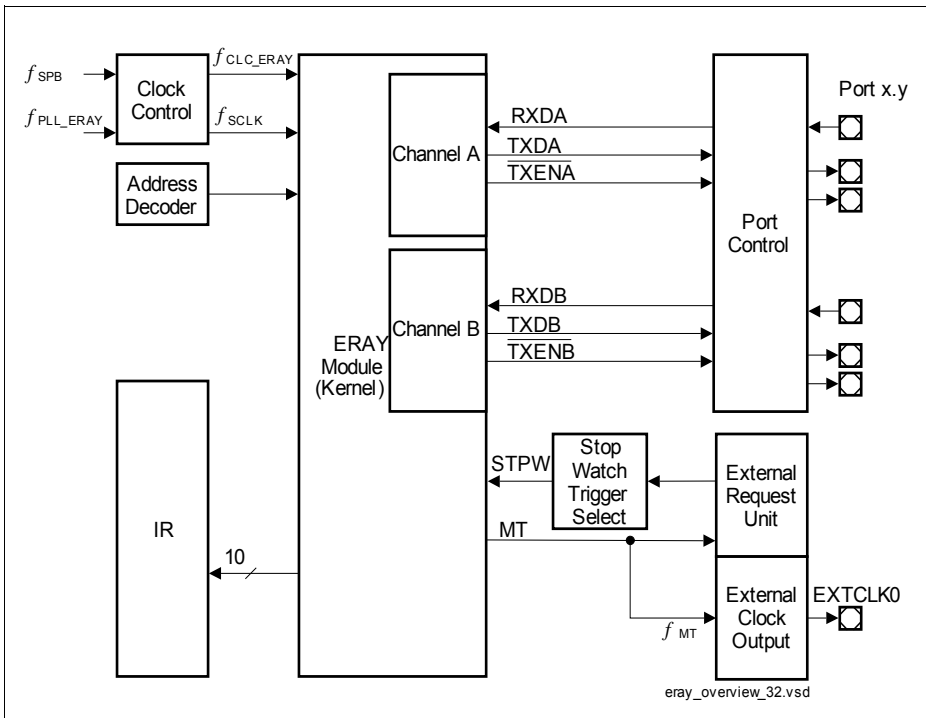


Figure 25-1 General Block Diagram of the E-Ray Interface

The E-Ray module communicates with the external world via three I/O lines each channel. The RXDA_x and RXDB_x lines are the receive data input signals, TXDA and

1) Infineon®, Infineon Technologies®, are trademarks of Infineon Technologies AG. FlexRay™ is a trademark of FlexRay Consortium.

FlexRay™ Protocol Controller (E-Ray)

TXDB lines are the transmit output signals, $\overline{\text{TXENA}}$ and $\overline{\text{TXENB}}$ the transmit enable signals.

Clock control, address decoding, and service request control are managed outside the E-Ray module kernel.

25.2 Overview

For communication on a FlexRay™ network, individual Message Buffers with up to 254 data byte are configurable. The message storage consists of a single-ported Message RAM that holds up to 128 Message Buffers. All functions concerning the handling of messages are implemented in the Message Handler. Those functions are the acceptance filtering, the transfer of messages between the two FlexRay™ Channel Protocol Controllers and the Message RAM, maintaining the transmission schedule as well as providing message status information.

The register set of the E-Ray IP-module can be accessed directly by an external Host via the module's Host interface. These registers are used to control/configure/monitor the FlexRay™ Channel Protocol Controllers, Message Handler, Global Time Unit, System Universal Control, Frame and Symbol Processing, Network Management, Service Request Control, and to access the Message RAM via Input / Output Buffer.

The E-Ray IP-module supports the following features:

- Conformance with FlexRay™ protocol specification v2.1
- Data rates of up to 10 Mbit/s on each channel
- Up to 128 Message Buffers configurable
- 8 Kbyte of Message RAM for storage of e.g. 128 Message Buffers with max. 48 byte data field or up to 30 Message Buffers with 254 byte Data Sections
- Configuration of Message Buffers with different payload lengths possible
- One configurable receive FIFO
- Each Message Buffer can be configured as receive buffer, as transmit buffer or as part of the receive FIFO
- Host access to Message Buffers via Input and Output Buffer.
Input Buffer: Holds message to be transferred to the Message RAM
Output Buffer: Holds message read from the Message RAM
- Filtering for slot counter, cycle counter, and channel
- Maskable module service requests
- Network Management supported
- Four service request lines
- Automatic delayed read access to Output Command Request Register (OBCR) if a data transfer from Message RAM to Output Shadow Buffer (initiated by a previous write access to the OBCR) is ongoing.
- Automatic delayed read access to Input Command Request Register (IBCR) if a data transfer from Input Shadow Buffer to Message RAM to (initiated by a previous write access to the IBCR) is ongoing.

FlexRay™ Protocol Controller (E-Ray)

- Four Input Buffer for building up transmission Frames in parallel.
- Flag indicating which Input Buffer is currently accessible by the host.

25.3 Definitions

FlexRay™ Frame: Header Segment + Payload Segment

Message Buffer: Header Section + Data Section

Message RAM: Header Partition + Data Partition

Data Frame: FlexRay™ Frame that is not a NULL Frame

25.4 Block Diagram

The E-Ray is built up by the following main submodules:

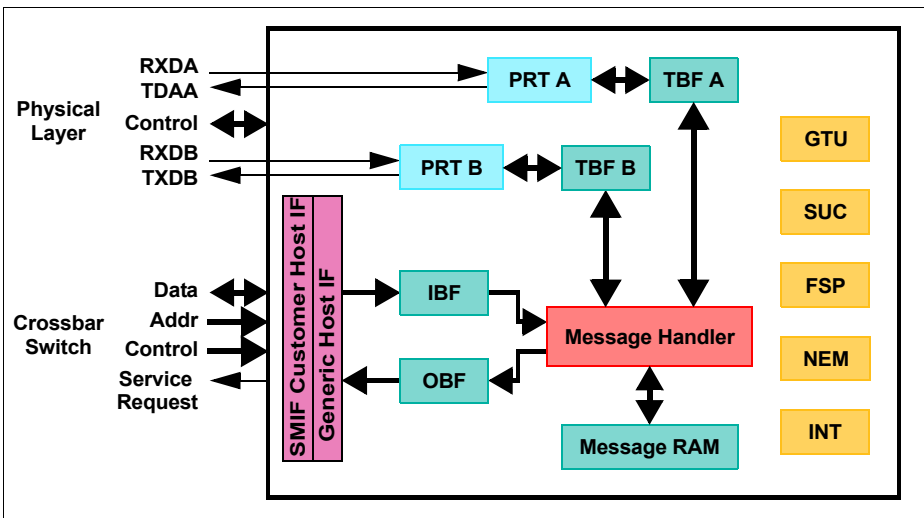


Figure 25-2 E-Ray Block Diagram

Customer Host Interface (CIF)

Connects the FPI Bus to the E-Ray IP-module via the Generic Host Interface.

Generic Host Interface (GIF)

The E-Ray IP-module is provided with an 8/16/32-bit Generic Host Interface prepared for the connection to a wide range of customer-specific Hosts. Configuration registers, status registers, and service request registers are attached to the respective blocks and can be accessed via the Generic Host Interface.

FlexRay™ Protocol Controller (E-Ray)**Input Buffer (IBF)**

For write access to the Message Buffers configured in the Message RAM, the Host can write the Header and Data Section for a specific Message Buffer to the Input Buffer. The Message Handler then transfers the data from the Input Buffer to the selected Message Buffer in the Message RAM.

Because the Input Buffer (IBF) Scheme does only allow to write the entire Message Frame, not only parts of it, the number of IBF has been increased from originally 2 to 4. This enables to fill the buffer partly and at the end request transfer into Message RAM. Therefore 2 extra bits allow to switch between the two banks of IBF and one status bit signals the IBF currently active for Host writes.

Output Buffer (OBF)

For read access to a Message Buffer configured in the Message RAM the Message Handler transfers the selected Message Buffer to the Output Buffer. After the transfer has completed, the Host can read the Header and Data Section of the transferred Message Buffer from the Output Buffer.

Message Handler (MHD)

The E-Ray Message Handler controls data transfers between the following components:

- Input / Output Buffer and Message RAM
- Transient Buffer RAMs of the two FlexRay™ Protocol Controllers and Message RAM

Message RAM (MRAM)

The Message RAM consists of a single-ported RAM that stores up to 128 FlexRay™ Message Buffers together with the related configuration data (Header and Data Partition).

Transient Buffer RAM (TBF 1/2)

Stores the Data Section of two complete messages.

FlexRay™ Channel Protocol Controller (PRT A/B)

The FlexRay™ Channel Protocol Controllers consist of shift register and FlexRay™ protocol FSM. They are connected to the Transient Buffer RAMs for intermediate message storage and to the physical layer via bus driver BD.

They perform the following functionality:

- Control and check of bit timing
- Reception and transmission of FlexRay™ Frames and symbols
- Check of Header CRC
- Generation / check of Frame CRC

FlexRay™ Protocol Controller (E-Ray)

- Interfacing to bus driver

The FlexRay™ Channel Protocol Controllers have interfaces to:

- Physical Layer (bus driver)
- Transient Buffer RAM
- Message Handler
- Global Time Unit
- System Universal Control
- Frame and Symbol Processing
- Network Management
- Service Request Control

Global Time Unit (GTU)

The Global Time Unit performs the following functions:

- Generation of Microtick
- Generation of Macrotick
- Fault tolerant clock synchronization by FTM algorithm
 - Rate correction
 - Offset correction
- Cycle counter
- Timing control of static segment
- Timing control of dynamic segment (minislottng)
- Support of external clock correction

System Universal Control (SUC)

The System Universal Control controls the following functions:

- Configuration
- Wakeup
- Startup
- Normal Operation
- Passive Operation
- Monitor Mode

Frame and Symbol Processing (FSP)

The Frame and Symbol Processing controls the following functions:

- Checks the correct timing of Frames and symbols
- Tests the syntactical and semantical correctness of received Frames
- Sets the slot status flags

Network Management (NEM)

Handles of the Network Management vector

Service Request Control (INT)

The Service Request Controller performs the following functions:

- Provides error and status service request flags
- Enables and disables service request sources
- Assignment of service request sources to one of the two module service request lines
- Enables and disables module service request lines
- Manages the two service request timers
- Stop watch time capturing

25.5 Programmer's Model

The programmer's model of the E-Ray module follows the principle of memory mapped peripheral. Some portion of the memory follows the principle of segmented/paged memory organization.

25.5.1 Register Map

The E-Ray module allocates an address space of 4 Kbyte (000_H to FFFF_H). The registers are organized as 32-bit registers. 8/16-bit accesses are also supported. Host access to the Message RAM is done via the Input and Output Buffers. They buffer data to be transferred to and from the Message RAM under control of the Message Handler, avoiding conflicts between Host accesses and message reception / transmission. Addresses 0004_H - 000F_H, 03C8_H - 03EC_H and 0800_H - 0FFF_H are reserved for customer specific purposes. All functions related to these addresses are located in the Customer Host Interface. The test registers located on address 0010_H and 0014_H are writable only under the conditions described in [“Special Registers” on Page 25-22](#).

The assignment of the Message Buffers is done according to the scheme shown in [Table 25-1](#) below. The number N of available Message Buffers depends on the payload length of the configured Message Buffers. The maximum number of Message Buffers is 128. The maximum payload length supported is 254 byte.

The Message Buffers are separated into three consecutive groups:

- Static Buffers: Transmit / Receive Buffers assigned to static segment
- Static and Dynamic Buffers: Transmit / Receive Buffers assigned to static or dynamic segment
- FIFO- Receive FIFO

The Message Buffer separation configuration can be changed only in “DEFAULT_CONFIG” or “CONFIG” state only by programming the Message RAM Configuration register (MRC).

The first group starts with Message Buffer 0 and consists of static Message Buffers only. Message Buffer 0 is dedicated to hold the startup / SYNC Frame or the single slot Frame, if node transmit one, as configured by SUCC1.TXST, SUCC1.TXSY, and SUCC1.TSM

FlexRay™ Protocol Controller (E-Ray)

in the SUC Configuration Register 1 (SUCC1). In addition, Message Buffer 1 may be used for SYNC Frame transmission in case that SYNC Frames or single-slot Frames should have different payloads on the two channels. In this case bit MRC.SPLM has to be programmed to 1 and Message Buffers 0 and 1 have to be configured with the key slot ID and can be (re)configured in “DEFAULT_CONFIG” or “CONFIG” state only.

The second group consists of Message Buffers assigned to the static or to the dynamic segment. Message Buffers belonging to this group may be reconfigured during run time from dynamic to static or vice versa depending on the state of MRC.SEC.

The Message Buffers belonging to the third group are concatenated to a single receive FIFO.

Table 25-1 Assignment of Message Buffers

Message Buffer 0	↓ Static Buffers	
Message Buffer 1		
...		
	↓ Static + Dynamic Buffers	← FDB
	↓ FIFO	← FFB
Message Buffer N-1		
Message Buffer N		← LCB

25.5.2 E-Ray Kernel Registers

This chapter describes all registers of the E-Ray kernel.

All registers in the E-Ray address spaces are reset with the application reset with one exception: OCS is reset with debug reset only. (Definition see SCU section “Reset Operation”)

Table 25-2 Registers Address Space E-Ray Kernel Register Address Space

Module	Base Address	End Address	Note
ERAY0	F001C000 _H	F001CFFF _H	4Kbyte

Table 25-3 Registers Overview E-Ray Kernel Registers

Register Short Name	Register Long Name	Offset Addr. ¹⁾	Access Mode		Description see
			Read	Write	

Customer Registers

CLC	Clock Control Register	0000 _H	U, SV	SV, E	Page 25-263
CUST1	Busy and Input Buffer Control Register	0004 _H	SV,U	SV,U	Page 25-16
ID	Module Identification Register	0008 _H	SV,U	BE	Page 25-15
CUST3	Customer Interface Timeout Counter	000C _H	SV,U	SV,U	Page 25-19

Special Registers

TEST1	Test Register 1	0010 _H	SV,U	SV,U	Page 25-22
TEST2	Test Register 2	0014 _H	SV,U	SV,U	Page 25-27
-	Reserved	0018 _H	BE	BE	-
LCK	Lock Register	001C _H	SV,U	SV,U	Page 25-30

Service Request Registers

EIR	Error Service Request Register	0020 _H	SV,U	SV,U	Page 25-32
SIR	Status Service Request Register	0024 _H	SV,U	SV,U	Page 25-38
EILS	Error Service Request Line Select	0028 _H	SV,U	SV,U	Page 25-43
SILS	Status Service Request Line Select	002C _H	SV,U	SV,U	Page 25-47

FlexRay™ Protocol Controller (E-Ray)
Table 25-3 Registers OverviewE-Ray Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset Addr. ¹⁾	Access Mode		Description see
			Read	Write	
EIES	Error Service Request Enable Set	0030 _H	SV,U	SV,U	Page 25-51
EIER	Error Service Request Enable Reset	0034 _H	SV,U	SV,U	Page 25-56
SIES	Status Service Request Enable Set	0038 _H	SV,U	SV,U	Page 25-61
SIER	Status Service Request Enable Reset	003C _H	SV,U	SV,U	Page 25-66
ILE	Service Request Line Enable	0040 _H	SV,U	SV,U	Page 25-71
T0C	Timer 0 Configuration	0044 _H	SV,U	SV,U	Page 25-72
T1C	Timer 1 Configuration	0048 _H	SV,U	SV,U	Page 25-74
STPW1	Stop Watch Register 1	004C _H	SV,U	SV,U	Page 25-76
STPW2	Stop Watch Register 2	0050 _H	SV,U	SV,U	Page 25-78
-	Reserved	0054 _H - 007C _H	BE	BE	-

Communication Controller Control Registers

SUCC1	SUC Configuration Register 1	0080 _H	SV,U	SV,U	Page 25-79
SUCC2	SUC Configuration Register 2	0084 _H	SV,U	SV,U	Page 25-87
SUCC3	SUC Configuration Register 3	0088 _H	SV,U	SV,U	Page 25-88
NEMC	NEM Configuration Register	008C _H	SV,U	SV,U	Page 25-89
PRTC1	PRT Configuration Register 1	0090 _H	SV,U	SV,U	Page 25-90
PRTC2	PRT Configuration Register 2	0094 _H	SV,U	SV,U	Page 25-92
MHDC	MHD Configuration Register	0098 _H	SV,U	SV,U	Page 25-93
-	Reserved	009C _H	BE	BE	-
GTUC01	GTU Configuration Register 1	00A0 _H	SV,U	SV,U	Page 25-94
GTUC02	GTU Configuration Register 2	00A4 _H	SV,U	SV,U	Page 25-95
GTUC03	GTU Configuration Register 3	00A8 _H	SV,U	SV,U	Page 25-96
GTUC04	GTU Configuration Register 4	00AC _H	SV,U	SV,U	Page 25-97
GTUC05	GTU Configuration Register 5	00B0 _H	SV,U	SV,U	Page 25-98
GTUC06	GTU Configuration Register 6	00B4 _H	SV,U	SV,U	Page 25-99

FlexRay™ Protocol Controller (E-Ray)
Table 25-3 Registers OverviewE-Ray Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset Addr. ¹⁾	Access Mode		Description see
			Read	Write	
GTUC07	GTU Configuration Register 7	00B8 _H	SV,U	SV,U	Page 25-100
GTUC08	GTU Configuration Register 8	00BC _H	SV,U	SV,U	Page 25-101
GTUC09	GTU Configuration Register 9	00C0 _H	SV,U	SV,U	Page 25-102
GTUC10	GTU Configuration Register 10	00C4 _H	SV,U	SV,U	Page 25-103
GTUC11	GTU Configuration Register 11	00C8 _H	SV,U	SV,U	Page 25-104
-	Reserved	00CC _H - 00FC _H	BE	BE	-

Communication Controller Status Registers

CCSV	Communication Controller Status Vector	0100 _H	SV,U	BE	Page 25-106
CCEV	Communication Controller Error Vector	0104 _H	SV,U	BE	Page 25-111
-	Reserved	0108 _H	nBE	BE	-
-	Reserved	010C _H	nBE	BE	-
SCV	Slot Counter Value	0110 _H	SV,U	BE	Page 25-112
MTCCV	Macrotick and Cycle Counter Value	0114 _H	SV,U	BE	Page 25-113
RCV	Rate Correction Value	0118 _H	SV,U	BE	Page 25-114
OCV	Offset Correction Value	011C _H	SV,U	BE	Page 25-115
SFS	SYNC Frame Status	0120 _H	SV,U	BE	Page 25-116
SWNIT	Symbol Window and Network Idle Time Status	0124 _H	SV,U	BE	Page 25-118
ACS	Aggregated Channel Status	0128 _H	SV,U	SV,U	Page 25-121
-	Reserved	012C _H	nBE	BE	-
ESIDnn	Even Sync ID Symbol Window nn	0130 _H - 0168 _H	SV,U	BE	Page 25-124
-	Reserved	016C _H	SV,U	BE	-
OSIDnn	Odd Sync ID Symbol Window nn	0170 _H - 01A8 _H	SV,U	BE	Page 25-126
-	Reserved	01AC _H	SV,U	BE	-

FlexRay™ Protocol Controller (E-Ray)
Table 25-3 Registers OverviewE-Ray Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset Addr. ¹⁾	Access Mode		Description see
			Read	Write	
NMVx	Network Management Vector [1...3]	01B0 _H - 01B8 _H	SV,U	BE	Page 25-128
-	Reserved	01BC _H - 02FC _H	nBE	BE	-

Message Buffer Control Registers

MRC	Message RAM Configuration	0300 _H	SV,U	SV,U	Page 25-129
FRF	FIFO Rejection Filter	0304 _H	SV,U	SV,U	Page 25-132
FRFM	FIFO Rejection Filter Mask	0308 _H	SV,U	SV,U	Page 25-134
FCL	FIFO Critical Level	030C _H	SV,U	SV,U	Page 25-135

Message Buffer Status Registers

MHDS	Message Handler Status	0310 _H	SV,U	SV,U	Page 25-136
LDTS	Last Dynamic Transmit Slot	0314 _H	SV,U	SV,U	Page 25-138
FSR	FIFO Status Register	0318 _H	SV,U	SV,U	Page 25-139
MHDF	Message Handler Constraints Flags	031C _H	SV,U	BE	Page 25-141
TXRQ1	Transmission Request Register 1	0320 _H	SV,U	BE	Page 25-144
TXRQ2	Transmission Request Register 2	0324 _H	SV,U	BE	Page 25-145
TXRQ3	Transmission Request Register 3	0328 _H	SV,U	BE	Page 25-146
TXRQ4	Transmission Request Register 4	032C _H	SV,U	BE	Page 25-147
NDAT1	New Data Register 1	0330 _H	SV,U	BE	Page 25-148
NDAT2	New Data Register 2	0334 _H	SV,U	BE	Page 25-149
NDAT3	New Data Register 3	0338 _H	SV,U	BE	Page 25-150
NDAT4	New Data Register 4	033C _H	SV,U	BE	Page 25-151
MBSC1	Message Buffer Status Changed 1	0340 _H	SV,U	BE	Page 25-152
MBSC2	Message Buffer Status Changed 2	0344 _H	SV,U	BE	Page 25-153
MBSC3	Message Buffer Status Changed 3	0348 _H	SV,U	BE	Page 25-154

FlexRay™ Protocol Controller (E-Ray)
Table 25-3 Registers Overview E-Ray Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset Addr. ¹⁾	Access Mode		Description see
			Read	Write	
MBSC4	Message Buffer Status Changed 4	034C _H	SV,U	BE	Page 25-155
-	Reserved	0350 _H - 03A4 _H	BE	BE	-
NDIC1	New Data Interrupt Control 1	03A8 _H	SV,U	SV,U	Page 25-272
NDIC2	New Data Interrupt Control 2	03AC _H	SV,U	SV,U	Page 25-273
NDIC3	New Data Interrupt Control 3	03B0 _H	SV,U	SV,U	Page 25-274
NDIC4	New Data Interrupt Control 4	03B4 _H	SV,U	SV,U	Page 25-275
MSIC1	Message Buffer Status Changed Interrupt Control 1	03B8 _H	SV,U	SV,U	Page 25-276
MSIC2	Message Buffer Status Changed Interrupt Control 2	03BC _H	SV,U	SV,U	Page 25-277
MSIC3	Message Buffer Status Changed Interrupt Control 3	03C0 _H	SV,U	SV,U	Page 25-278
MSIC4	Message Buffer Status Changed Interrupt Control 4	03C4 _H	SV,U	SV,U	Page 25-275

Identification Registers

CREL	Core Release Registers	03F0 _H	SV,U	nBE	Page 25-156
ENDN	Endian Register	03F4 _H	SV,U	nBE	Page 25-158
-	Reserved	03F6 _H - 03FC _H	BE	BE	-

Input Buffer

WRDSn	Write Data Section [1...64]	0400 _H - 04FC _H	SV,U	SV,U	Page 25-159
WRHS1	Write Header Section 1	0500 _H	SV,U	SV,U	Page 25-160
WRHS2	Write Header Section 2	0504 _H	SV,U	SV,U	Page 25-163
WRHS3	Write Header Section 3	0508 _H	SV,U	SV,U	Page 25-164
	Reserved	050C _H	BE	BE	
IBCM	Input Buffer Command Mask	0510 _H	SV,U	SV,U	Page 25-165
IBCR	Input Buffer Command Request	0514 _H	SV,U	SV,U	Page 25-167

FlexRay™ Protocol Controller (E-Ray)
Table 25-3 Registers Overview E-Ray Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset Addr. ¹⁾	Access Mode		Description see
			Read	Write	
	Reserved	0518 _H - 05FC _H	BE	BE	
Output Buffer					
RDDSn	Read Data Section [1...64]	0600 _H - 06FC _H	SV,U	BE	Page 25-169
RDHS1	Read Header Section 1	0700 _H	SV,U	BE	Page 25-170
RDHS2	Read Header Section 2	0704 _H	SV,U	BE	Page 25-172
RDHS3	Read Header Section 3	0708 _H	SV,U	BE	Page 25-174
MBS	Message Buffer Status	070C _H	SV,U	BE	Page 25-176
OBCM	Output Buffer Command Mask	0710 _H	SV,U	SV,U	Page 25-182
OBCR	Output Buffer Command Request	0714 _H	SV,U	SV,U	Page 25-185
	Reserved	0718 _H -	BE	BE	-
OTS Kernel Registers					
OTSS	OCDS Trigger Set Select	0870 _H	U, SV	SV, E	Page 25-262
BPI Kernel Registers					
OCS	OCDS Control and Status Register	08E8 _H	U, SV	SV, P	Page 25-265
KRSTCLR	Reset Status Clear Register	08EC _H	U, SV	SV, P	Page 25-270
KRST1	Reset Control Register 1	08F0 _H	U, SV	SV, P	Page 25-269
KRST0	Reset Control Register 0	08F4 _H	U, SV	SV, P	Page 25-268
ACCEN1	Access Enable Register 1	08F8 _H	U, SV	SV, SE	Page 25-267
ACCEN0	Access Enable Register 0	08FC _H	U, SV	SV, SE	Page 25-266
	Reserved	0900 _H - 0FFF _H	BE	BE	-

FlexRay™ Protocol Controller (E-Ray)

- 1) The absolute register address is calculated as follows:
Module Base Address + Offset Address (shown in this column)

Note: Registers covered by the ACCENx write protection mechanism are marked with 'P' in Access Mode, Write.

25.5.2.1 Customer Registers

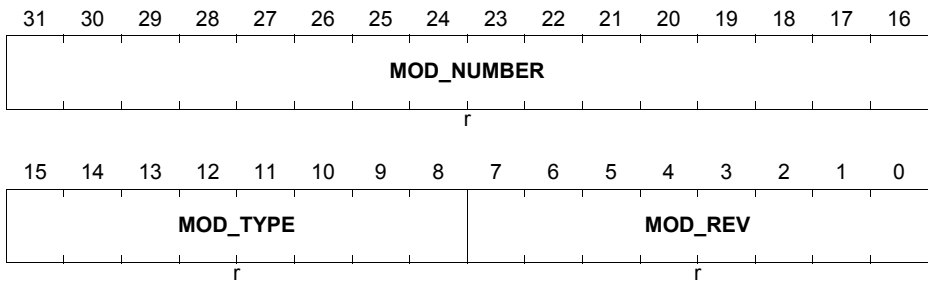
The addresses 0004_H - 000F_H, 03C8_H - 03EC_H and 0800_H - 0FFF_H are reserved for customer-specific registers.

Module Identification Register (ID)

This register contains bit fields identifying the E-Ray module in Infineons Module portfolio and is read only.

ID

Module Identification Register (0008_H) **Reset Value: 0044 C0XX_H**



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number MOD_REV defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MOD_TYPE	[15:8]	r	Module Type The value of this bit field is C0 _H . It defines the module as a 32-bit module.
MOD_NUMBER	[31:16]	r	Module Number Value This bit field defines a module identification number. For the E-Ray module the module identification number is 44 _H .

FlexRay™ Protocol Controller (E-Ray)

Busy Control Register (CUST1)

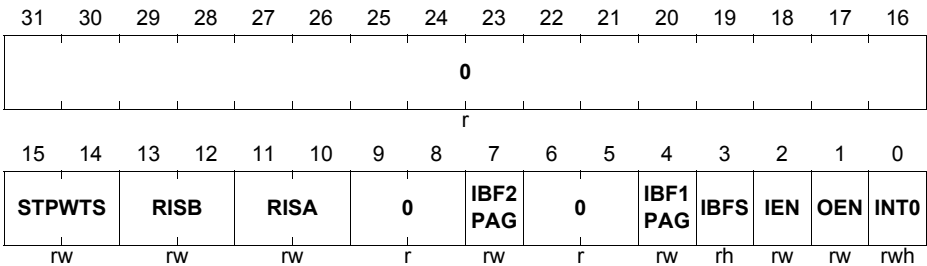
The Busy Control Register enables the automatic delay scheme. Furthermore it signals a time-out service request for the automatic delay scheme. IBFS changes 2 clock cycles after any reset to '1'. Thus the application will read this bit as '1' already at first access after any reset.

CUST1

Busy and Input Buffer Control Register

(0004_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
INT0	0	rwh	<p>CIF Timeout Service Request Status</p> <p>INT0 will be set if a timeout has occurred during the auto delay scheme and must be reset by writing zero to INTO.</p> <p><i>Note: In case hardware sets INTO and at the same point of time software clears INTO, INTO is cleared.</i></p>
OEN	1	rw	<p>Enable auto delay scheme for Output Buffer Control Register (OBCR)</p> <p>This control bit controls the delay scheme for Output Buffer Control Register (OBCR) read accesses.</p> <p>0_B Disable auto delay scheme for Output Buffer Control Register (OBCR)</p> <p>1_B Enable auto delay scheme for Output Buffer Control Register (OBCR)</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
IEN	2	rw	<p>Enable auto delay scheme for Input Buffer Control Register (IBCR)</p> <p>This control bit controls the auto delay scheme for Input Buffer Control Register (IBCR) read accesses.</p> <p>0_B Disable auto delay scheme for Input Buffer Control Register (IBCR)</p> <p>1_B Enable auto delay scheme for Input Buffer Control Register (IBCR)</p>
IBFS	3	rh	<p>Input Buffer Status Register</p> <p>This status bit indicates which of the two Input Buffer RAMs (IBF) is accessible by the host (via CIF) as Input Buffer. The other non accessible buffer RAM is currently used as shadow buffer RAM by the ERAY message handler and therefore not accessible by the host. After reset, it is set by hardware.</p> <p>0_B Input Buffer RAM 2 (IBF2) is accessible as Input Buffer by the host (CIF)</p> <p>1_B Input Buffer RAM 1 (IBF1) is accessible as Input Buffer by the host (CIF)</p>
IBF1PAG	4	rw	<p>Input Buffer 1 Page Select Register</p> <p>This control bit selects if the upper page or lower page of Input Buffer 1 (IBF1) currently active.</p> <p>0_B Read: Lower Page (256 Bytes) of Input Buffer RAM 1 selected Write: Select Lower Page (256 Bytes) of Input Buffer RAM 1</p> <p>1_B Read: Upper Page (256 Bytes) of Input Buffer RAM 1 selected Write: Select Upper Page (256 Bytes) of Input Buffer RAM 1</p> <p><i>Note: Write is only possible, if Input Buffer RAM 1 is currently accessible by the host (via CIF) and therefore IBFS set.</i></p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
IBF2PAG	7	rw	Input Buffer 2 Page Select Register This control bit selects if the upper page or lower page of Input Buffer 2 (IBF2) currently active. 0 _B Read: Lower Page (256 Bytes) of Input Buffer RAM 2 selected Write: Select Lower Page (256 Bytes) of Input Buffer RAM 2 1 _B Read: Upper Page (256 Bytes) of Input Buffer RAM 2 selected Write: Select Upper Page (256 Byte) of Input Buffer RAM 2 <i>Note: Write is only possible, if Input Buffer RAM 2 is currently accessible by the host (via CIF) and therefore IBFS cleared.</i>
RISA	[11:10]	rw	Receive Input Select Channel A 00 _B Channel A receiver input RXDA0 selected 01 _B Channel A receiver input RXDA1 selected 10 _B Channel A receiver input RXDA2 selected 11 _B Channel A receiver input RXDA3 selected
RISB	[13:12]	rw	Receive Input Select Channel B 00 _B Channel B receiver input RXDB0 selected 01 _B Channel B receiver input RXDB1 selected 10 _B Channel B receiver input RXDB2 selected 11 _B Channel B receiver input RXDB3 selected
STPWTS	[15:14]	rw	Stop Watch Trigger Input Select 00 _B Stop Watch Trigger input STPWT0 selected 01 _B Stop Watch Trigger input STPWT1 selected 10 _B Stop Watch Trigger input STPWT2 selected 11 _B Stop Watch Trigger input STPWT3 selected
0	[6:5], [9:8], [31:16]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

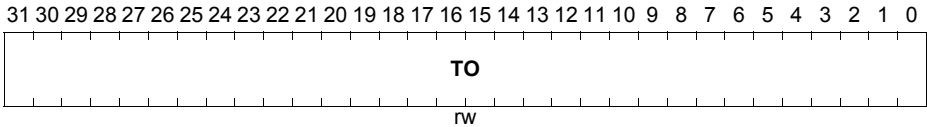
Customer Interface Time-out Counter Register (CUST3)

The Time-out Counter Register is realizing the time-out counter reload (startup) value for the automatic delay scheme (not the time-out down counter itself).

CUST3

Customer Interface Timeout Counter (000C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TO	[31:0]	rw	CIF Timeout Reload Value The 32-bit down counter reload (start-up) value must be setup for the automatic delay scheme.

Automatic Delayed Write Access to OBCR and IBCR

Write and read accesses to the Output Buffer Control Register (OBCR) can be automatically stalled due to a ongoing transfer from the Message Buffer to the Output Buffer. Also write and read accesses to the Input Buffer Control Register (IBCR) may be automatically delayed due to a ongoing transfer from the Input Buffer to the Message Buffer.

This delay scheme can be controlled (enabled or disabled) by CUST1.IEN and CUST1.OEN. The maximum time to stall a write or read access is determined by a single time-out counter precluded with the 32-bit value specified in the bit field CUST3.TO. If the time-out counter counts down to zero before the transfer to/from the Message Buffer is completed, the access (read or write) will be canceled and a service request will be generated. A canceled read access provides a 0 value. A canceled write access does not modify any bits in the OBCR or IBCR. In addition the bit CUST1.INT0 of the service request status register will be set and must be reset by the host to disable the service request line.

The read and write access to the Output Buffer Control Register (OBCR) may be configured without automatic delay by clearing CUST1.OEN. Setting OBCR.REQ and immediately afterwards reading or writing OBCR, e.g. to set OBCR.VIEW will lead to a canceled read or write operation, e.g. OBCR.VIEW remains cleared, and an error is signalled by a set EIR.IOBA. Besides canceling the erroneous read or write operation, and setting the error bit, no further state change happens. So full operation is granted. OBCR remains read and write inaccessible until the transfer of data from the Message Buffer to the Output Buffer (MBF⇒OBF) is completed. During this time span all read and

FlexRay™ Protocol Controller (E-Ray)

write accesses to the Output Buffer Control Register (OBCR) are canceled. The transfer is completed when OBCR.OBSYS is cleared. Additionally signal TOBC may be used, e.g. for service request triggering, DMA triggering, or driving a pin, to communicate the access status.

The read and write access to the Output Buffer Control Register (OBCR) may be configured to be automatic delayed by setting CUST1.OEN and configuring CUST3.TO to the maximum stall time acceptable to the system. If setting OBCR.REQ and immediately afterwards reading or writing to OBCR, e.g. to set the OBCR.VIEW bit, this read or write will be stalled until either the maximum delay time elapsed (in this case the read or write operation is cancelled after the stall time, e.g. OBCR.VIEW remains cleared, and an error is signalled by setting EIR.IOBA) or the read or write completes normally, e.g. set OBCR.VIEW after the transfer of data from the Message Buffer to the Output Buffer (MBF⇒OBF) is finalized. During this time the bus is locked and no further access to the E-Ray module is possible due to the ongoing stalled read or write operation. Because no access is possible to the E-Ray module, read or write stall may only be detected through the signal TOBC or due to other not processed read or write accesses to the E-Ray module.

The read and write access to the Input Buffer Control Register (IBCR) may also be configured without automatic delay by clearing CUST1.IEN. By writing to IBCR.IBRH the Input Buffers are swapped (shadow IBF changes to host IBF and host IBF to shadow IBF), the content of the shadow IBF is copied into the MBF (IBF⇒MBF), and IBCR.IBSYS is set. Writing to IBCR.IBRH a second time while IBCR.IBSYS remained set (previously initiated copy process IBF⇒MBF ongoing) will correctly update IBCR.IBRH and set IBCR.IBSYH. This will set the signal IBUSY. A third access, read or write, to IBCR while IBCR.IBSYH remains set will cancel this third access and an error is signalled by setting EIR.IIBA. Besides canceling this last access to IBCR and setting the error bit, no further state change happens. So full operation is granted. IBCR remains read and write inaccessible until the transfer of data from the Input Shadow Buffer to the Message Buffer (IBF⇒MBF) is completed and once more the Input Buffers are swapped (shadow IBF changes to host IBF and host IBF to shadow IBF). During this time span all read and write accesses to the Input Buffer Control Register (IBCR) are canceled. The transfer is completed when IBCR.IBSYH is cleared. Additionally signal TIBC may be used, e.g. for service request triggering, DMA triggering, or driving a pin, to communicate the access status.

The read and write access to the Input Buffer Control Register (IBCR) may be configured for being automatically delayed by setting CUST1.IEN and configuring CUST3.TO to the maximum stall time acceptable to the system. By writing to IBCR.IBRH the Input Buffers are swapped (shadow IBF changes to host IBF and host IBF to shadow IBF), the content of the shadow IBF copied into the MBF, and IBCR.IBSYS is set. Writing to IBCR.IBRH a second time while IBCR.IBSYS remains set (previously initiated copy process ongoing) will correctly update IBCR.IBRH and set IBCR.IBSYH. A third access to IBCR while IBCR.IBSYH remains set will stall this read or write until either the maximum delay

FlexRay™ Protocol Controller (E-Ray)

time elapsed (in this case the read or write operation is cancelled after the stall time and an error is signalled by setting EIR.IOBA) or the read or write completes normally, after the transfer of data from the Input Shadow Buffer to the Message Buffer (IBF⇒MBF) is finalized and once more the Input Buffers are swapped (shadow IBF changes to host IBF and host IBF to shadow IBF). During this time the bus is locked and no further access to E-Ray module is possible due to the ongoing stalled read or write operation. Because no access is possible to the E-Ray module, read or write stall may only be detected through the signal TIBC or due to other not processed read or write accesses to the E-Ray module.

So setting CUST3.TO = FFFFFFFF_H, CUST1.IEN = 1, and CUST1.OEN = 1 will always grant a consistent data access of the host to the Output and Input Buffers without the need of reading and taking into account the status of OBCR.OBSYS or IBCR.IBSYH. But this simplified access may cause system latencies and system performance loss.

25.5.2.2 Special Registers

Test Register 1 (TEST1)

The Test Register 1 holds the control bits to configure the test modes of the E-Ray module. Write access to these bits is only possible if bit TEST1.WRTEN is set.

The Test Register 1 bits therefore can be used to test the interface to the physical layer (connectivity test) by driving / reading the respective pins.

When the E-Ray IP is operated in one of its test modes that requires TEST1.WRTEN to be set (RAM Test Mode, I/O Test Mode, Asynchronous Transmit Mode, and Loop Back Mode) only the selected test mode functionality is available.

The test functions are not available in addition to the normal operational mode functions, they change the functions of parts of the E-Ray module. Therefore normal operation as specified outside this chapter and as required by the FlexRay™ protocol specification and the FlexRay™ conformance test is not possible. Test mode functions may not be combined with each other or with FlexRay™ protocol functions.

The test mode features are intended for hardware testing or for FlexRay™ bus analyzer tools. They are not intended to be used in FlexRay™ applications

TEST1

Test Register 1

(0010_H)
Reset Value: 0000 0300_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CERB				CERA				0	TXE NB	TXE NA	TXB	TXA	RXB	RXA	
rh				rh				r	rwh	rwh	rwh	rwh	rh	rh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				AOB		AOA	0	TMC		0		ELB E	WRT EN		
r				rh		rh	r	rw		r		rw	rw		

Field	Bits	Type	Description
WRTEN	0	rw	Write Test Register Enable Enables write access to the test registers. To set the bit from 0 to 1 the test mode key has to be written as defined on “Lock Register (LCK)” on Page 25-30 . The unlock sequence is not required when TEST1.WRTEN is kept at 1 while other bits of the register are changed. The bit can be reset to 0 at any time. 0 _B Write access to test registers disabled. 1 _B Write access to test registers enabled.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
ELBE	1	rw	External Loop Back Enable There are two possibilities to perform a loop back test. External loop back via physical layer or internal loop back for in-system self-test (default). In case of an internal loop back pins $\overline{\text{TXENA}}$ and $\overline{\text{TXENB}}$ are in their inactive state, pins TXDA and TXDB are set to HIGH, pins RXDA and RXDB are not evaluated. Bit ELBE is evaluated only when POC is in loop back mode and test multiplexer control is in non multiplexed mode TMC = 00. 0_B Internal loop back (default) 1_B External loop back
TMC	[5:4]	rw	Test Multiplexer Control 00_B Normal signal path (default). 01_B RAM Test Mode: Internal busses are multiplexed to make all RAM blocks of the E-Ray module directly accessible by the Host. This mode is intended to enable testing of the embedded RAM blocks during production testing. 10_B I/O Test Mode: Output pins are driven to the values defined by bits TXA, TXB, $\overline{\text{TXENA}}$, $\overline{\text{TXENB}}$. The values applied to the input pins can be read from register bits RXA and RXB. 11_B Reserved; should not be used.
AOA	8	rh	Activity on A The channel idle condition is specified in the FlexRay™ protocol spec v2.1, chapter 3, BITSTRB process (zChannellIdle). 0_B No activity detected, channel A idle 1_B Activity detected, channel A not idle
AOB	9	rh	Activity on B The channel idle condition is specified in the FlexRay™ protocol spec v2.1, chapter 3, BITSTRB process (zChannellIdle). 0_B No activity detected, channel B idle 1_B Activity detected, channel B not idle
RXA	16	rh	Read Channel A Receive Pin 0_B RXDA = 0 1_B RXDA = 1

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
RXB	17	rh	Read Channel B Receive Pin 0_B RXDB = 0 1_B RXDB = 1
TXA	18	rwh	Read or Write to Channel A Transmit Pin 0_B TXDA = 0 1_B TXDA = 1
TXB	19	rwh	Read or Write to Channel B Transmit Pin 0_B TXDB = 0 1_B TXDB = 1
TXENA	20	rwh	Read or Write to Channel A Transmit Enable Pin 0_B TXENA = 0 1_B TXENA = 1
TXENB	21	rwh	Read or Write to Channel B Transmit Enable Pin 0_B TXENB = 0 1_B TXENB = 1
CERA	[27:24]	rh	Coding Error Report Channel A¹⁾ Set when a coding error is detected on channel A. Reset to zero when register TEST1 is read or written. Once the CERA is set it will remain unchanged until the Host accesses the TEST1 register. 0000_B No coding error detected 0001_B Header CRC error detected 0010_B Frame CRC error detected 0011_B Frame Start Sequence FSS too long 0100_B First bit of Byte Start Sequence BSS seen LOW 0101_B Second bit of Byte Start Sequence BSS seen HIGH 0110_B First bit of Frame End Sequence FES seen HIGH 0111_B Second bit of Frame End Sequence FES seen LOW 1000_B CAS / MTS symbol seen too short 1001_B CAS / MTS symbol seen too long Other combinations are reserved.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
CERB	[31:28]	rh	Coding Error Report Channel B¹⁾ Set when a coding error is detected on channel B. Reset to zero when register TEST1 is read or written. Once the CERB is set it will remain unchanged until the Host accesses the TEST1 register. 0000 _B No coding error detected 0001 _B Header CRC error detected 0010 _B Frame CRC error detected 0011 _B Frame Start Sequence FSS too long 0100 _B First bit of Byte Start Sequence BSS seen LOW 0101 _B Second bit of Byte Start Sequence BSS seen HIGH 0110 _B First bit of Frame End Sequence FES seen HIGH 0111 _B Second bit of Frame End Sequence FES seen LOW 1000 _B CAS / MTS symbol seen too short 1001 _B CAS / MTS symbol seen too long Other combinations are reserved.
0	[3:2], [7:6], [15:10], [23:22]	r	Reserved Returns 0 if read; should be written with 0.

1) Coding errors are also signalled when the Communication Controller is in "MONITOR_MODE". The error codes regarding CAS / MTS symbols concern only the monitored bit pattern, irrelevant whether those bit patterns are seen in the symbol window or elsewhere.

Asynchronous Transmit Mode (ATM)

The asynchronous transmit mode is entered by writing 1110_B to the CHI Command Vector SUCC1.CMD in the SUC Configuration Register 1 (CHI command: ATM) while the Communication Controller is in "CONFIG" state and bit TEST1.WRTEN in the Test Register 1 is set. This write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). When called in any other state or when bit TEST1.WRTEN is not set, SUCC1.CMD will be reset to 0000_B = "COMMAND_NOT_ACCEPTED". CCSV.POCS in the Communication Controller Status Vector will return 1110_B while the E-Ray module is in ATM mode. Asynchronous Transmit mode can be left by writing 0001_B (CHI command: "CONFIG") to the CHI Command Vector SUCC1.CMD in the SUC Configuration Register 1.

In ATM mode transmission of a FlexRay™ Frame is triggered by writing the number of the respective Message Buffer to the Input Buffer Command Request register (IBCR.IBRH) while bit IBCM.STXRS in the Input Buffer Command Mask register is set to 1. In this mode wake-up, startup, and clock synchronization are bypassed. The CHI command SEND_MTS results in the immediate transmission of an MTS symbol.

FlexRay™ Protocol Controller (E-Ray)

The cycle counter value of Frames send in ATM mode can be programmed via MTCCV.CCV (writable in ATM and loop back mode only).

Loop Back Mode

The loop back mode is entered by writing 1111_B to the CHI Command Vector SUCC1.CMD in the SUC Configuration Register 1 (CHI command: LOOP_BACK) while the Communication Controller is in “CONFIG” state and bit TEST1.WRTEN in the Test Register 1 is set. This write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). When called in any other state or when bit TEST1.WRTEN is not set, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”. CCSV.POCS in the Communication Controller Status Vector will show $0000\ 1101_H$ while the E-Ray module is in loop back mode.

Loop Back mode can be left by writing 0001_B (CHI command: “CONFIG”) to the CHI Command Vector SUCC1.CMD in the SUC Configuration Register 1.

The loop back test mode is intended to check the module’s internal data paths. Normal, time triggered operation is not possible in loop back mode.

There are two possibilities to perform a loop back test. External loop back via physical layer (TEST1.ELBE = 1) or internal loop back for in-system self-test (TEST1.ELBE = 0). In case of an internal loop back pins TXENA, TXENB are in their inactive state, pins TXDA and TXDB are set to HIGH, pins RXDAn and RXDBn are not evaluated.

When the Communication Controller is in loop back mode, a loop back test is started by the Host writing a message to the Input Buffer and requesting the transmission by writing to the Input Buffer Command Request register IBCR. The Message Handler will transfer the message into the Message RAM and then into the Transient Buffer of the selected channel. The Channel Protocol Controller (PRT) will read (in 32-bit words) the message from the transmit part of the Transient Buffer and load it into its Rx / Tx shift register. The serial transmission is looped back into the shift register; its content is written into the receive part of the channels’s Transient Buffer before the next word is loaded.

The PRT and the Message Handler will then treat this transmitted message like a received message, perform an acceptance filtering on Frame ID and receive channel, and store the message into the Message RAM if it passed acceptance filtering. The loop back test ends with the Host requesting this received message from the Message RAM and then checking the contents of the Output Buffer.

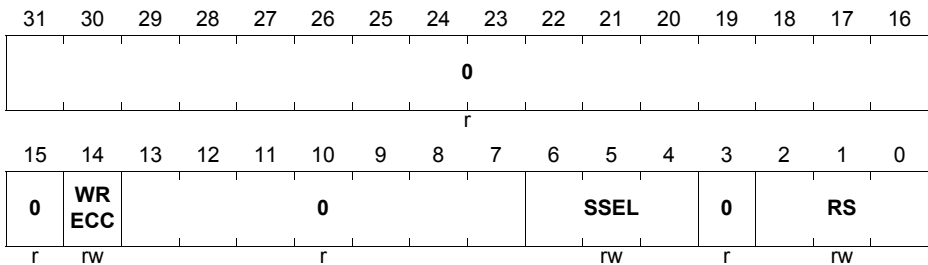
Each FlexRay™ channel is tested separately. The E-Ray cannot receive messages from the FlexRay™ bus while it is in the loop back mode.

The cycle counter value of Frames used in loop back mode can be programmed via MTCCV.CCV (writable in ATM and loop back mode only).

Note that in case of an odd payload the last two bytes of the looped-back payload will be shifted by 16 bits to the right inside the last 32-bit data word.

FlexRay™ Protocol Controller (E-Ray)
Test Register 2 (TEST2)

The Test Register 2 holds all bits required for the RAM test of the seven embedded RAM blocks of the E-Ray module. Write access to this register is only possible when TEST1.WRTEN in the Test Register 1 is set to 1.

TEST2
Test Register 2
(0014_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RS	[2:0]	rw	RAM Select In RAM Test mode the RAM blocks selected by RS are mapped to module address 0000 0400 _H to 0000 07FF _H (1024 byte addresses). 000 _B Input Buffer RAM 1 (IBF1) 001 _B Input Buffer RAM 2 (IBF2) 010 _B Output Buffer RAM 1 (OBF1) 011 _B Output Buffer RAM 2 (OBF2) 100 _B Transient Buffer RAM A (TBF1) 101 _B Transient Buffer RAM B (TBF2) 110 _B Message RAM (MBF) 111 _B Reserved; should not be used.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
SSEL	[6:4]	rw	Segment Select To enable access to the complete Message RAM (8192 byte addresses) the Message RAM is segmented. 000 _B access to RAM byte 0000 _H to 03FF _H enabled 001 _B access to RAM byte 0400 _H to 07FF _H enabled 010 _B access to RAM byte 0800 _H to 0BFF _H enabled 011 _B access to RAM byte 0C00 _H to 0FFF _H enabled 100 _B access to RAM byte 1000 _H to 11FF _H enabled 101 _B access to RAM byte 1400 _H to 17FF _H enabled 110 _B access to RAM byte 1800 _H to 1BFF _H enabled 111 _B access to RAM byte 1C00 _H to 1FFF _H enabled
WR ECC	14	rw	Write ECC Data Enable Content of ECCW is transferred to the RAM: 0 _B disabled 1 _B enabled <i>Note: Test mode must be entered. See “Test Register 1 (TEST1)” on Page 25-22</i>
0	3, [13:7], 15, [31:16]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

RAM Test Mode

In RAM test mode (TEST1.TMC = 1), one of the seven RAM blocks can be selected for direct RD/WR access by programming TEST2.RS.

For external access the selected RAM block is mapped to address space 400_H to 7FF_H (1024 byte addresses or 256 word addresses).

Because the length of the Message RAM exceeds the available address space, the Message RAM is segmented into segments of 1024 byte. The segments can be selected by programming TEST2.SSEL in the Test Register 2.

In RAM test mode the memory must be accessed with 32Bit accesses only.

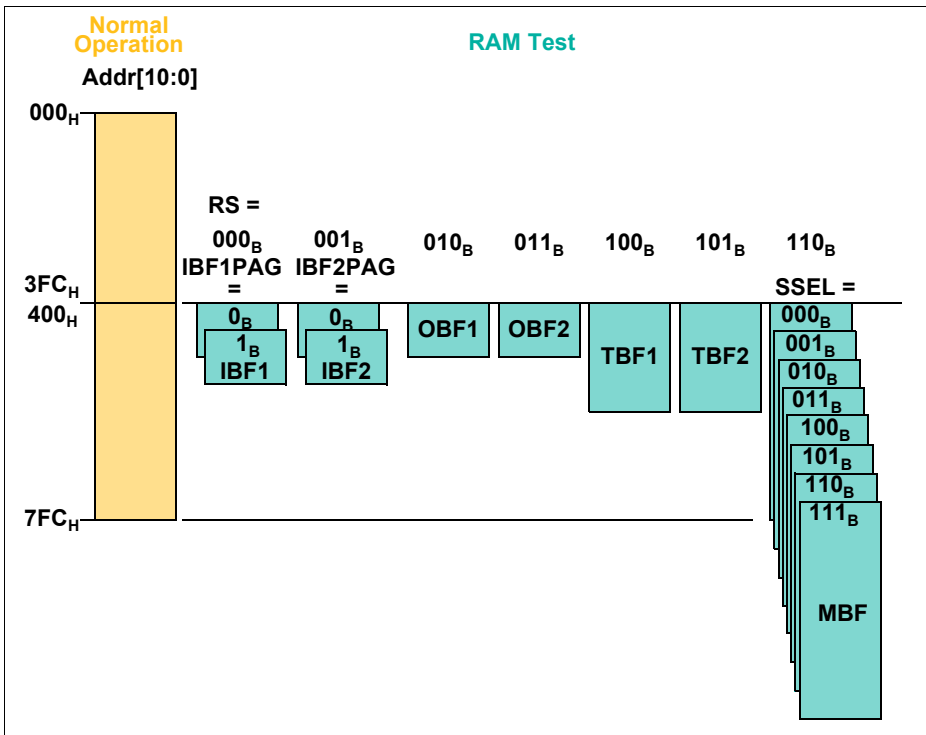


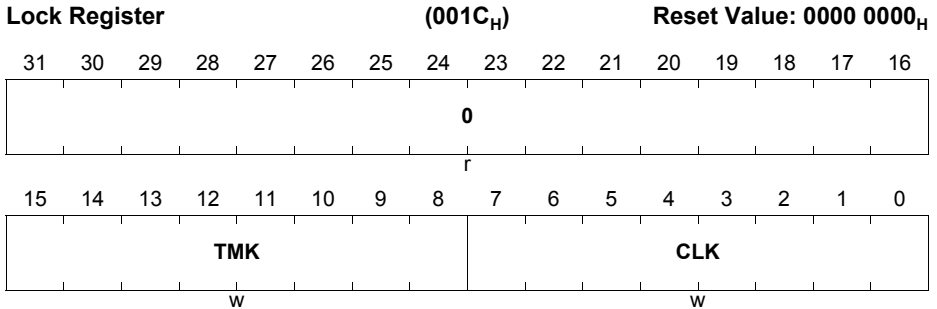
Figure 25-3 RAM test mode Access to E-Ray RAM Blocks

FlexRay™ Protocol Controller (E-Ray)

Lock Register (LCK)

The Lock Register is write-only. Reading the register will return 0000 0000_H.

LCK



Field	Bits	Type	Description
CLK	[7:0]	w	<p>Configuration Lock Key</p> <p>To leave “CONFIG” state by writing to SUCC1.CMD commands READY, MONITOR_MODE, ATM, LOOP_BACK) in the SUC Configuration Register 1, the write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). If the write sequence below is interrupted by other write accesses between the second write to the Configuration Lock Key and the write access to the SUCC1 register, the Communication Controller remains in “CONFIG” state and the sequence has to be repeated.</p> <p>First write: LCK.CLK = CE_H = 1100 1110_B</p> <p>Second write: LCK.CLK = 31_H = 0011 0001_B</p> <p>Third write: SUCC1.CMD</p> <p>Returns 0 if read</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TMK	[15:8]	w	Test Mode Key To set bit TEST1.WRTEN the write operation has to be directly preceded by two consecutive write accesses to the Test Mode Key. If the write sequence is interrupted by other write accesses between the second write to the Test Mode Key and the write access to the Test1 register, bit TEST1.WRTEN is not set to 1 and the sequence has to be repeated. First write: LCK.TMK = 75 _H = 0111 0101 _B Second write: LCK.TMK = 8A _H = 1000 1010 _B Second write: TEST1.WRTEN = 1 Returns 0 if read
0	[31:16]	r	Reserved Returns 0 if read; should be written with 0.

Note: In case the Host uses 8/16-bit accesses to write the listed bit fields, the programmer has to ensure that no “dummy accesses” e.g. to the remaining register bytes / words are inserted by the compiler.

To exit “CONFIG” state by writing to SUCC1.CMD in the SUC Configuration Register 1, the write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key. If this write sequence is service requested by read accesses or write accesses to other locations, the Communication Controller remains in “CONFIG” state and the sequence has to be repeated.

First write: LCK.CLK = CE_H = 1100 1110_B

Second write: LCK.CLK = 31_H = 0011 0001_B

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
SFBM	2	rwh	<p>SYNC Frames Below Minimum</p> <p>This flag signals that the number of SYNC Frames received during the last communication cycle was below the limit required by the FlexRay™ protocol. May be set during startup and therefore should be cleared by the Host after the Communication Controller entered “NORMAL_ACTIVE” state.</p> <p>0_B Sync node: 1 or more SYNC Frames received Non-sync node: 2 or more SYNC Frames received</p> <p>1_B Less than the required minimum of SYNC Frames received</p> <p>This flag is cleared by writing a 1.</p>
SFO	3	rwh	<p>SYNC Frame Overflow</p> <p>Set when either the number of SYNC Frames received during the last communication cycle or the total number of SYNC Frames received during the last double cycle exceeds the maximum number of SYNC Frames as defined by GTUC02.SNM in the GTU Configuration Register 2.</p> <p>0_B Number of received SYNC Frames ≤ GTUC02.SNM</p> <p>1_B More SYNC Frames received than configured by GTUC02.SNM</p> <p>This flag is cleared by writing a 1.</p>
CCF	4	rwh	<p>Clock Correction Failure</p> <p>This flag is set at the end of the cycle whenever one of the following errors occurred:</p> <ul style="list-style-type: none"> • Missing offset and / or rate correction • Clock Correction limit reached <p>The clock correction status is monitored in registers CCEV and SFS. A failure may occur during startup, therefore bit CCF should be cleared by the Host after the Communication Controller entered “NORMAL_ACTIVE” state.</p> <p>0_B Clock correction successful so far</p> <p>1_B Clock correction failed</p> <p>This flag is cleared by writing a 1.</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
CCL	5	rwh	<p>CHI Command Locked</p> <p>The flag signals that the write access to the CHI command vector SUCC1.CMD was not successful because the execution of the previous CHI command has not yet completed. In this case bit EIR.CNA is also set to 1.</p> <p>0_B CHI command accepted 1_B CHI command not accepted This flag is cleared by writing a 1.</p>
EERR	6	rh	<p>ECC Error</p> <p>The flag signals an ECC error to the Host. It is set whenever one of the flags MHDS.EIBF, MHDS.EOBF, MHDS.EMR, MHDS.ETBF1, MHDS.ETBF2 changes from 0 to 1.</p> <p>See also “Message Handler Status (MHDS)” on Page 25-136. This bit must be cleared at initialization of the module!</p> <p>0_B No error detected 1_B Error detected</p>
RFO	7	rh	<p>Receive FIFO Overrun</p> <p>The flag is set by the Communication Controller when a receive FIFO overrun is detected. When a receive FIFO overrun occurs, the oldest message is overwritten with the actual received message. The actual state of the FIFO is monitored in register FSR.</p> <p>0_B No receive FIFO overrun detected 1_B A receive FIFO overrun has been detected</p>
EFA	8	rwh	<p>Empty FIFO Access</p> <p>This flag is set by the Communication Controller when the Host requests the transfer of a message from the receive FIFO via Output Buffer while the receive FIFO is empty.</p> <p>0_B No Host access to empty FIFO occurred 1_B Host access to empty FIFO occurred</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
IIBA	9	rwh	<p>Illegal Input Buffer Access</p> <p>This flag is set by the Communication Controller when the Host wants to modify a Message Buffer via Input Buffer while the Communication Controller is not in “CONFIG” or “DEFAULT_CONFIG” state and one of the following conditions applies:</p> <ol style="list-style-type: none"> The Host writes to the Input Buffer Command Request register to modify the: <ol style="list-style-type: none"> Header Section of Message Buffer 0, 1 if configured for transmission in key slot Header Section of static Message Buffers with buffer number < MRC.FDB while MRC.SEC = 0_B Header Section of any static or dynamic Message Buffer while MRC.SEC = 1_B Header and / or Data Section of any message buffer belonging to the receive FIFO The Host writes to any register of the Input Buffer while IBCR.IBSYS is set. <p>0_B No illegal Host access to Input Buffer occurred 1_B Illegal Host access to Input Buffer occurred</p>
IOBA	10	rwh	<p>Illegal Output Buffer Access</p> <p>This flag is set by the Communication Controller when the Host requests the transfer of a Message Buffer from the Message RAM to the Output Buffer while OBCR.OBSYS is set to 1.</p> <p>0_B No illegal Host access to Output Buffer occurred 1_B Illegal Host access to Output Buffer occurred</p>
MHF	11	rwh	<p>Message Handler Constraints Flag</p> <p>The flag signals a Message Handler constraints violation condition. It is set whenever one of the flags MHDF.SNUA, MHDF.SNUB, MHDF.FNFA, MHDF.FNFB, MHDF.TBFA, MHDF.TBFB, MHDF.TNSA, MHDF.TNSB, MHDF.WAHP changes from 0 to 1.</p> <p>0_B No Message Handler failure detected 1_B Message Handler failure detected</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
EDA	16	rwh	Error Detected on Channel A This bit is set whenever one of the flags ACS.SEDA, ACS.CEDA, ACS.CIA, ACS.SBVA changes from 0 to 1. 0_B No error detected on channel A 1_B Error detected on channel A This flag is cleared by writing a 1.
LTVA	17	rwh	Latest Transmit Violation Channel A The flag signals a latest transmit violation on channel A to the Host. 0_B No latest transmit violation detected on channel A 1_B Latest transmit violation detected on channel A This flag is cleared by writing a 1.
TABA	18	rwh	Transmission Across Boundary Channel A The flag signals to the Host that a transmission across a slot boundary occurred for channel A. 0_B No transmission across slot boundary detected on channel A 1_B Transmission across slot boundary detected on channel A This flag is cleared by writing a 1.
EDB	24	rwh	Error Detected on Channel B This bit is set whenever one of the flags ACS.SEDB, ACS.CEDB, ACS.CIB, ACS.SBVB changes from 0 to 1. 0_B No error detected on channel B 1_B Error detected on channel B This flag is cleared by writing a 1.
LTVB	25	rwh	Latest Transmit Violation Channel B The flag signals a latest transmit violation on channel B to the Host. 0_B No latest transmit violation detected on channel B 1_B Latest transmit violation detected on channel B This flag is cleared by writing a 1.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TABB	26	rwh	<p>Transmission Across Boundary Channel B</p> <p>The flag signals to the Host that a transmission across a slot boundary occurred for channel B.</p> <p>0_B No transmission across slot boundary detected on channel B</p> <p>1_B Transmission across slot boundary detected on channel B</p> <p>This flag is cleared by writing a 1.</p>
0	[15:12], [23:19], [31:27]	r	<p>Reserved</p> <p>Returns 0 if read; should be written with 0.</p>

FlexRay™ Protocol Controller (E-Ray)
Status Service Request Register (SIR)

The flags are set whenever the Communication Controller detects one of the listed events. The flags remain set until the Host clears them. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. An application reset will also clear the register.

SIR
Status Service Request Register (0024_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						MTS B	WUP B	0						MTS A	WUP A
r						rwh	rwh	r						rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDS	MBS I	SUC S	SWE	TOB C	TIBC	TI1	TIO	NMV C	RF CL	RF NE	RXI	TXI	CYC S	CAS	WST
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rh	rh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WST	0	rwh	Wakeup Status This flag is set when the wakeup status vector CCSV.WSV in the Communication Controller Status Vector register changes to a value other than UNDEFINED. 0 _B Wake-up status unmodified 1 _B Wake-up status modified (and not UNDEFINED) This flag is cleared by writing a 1.
CAS	1	rwh	Collision Avoidance Symbol This flag is set by the Communication Controller during STARTUP state when a CAS or potential CAS was received. 0 _B No bit pattern matching the CAS symbol received 1 _B Bit pattern matching the CAS symbol received This flag is cleared by writing a 1.
CYCS	2	rwh	Cycle Start Service Request This flag is set by the Communication Controller when a communication cycle starts 0 _B No communication cycle started 1 _B Communication cycle started This flag is cleared by writing a 1.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TXI	3	rwh	<p>Transmit Service Request</p> <p>This flag is set by the Communication Controller at the end of Frame transmission if bit WRHS1.MBI in the respective Message Buffer is set (see Table 25-24).</p> <p>0_B No Frame transmitted from a transmit buffer with WRHS1.MBI = 1</p> <p>1_B At least one Frame was transmitted from a transmit buffer with WRHS1.MBI = 1</p> <p>This flag is cleared by writing a 1.</p>
RXI	4	rwh	<p>Receive Service Request</p> <p>This flag is set by the Communication Controller whenever the set condition of a Message Buffer ND flag is fulfilled and if bit WRHS1.MBI of that Message Buffer is set to 1(see Table 25-24).</p> <p>0_B No ND flag of a receive buffer with WRHS1.MBI = 1 has been set to 1</p> <p>1_B At least one ND flag of a receive buffer with WRHS1.MBI = 1 has been set to 1</p> <p>This flag is cleared by writing a 1.</p>
RFNE	5	rh	<p>Receive FIFO Not Empty</p> <p>This flag is set by the Communication Controller when a received valid Frame was stored into the empty receive FIFO.m The actual state of the receive FIFO is monitored in register FSR</p> <p>0_B Receive FIFO is empty</p> <p>1_B Receive FIFO is not empty</p>
RFCL	6	rh	<p>Receive FIFO Critical Level</p> <p>This flag is set when a valid receive FIFO fill level FSR.RFFL is equal or greater than the critical level as configured by FCL.CL.</p> <p>0_B Receive FIFO below critical level</p> <p>1_B Receive FIFO critical level reached</p>
NMVC	7	rwh	<p>Network Management Vector Changed</p> <p>This service request flag signals a change in the Network Management Vector visible to the Host.</p> <p>0_B No change in the Network Management vector</p> <p>1_B Network Management vector changed</p> <p>This flag is cleared by writing a 1.</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TIO	8	rwh	<p>Timer Service Request 0</p> <p>This flag is set whenever timer 0 matches the conditions configured in the Timer Service Request 0 Configuration Register T0C. A Timer Service Request 0 is also signalled by TINT0SRC .</p> <p>0_B No Timer Service Request 0 1_B Timer Service Request 0 occurred This flag is cleared by writing a 1.</p>
TI1	9	rwh	<p>Timer Service Request 1</p> <p>This flag is set whenever the conditions programmed in the Timer Service Request 1 Configuration Register T1C are met. A Timer Service Request 1 is also signalled by TINT1SRC.</p> <p>0_B No Timer Service Request 1 1_B Timer Service Request 1 occurred This flag is cleared by writing a 1.</p>
TIBC	10	rwh	<p>Transfer Input Buffer Completed</p> <p>This flag is set whenever a transfer from Input Buffer to the Message RAM has completed and bit IBCR.IBSYS in the Input Buffer Command Request register has been reset by the Message Handler.</p> <p>0_B No transfer completed 1_B Transfer between Input Buffer and Message RAM completed This flag is cleared by writing a 1.</p>
TOBC	11	rwh	<p>Transfer Output Buffer Completed</p> <p>This flag is set whenever a transfer from Message RAM to the Output Buffer has completed and bit OBCR.OBSYS in the Output Buffer Command Request register has been reset by the Message Handler.</p> <p>0_B No transfer completed 1_B Transfer between Message RAM and the Output Buffer completed This flag is cleared by writing a 1.</p>
SWE	12	rwh	<p>Stop Watch Event</p> <p>This flag is set after a stop watch activation when the current cycle counter and Macrotick value are stored in the Stop Watch Register 1 (STPW1).</p> <p>0_B No Stop Watch Event 1_B Stop Watch Event occurred This flag is cleared by writing a 1.</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
SUCS	13	rwh	<p>Startup Completed Successfully</p> <p>This flag is set whenever a startup completed successfully and the Communication Controller entered “NORMAL_ACTIVE” state.</p> <p>0_B No startup completed successfully 1_B Startup completed successfully</p> <p>This flag is cleared by writing a 1.</p>
MBSI	14	rwh	<p>Message Buffer Status Service Request</p> <p>This flag is set by the Communication Controller when the Message Buffer status MBS has changed and if bit RDHS1.MBI of that Message Buffer is set (see Table 25-24).</p> <p>0_B No Message Buffer status change of Message Buffer with RDHS1.MBI= 1 has changed 1_B Message Buffer status of at least one Message Buffer with RDHS1.MBI= 1 has changed</p> <p>This flag is cleared by writing a 1.</p>
SDS	15	rwh	<p>Start of Dynamic Segment</p> <p>This flag is set by the Communication Controller when the dynamic segment starts.</p> <p>0_B Dynamic segment not yet started 1_B Dynamic segment started</p>
WUPA	16	rwh	<p>Wakeup Pattern Channel A</p> <p>This flag is set by the Communication Controller when a wakeup pattern was received on channel A. Only set when the Communication Controller is in “WAKEUP”, “READY”, or “STARTUP” state, or when in Monitor mode.</p> <p>0_B No wake-up pattern received on channel A 1_B Wake-up pattern received on channel A</p> <p>This flag is cleared by writing a 1.</p>
MTSA	17	rwh	<p>MTS Received on Channel A (vSS!ValidMTSA)</p> <p>Media Access Test symbol received on channel A during the proceeding symbol window. Updated by the Communication Controller for each channel at the end of the symbol window.</p> <p>0_B No MTS symbol received on channel A 1_B MTS symbol received on channel A</p> <p>This flag is cleared by writing a 1.</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
WUPB	24	rwh	Wakeup Pattern Channel B This flag is set by the Communication Controller when a wakeup pattern was received on channel B. Only set when the Communication Controller is in “WAKEUP”, “READY”, or “STARTUP” state, or when in Monitor mode. 0 _B No wake-up pattern received on channel B 1 _B Wake-up pattern received on channel B This flag is cleared by writing a 1.
MTSB	25	rwh	MTS Received on Channel B (vSSI!ValidMTSB) Media Access Test symbol received on channel B during the preceding symbol window. Updated by the Communication Controller for each channel at the end of the symbol window. 0 _B No MTS symbol received on channel B 1 _B MTS symbol received on channel B This flag is cleared by writing a 1.
0	[23:18], [31:26]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
SFOL	3	rw	SYNC Frame Overflow Service Request Line
			0 _B Service Request assigned to service request line INT0SRC
CCFL	4	rw	Clock Correction Failure Service Request Line
			1 _B Service Request assigned to service request line INT1SRC
CCLL	5	rw	CHI Command Locked Service Request Line
			1 _B Service Request assigned to service request line INT1SRC
EERRL	6	rw	ECC Error Service Request Line
			1 _B Service Request assigned to service request line INT1SRC
RFOL	7	rw	Receive FIFO Overrun Service Request Line
			1 _B Service Request assigned to service request line INT1SRC
EFAL	8	rw	Empty FIFO Access Service Request Line
			1 _B Service Request assigned to service request line INT1SRC
IIBAL	9	rw	Illegal Input Buffer Access Service Request Line
			1 _B Service Request assigned to service request line INT1SRC

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
IOBAL	10	rw	Illegal Output Buffer Access Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
MHFL	11	rw	Message Handler Constrains Flag Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
EDAL	16	rw	Error Detected on Channel A Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
LTVAL	17	rw	Latest Transmit Violation Channel A Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
TABAL	18	rw	Transmission Across Boundary Channel A Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
EDBL	24	rw	Error Detected on Channel B Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
LTVBL	25	rw	Latest Transmit Violation Channel B Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TABBL	26	rw	Transmission Across Boundary Channel A Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
0	[15:12], [23:19], [31:27]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)
Status Service Request Line Select (SILS)

The Status Service Request Line Select register assign an service request generated by a specific status service request flag from register SIR to one of the two module service request lines INT0SRC or INT1SRC:

0 = Interrupt assigned to interrupt line INT0SRC

1 = Interrupt assigned to interrupt line INT1SRC

SILS
Status Service Request Line Select (002C_H)
Reset Value: 0303 FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						MTS BL	WUP BL	0						MTS AL	WUP AL
r						rw	rw	r						rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDS L	MBS IL	SUC SL	SWE L	TOB CL	TIBC L	TI1L	TI0L	NMV CL	RFC LL	RFN EL	RXIL	TXIL	CYC SL	CAS L	WST L
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
WSTL	0	rw	Wakeup Status Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
CASL	1	rw	Collision Avoidance Symbol Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
CYCSL	2	rw	Cycle Start Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TXIL	3	rw	Transmit Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
RXIL	4	rw	Receive Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
RFNEL	5	rw	Receive FIFO Not Empty Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
RFCLL	6	rw	Receive FIFO Critical Level Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
NMVCL	7	rw	Network Management Vector Changed Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
TI0L	8	rw	Timer Service Request 0 Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
TI1L	9	rw	Timer Service Request 1 Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TIBCL	10	rw	Transfer Input Buffer Completed Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
TOBCL	11	rw	Transfer Output Buffer Completed Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
SWEL	12	rw	Stop Watch Event Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
SUCSL	13	rw	Startup Completed Successfully Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
MBSIL	14	rw	Message Buffer Status Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
SDSL	15	rw	Start of Dynamic Segment Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
WUPAL	16	rw	Wakeup Pattern Channel A Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
MTSAL	17	rw	Media Access Test Symbol Channel A Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
WUPBL	24	rw	Wakeup Pattern Channel B Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
MTSBL	25	rw	Media Access Test Symbol Channel B Service Request Line 0 _B Service Request assigned to service request line INT0SRC 1 _B Service Request assigned to service request line INT1SRC
0	[23:18], [31:26]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
SFOE	3	rwh	SYNC Frame Overflow Service Request Enable 0 _B Read: SYNC Frame Overflow Service Request disabled Write: Unchanged 1 _B Read: SYNC Frame Overflow Service Request enabled Write: Enable Protocol Error Mode Changed Service Request
CCFE	4	rwh	Clock Correction Failure Service Request Enable 0 _B Read: Clock Correction Failure Service Request disabled Write: Unchanged 1 _B Read: Clock Correction Failure Service Request enabled Write: Enable Clock Correction Failure Service Request
CCLE	5	rwh	CHI Command Locked Service Request Enable 0 _B Read: CHI Command Locked Service Request disabled Write: Unchanged 1 _B Read: CHI Command Locked Service Request enabled Write: Enable CHI Command Locked Service Request
EERRE	6	rwh	ECC Error Service Request Enable 0 _B Read: ECC Error Service Request disabled Write: Unchanged 1 _B Read: ECC Error Service Request enabled Enable ECC Error Service Request Write: Unchanged
RFOE	7	rwh	Receive FIFO Overrun Service Request Enable 0 _B Read: Receive FIFO Overrun Service Request disabled Write: Unchanged 1 _B Read: Receive FIFO Overrun Service Request enabled Write: Enable Receive FIFO Overrun Service Request
EFAE	8	rwh	Empty FIFO Access Service Request Enable 0 _B Read: Empty FIFO Access Service Request disabled Write: Unchanged 1 _B Read: Empty FIFO Access Service Request enabled Write: Enable Empty FIFO Access Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
IIBAE	9	rwh	Illegal Input Buffer Access Service Request Enable 0 _B Read: Illegal Input Buffer Access Service Request disabled Write: Unchanged 1 _B Read: Illegal Input Buffer Access Service Request enabled Write: Enable Illegal Input Buffer Access Service Request
IOBAE	10	rwh	Illegal Output Buffer Access Service Request Enable 0 _B Read: Illegal Output Buffer Access Service Request disabled Write: Unchanged 1 _B Read: Illegal Output Buffer Access Service Request enabled Write: Enable Illegal Output Buffer Access Service Request
MHFE	11	rwh	Message Handler Constraints Flag Service Request Enable 0 _B Read: Message Handler Constraints Flag Service Request disabled Write: Unchanged 1 _B Read: Message Handler Constraints Flag Service Request enabled Write: Enable Message Handler Constraints Flag Service Request
EDAE	16	rwh	Error Detected on Channel A Service Request Enable 0 _B Read: Error Detected on Channel A Service Request disabled Write: Unchanged 1 _B Read: Error Detected on Channel A Service Request enabled Write: Enable Error Detected on Channel A Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
LTVAE	17	rwh	Latest Transmit Violation Channel A Service Request Enable 0 _B Read: Latest Transmit Violation Channel A Service Request disabled Write: Unchanged 1 _B Read: Latest Transmit Violation Channel A Service Request enabled Write: Enable Latest Transmit Violation Channel A Service Request
TABAE	18	rwh	Transmission Across Boundary Channel A Service Request Enable 0 _B Read: Transmission Across Boundary Channel A Service Request disabled Write: Unchanged 1 _B Read: Transmission Across Boundary Channel A Service Request enabled Write: Enable Transmission Across Boundary Channel A Service Request
EDBE	24	rwh	Error Detected on Channel B Service Request Enable 0 _B Read: Error Detected on Channel B Service Request disabled Write: Unchanged 1 _B Read: Error Detected on Channel B Service Request enabled Write: Enable Error Detected on Channel B Service Request
LTVBE	25	rwh	Latest Transmit Violation Channel B Service Request Enable 0 _B Read: Latest Transmit Violation Channel B Service Request disabled Write: Unchanged 1 _B Read: Latest Transmit Violation Channel B Service Request enabled Write: Enable Latest Transmit Violation Channel B Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TABBE	26	rwh	<p>Transmission Across Boundary Channel B Service Request Enable</p> <p>0_B Read: Transmission Across Boundary Channel B Service Request disabled Write: Enable Transmission Across Boundary Channel B Service Request</p> <p>1_B Read: Transmission Across Boundary Channel B Service Request enabled Write: Enable Transmission Across Boundary Channel B Service Request</p>
0	[15:12], [23:19], [31:27]	r	<p>Reserved Returns 0 if read; should be written with 0.</p>

FlexRay™ Protocol Controller (E-Ray)
Error Service Request Enable Reset (EIER)

The settings in the Error Service Request Enable register determine which status changes in the Error Service Request Register will result in a service request. The enable bits are set by writing to EIES and reset by writing to EIER. Writing a 1 resets the specific enable bit, a 0 has no effect.

EIER
Error Service Request Enable Reset (0034_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
		0			TAB BE	LTV BE	EDB E			0			TAB AE	LTV AE	EDA E	
		r			rwh	rwh	rwh			r			rwh	rwh	rwh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
		0			MHF E	IOB AE	IIBA E	EFA E	RFO E	EER RE	CCL E	CCF E	SFO E	SFB ME	CNA E	PEM CE
		r			rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PEMCE	0	rwh	POC Error Mode Changed Service Request Enable
			0 _B Read: Protocol Error Mode Changed Service Request disabled Write: Unchanged 1 _B Read: Protocol Error Mode Changed Service Request enabled Write: Disable Protocol Error Mode Changed Service Request
CNAE	1	rwh	Command Not Accepted Service Request Enable
			0 _B Read: Command Not Accepted Service Request disabled Write: Unchanged 1 _B Read: Command Not Accepted Service Request enabled Write: Disable Command Not Accepted Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
SFBME	2	rwh	<p>SYNC Frames Below Minimum Service Request Enable</p> <p>0_B Read: SYNC Frames Below Minimum Service Request disabled Write: Unchanged</p> <p>1_B Read: SYNC Frames Below Minimum Service Request enabled Write: Disable SYNC Frames Below Minimum Service Request</p>
SFOE	3	rwh	<p>SYNC Frame Overflow Service Request Enable</p> <p>0_B Read: SYNC Frame Overflow Service Request disabled Write: Unchanged</p> <p>1_B Read: SYNC Frame Overflow Service Request enabled Write: Disable Protocol Error Mode Changed Service Request</p>
CCFE	4	rwh	<p>Clock Correction Failure Service Request Enable</p> <p>0_B Read: Clock Correction Failure Service Request disabled Write: Unchanged</p> <p>1_B Read: Clock Correction Failure Service Request enabled Write: Disable Clock Correction Failure Service Request</p>
CCLE	5	rwh	<p>CHI Command Locked Service Request Enable</p> <p>0_B Read: CHI Command Locked Service Request disabled Write: Unchanged</p> <p>1_B Read: CHI Command Locked Service Request enabled Write: Disable CHI Command Locked Service Request</p>
EERRE	6	rwh	<p>ECC Error Service Request Enable</p> <p>0_B Read: ECC Error Service Request disabled Write: Unchanged</p> <p>1_B ERead: CC Error Service Request enabled Write: Disable ECC Error Service Request</p>
RFOE	7	rwh	<p>Receive FIFO Overrun Service Request Enable</p> <p>0_B Read: Receive FIFO Overrun Service Request disabled Write: Unchanged</p> <p>1_B Read: Receive FIFO Overrun Service Request enabled Write: Disable Receive FIFO Overrun Service Request</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
EFAE	8	rwh	Empty FIFO Access Service Request Enable 0 _B Read: Empty FIFO Access Service Request disabled Write: Unchanged 1 _B Read: Empty FIFO Access Service Request enabled Write: Disable Empty FIFO Access Service Request
IIBAE	9	rwh	Illegal Input Buffer Access Service Request Enable 0 _B Read: Illegal Input Buffer Access Service Request disabled Write: Unchanged 1 _B Read: Illegal Input Buffer Access Service Request enabled Write: Disable Illegal Input Buffer Access Service Request
IOBAE	10	rwh	Illegal Output Buffer Access Service Request Enable 0 _B Read: Illegal Output Buffer Access Service Request disabled Write: Unchanged 1 _B Read: Illegal Output Buffer Access Service Request enabled Write: Disable Illegal Output Buffer Access Service Request
MHFE	11	rwh	Message Handler Constraints Flag Service Request Enable 0 _B Read: Message Handler Constraints Flag Service Request disabled Write: Unchanged 1 _B Read: Message Handler Constraints Flag Service Request enabled Write: Disable Message Handler Constraints Flag Service Request
EDAE	16	rwh	Error Detected on Channel A Service Request Enable 0 _B Read: Error Detected on Channel A Service Request disabled Write: Unchanged 1 _B Read: Error Detected on Channel A Service Request enabled Write: Disable Error Detected on Channel A Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
LTVAE	17	rwh	Latest Transmit Violation Channel A Service Request Enable 0 _B Read: Latest Transmit Violation Channel A Service Request disabled Write: Unchanged 1 _B Read: Latest Transmit Violation Channel A Service Request enabled Write: Disable Latest Transmit Violation Channel A Service Request
TABAE	18	rwh	Transmission Across Boundary Channel A Service Request Enable 0 _B Read: Transmission Across Boundary Channel A Service Request disabled Write: Unchanged 1 _B Read: Transmission Across Boundary Channel A Service Request enabled Write: Enable Transmission Across Boundary Channel A Service Request
EDBE	24	rwh	Error Detected on Channel B Service Request Enable 0 _B Read: Error Detected on Channel B Service Request disabled Write: Unchange 1 _B Read: Error Detected on Channel B Service Request enabled Write: Disable Error Detected on Channel B Service Request
LTVBE	25	rwh	Latest Transmit Violation Channel B Service Request Enable 0 _B Read: Latest Transmit Violation Channel B Service Request disabled Write: Unchanged 1 _B Read: Latest Transmit Violation Channel B Service Request enabled Write: Disable Latest Transmit Violation Channel B Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TABBE	26	rwh	<p>Transmission Across Boundary Channel B Service Request Enable</p> <p>0_B Read: Transmission Across Boundary Channel B Service Request disabled Write: Unchanged</p> <p>1_B Read: Transmission Across Boundary Channel B Service Request enabled Write: Disable Transmission Across Boundary Channel B Service Request</p>
0	[15:12], [23:19], [31:27]	r	<p>Reserved</p> <p>Returns 0 if read; should be written with 0.</p>

FlexRay™ Protocol Controller (E-Ray)
Status Service Request Enable Set (SIES)

The settings in the Status Service Request Enable Set register determine which status changes in the Status Service Request Register will result in a service request. The enable bits are set by writing to SIES and reset by writing to SIER. Writing a 1 sets the specific enable bit, a 0 has no effect.

SIES
Status Service Request Enable Set (0038_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						MTS BE	WUP BE	0						MTS AE	WUP AE
r						rwh	rwh	r						rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDS E	MBS IE	SUC SE	SWE E	TOB CE	TIBC E	T1IE	T10E	NMV CE	RFC LE	RFN EE	RXIE	TXIE	CYC SE	CAS E	WST E
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WSTE	0	rwh	Wakeup Status Service Request Enable 0 _B Read: Wake-up Status Service Request disabled Write: Unchanged 1 _B Read: Wake-up Status Service Request enabled Write: Enable Wakeup Status Service Request
CASE	1	rwh	Collision Avoidance Symbol Service Request Enable 0 _B Read: Collision Avoidance Symbol Service Request disabled Write: Unchanged 1 _B Read: Collision Avoidance Symbol Service Request enabled Write: Enable Collision Avoidance Symbol Service Request
CYCSE	2	rwh	Cycle Start Service Request Enable 0 _B Read: Cycle Start Service Request disabled Write: Unchanged 1 _B Read: Cycle Start Service Request enabled Write: Enable Cycle Start Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TXIE	3	rwh	Transmit Service Request Enable 0 _B Read: Transmit Service Request disabled Write: Unchanged 1 _B Transmit Service Request enabled Write: Enable Transmit Service Request
RXIE	4	rwh	Receive Service Request Enable 0 _B Read: Receive Service Request disabled Write: Unchanged 1 _B Read: Receive Service Request enabled Write: Enable Receive Service Request
RFNEE	5	rwh	Receive FIFO Not Empty Service Request Enable 0 _B Read: Receive FIFO Not Empty Service Request disabled Write: Unchanged 1 _B Read: Receive FIFO Not Empty Service Request enabled Write: Enable Receive FIFO Not Empty Service Request
RFCLE	6	rwh	Receive FIFO Critical Level Service Request Enable 0 _B Read: Receive FIFO Critical Level Service Request disabled Write: Unchanged 1 _B Read: Receive FIFO Critical Level Service Request enabled Write: Enable Receive FIFO Critical Level Service Request
NMVCE	7	rwh	Network Management Vector Changed Service Request Enable 0 _B Read: Network Management Vector Changed Service Request disabled Write: Unchanged 1 _B Read: Network Management Vector Changed Service Request enabled Write: Enable Network Management Vector Changed Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TI0E	8	rwh	Timer Service Request 0 Enable 0 _B Read: Timer Service Request 0 disabled Write: Unchanged 1 _B Read: Timer Service Request 0 enabled Write: Enable Timer Service Request 0
TI1E	9	rwh	Timer Service Request 1 Enable 0 _B Read: Timer Service Request 1 disabled Write: Unchanged 1 _B Read: Timer Service Request 1 enabled Write: Enable Timer Service Request 1
TIBCE	10	rwh	Transfer Input Buffer Completed Service Request Enable 0 _B Read: Wakeup Status Service Request disabled Write: Unchanged 1 _B Read: Wakeup Status Service Request enabled Write: Enable Wakeup Status Service Request
TOBCE	11	rwh	Transfer Output Buffer Completed Service Request Enable 0 _B Read: Transfer Input Buffer Completed Service Request disabled Write: Unchanged 1 _B Read: Transfer Input Buffer Completed Service Request enabled Write: Enable Transfer Input Buffer Completed Service Request
SWEE	12	rwh	Stop Watch Event Service Request Enable 0 _B Read: Stop Watch Event Service Request disabled Write: Unchanged 1 _B Read: Stop Watch Event Service Request enabled Write: Enable Stop Watch Event Service Request
SUCSE	13	rwh	Startup Completed Successfully Service Request Enable 0 _B Read: Startup Completed Successfully Service Request disabled Write: Unchanged 1 _B Read: Startup Completed Successfully Service Request enabled Write: Enable Startup Completed Successfully Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
MBSIE	14	rwh	Message Buffer Status Service Request Enable 0 _B Read: Message Buffer Status Service Request disabled Write: Unchanged 1 _B Read: Message Buffer Status Service Request enabled Write: Enable Message Buffer Status Service Request
SDSE	15	rwh	Start of Dynamic Segment Service Request Enable 0 _B Read: Start of Dynamic Service Request disabled Write: Unchanged 1 _B Read: Start of Dynamic Service Request enabled Write: Enable Start of Dynamic Service Request
WUPAE	16	rwh	Wakeup Pattern Channel A Service Request Enable 0 _B Read: Wakeup Pattern Channel A Service Request disabled Write: Unchanged 1 _B Read: Wakeup Pattern Channel A Service Request enabled Write: Enable Wakeup Pattern Channel A Service Request
MTSAE	17	rwh	Media Access Test Symbol Channel A Service Request Enable 0 _B Read: Media Access Test Symbol Channel A Service Request disabled Write: Unchanged 1 _B Read: Media Access Test Symbol Channel A Service Request enabled Write: Enable Media Access Test Symbol Channel A Service Request
WUPBE	24	rwh	Wakeup Pattern Channel B Service Request Enable 0 _B Read: Wakeup Pattern Channel B Service Request disabled Write: Unchanged 1 _B Read: Wakeup Pattern Channel B Service Request enabled Write: Enable Wakeup Pattern Channel B Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
MTSBE	25	rwh	Media Access Test Symbol Channel B Service Request Enable 0 _B Read: Media Access Test Symbol Channel B Service Request disabled Write: Unchanged 1 _B Read: Media Access Test Symbol Channel B Service Request enabled Write: Enable Media Access Test Symbol Channel B Service Request
0	[23:18], [31:26]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)
Status Service Request Enable Reset (SIER)

The settings in the Status Service Request Enable Reset register determine which status changes in the Status Service Request Register will result in a service request. The enable bits are set by writing to SIES and reset by writing to SIER. Writing a 1 resets the specific enable bit, a 0 has no effect.

SIER
Status Service Request Enable Reset(003C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						MTS BE	WUP BE	0						MTS AE	WUP AE
r						rwh	rwh	r						rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDS E	MBS IE	SUC SE	SWE E	TOB CE	TIBC E	T1IE	T10E	NMV CE	RFC LE	RFN EE	RXIE	TXIE	CYC SE	CAS E	WST E
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
WSTE	0	rwh	Wakeup Status Service Request Enable 0 _B Read: Wakeup Status Service Request disabled Write: Unchanged 1 _B Read: Wakeup Status Service Request enabled Write: Disable Wakeup Status Service Request
CASE	1	rwh	Collision Avoidance Symbol Service Request Enable 0 _B Read: Collision Avoidance Symbol Service Request disabled Write: Unchanged 1 _B Read: Collision Avoidance Symbol Service Request enabled Write: Disable Collision Avoidance Symbol Service Request
CYCSE	2	rwh	Cycle Start Service Request Enable 0 _B Read: Cycle Start Service Request disabled Write: Unchanged 1 _B Read: Cycle Start Service Request enabled Write: Disable Cycle Start Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TXIE	3	rwh	Transmit Service Request Enable 0 _B Read: Transmit Service Request disabled Write: Unchanged 1 _B Read: Transmit Service Request enabled Write: Disable Transmit Service Request
RXIE	4	rwh	Receive Service Request Enable 0 _B Read: Receive Service Request disabled Write: Unchanged 1 _B Read: Receive Service Request enabled Write: Disable Receive Service Request
RFNEE	5	rwh	Receive FIFO Not Empty Service Request Enable 0 _B Read: Receive FIFO Not Empty Service Request disabled Write: Unchanged 1 _B Read: Receive FIFO Not Empty Service Request enabled Write: Disable Receive FIFO Not Empty Service Request
RFCLE	6	rwh	Receive FIFO Critical Level Service Request Enable 0 _B Read: Service Request disabled Write: Unchanged 1 _B Read: Receive FIFO Critical Level Service Request enabled Write: Disable Receive FIFO Critical Level Service Request
NMVCE	7	rwh	Network Management Vector Changed Service Request Enable 0 _B Read: Network Management Vector Changed Service Request disabled Write: Unchanged 1 _B Read: Network Management Vector Changed Service Request enabled Write: Disable Network Management Vector Changed Service Request
TIOE	8	rwh	Timer Service Request 0 Enable 0 _B Read: Timer Service Request 0 disabled Write: Unchanged 1 _B Read: Timer Service Request 0 enabled Write: Disable Service Request 0

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TI1E	9	rwh	Timer Service Request 1 Enable 0 _B Read: Timer Service Request 1 disabled Write: Unchanged 1 _B Read: Timer Service Request 1 enabled Write: Disable Timer Service Request 1
TIBCE	10	rwh	Transfer Input Buffer Completed Service Request Enable 0 _B Read: Wakeup Status Service Request disabled Write: Unchanged 1 _B Read: Wakeup Status Service Request enabled Write: Disable Wakeup Status Service Request
TOBCE	11	rwh	Transfer Output Buffer Completed Service Request Enable 0 _B Read: Transfer Input Buffer Completed Service Request disabled Write: Unchanged 1 _B Read: Transfer Input Buffer Completed Service Request enabled Write: Disable Transfer Input Buffer Completed Service Request
SWEE	12	rwh	Stop Watch Event Service Request Enable 0 _B Read: Stop Watch Event Service Request disabled Write: Unchanged 1 _B Read: Stop Watch Event Service Request enabled Write: Disable Stop Watch Event Service Request
SUCSE	13	rwh	Startup Completed Successfully Service Request Enable 0 _B Read: Startup Completed Successfully Service Request disabled Write: Unchanged 1 _B Read: Startup Completed Successfully Service Request enabled Write: Disable Startup Completed Successfully Service Request
MBSIE	14	rwh	Message Buffer Status Service Request Enable 0 _B Read: Message Buffer Status Service Request disabled Write: Unchanged 1 _B Read: Message Buffer Status Service Request enabled Write: Disable Message Buffer Status Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
SDSE	15	rwh	Start of Dynamic Segment Service Request Enable 0 _B Read: Start of Dynamic Service Request disabled Write: Unchanged 1 _B Read: Start of Dynamic Service Request enabled Write: Disable Start of Dynamic Service Request
WUPAE	16	rwh	Wakeup Pattern Channel A Service Request Enable 0 _B Read: Wakeup Pattern Channel A Service Request disabled Write: Unchanged 1 _B Read: Wakeup Pattern Channel A Service Request enabled Write: Disable Wakeup Pattern Channel A Service Request
MTSAE	17	rwh	Media Access Test Symbol Channel A Service Request Enable 0 _B Read: Media Access Test Symbol Channel A Service Request disabled Write: Unchanged 1 _B Read: Media Access Test Symbol Channel A Service Request enabled Write: Disable Media Access Test Symbol Channel A Service Request
WUPBE	24	rwh	Wakeup Pattern Channel B Service Request Enable 0 _B Read: Wakeup Pattern Channel B Service Request disabled Write: Unchanged 1 _B Read: Wakeup Pattern Channel B Service Request enabled Write: Disable Wakeup Pattern Channel B Service Request
MTSBE	25	rwh	Media Access Test Symbol Channel B Service Request Enable 0 _B Read: Media Access Test Symbol Channel B Service Request disabled Write: Unchanged 1 _B Read: Media Access Test Symbol Channel B Service Request enabled Write: Disable Media Access Test Symbol Channel B Service Request

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
0	[23:18], [31:26]	r	Reserved Returns 0 if read; should be written with 0.

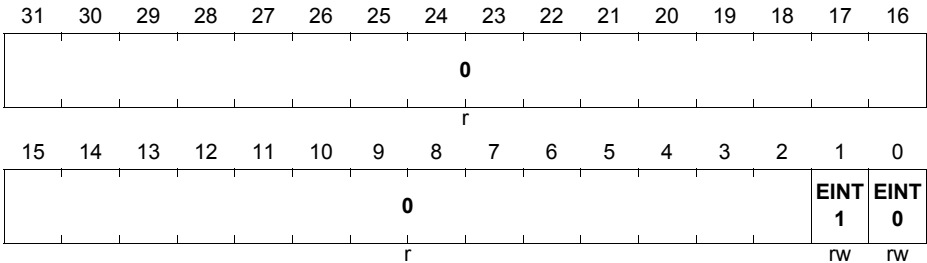
FlexRay™ Protocol Controller (E-Ray)

Service Request Line Enable (ILE)

Each of the two service request lines to the Host INT0SRC, INT1SRC can be enabled / disabled separately by programming bit EINT0 and EINT1.

ILE

Service Request Line Enable (0040_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
EINT0	0	rw	Enable Service Request Line 0 (INT0SRC) 0 _B Service Request line disabled 1 _B Service Request line enabled
EINT1	1	rw	Enable Service Request Line 1 (INT1SRC) 0 _B Service Request line disabled 1 _B Service Request line enabled
0	[31:2]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)
Timer 0 Configuration (T0C)

Absolute timer. Specifies in terms of cycle count and MacroTICK the point in time when the timer 0 service request occurs. When the timer 0 service request is asserted, output signal TINT0SR is set to 1 for the duration of one MacroTICK and SIR.TI0 is set to 1.

Timer 0 can be activated as long as the POC is either in “NORMAL_ACTIVE” state or in “NORMAL_PASSIVE” state. Timer 0 is deactivated when leaving “NORMAL_ACTIVE” state or “NORMAL_PASSIVE” state except for transitions between the two states.

Before reconfiguration of the timer, the timer has to be halted first by writing 0 to bit T0RC.

T0C
Timer 0 Configuration (0044_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		T0MO													
r		rw													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		T0CC					0					T0M	T0R		
r		rw					r					rw	rwh		

Field	Bits	Type	Description
T0RC	0	rwh	Timer 0 Run Control 0 _B Timer 0 halted 1 _B Timer 0 running
T0MS	1	rw	Timer 0 Mode Select 0 _B Single-shot mode 1 _B Continuous mode
T0CC	[14:8]	rw	Timer 0 Cycle Code The 7-bit timer 0 cycle code determines the cycle set used for generation of the timer 0 service request. For details about the configuration of the cycle code see “Cycle Counter Filtering” on Page 25-215 .
T0MO	[29:16]	rw	Timer 0 MacroTICK Offset Configures the MacroTICK offset from the beginning of the cycle where the service request is to occur. The Timer 0 Service Request occurs at this offset for each cycle of the cycle set.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
0	[7:2], 15, [31:30]	r	Reserved Returns 0 if read; should be written with 0.

Note: The configuration of timer 0 is compared against the Macrotick counter value, there is no separate counter for timer 0. In case the Communication Controller leaves "NORMAL_ACTIVE" or "NORMAL_PASSIVE" state, or if timer 0 is halted by Host command, output signal TINT0SR is reset to 0 immediately.

FlexRay™ Protocol Controller (E-Ray)
Timer 1 Configuration (T1C)

Relative timer. After the specified number of MacroTicks has expired, the timer 1 service request is asserted, output signal TINT1SR is set to 1 for the duration of one MacroTick and SIR.T11 is set to 1.

Timer 1 can be activated as long as the POC is either in “NORMAL_ACTIVE” state or in “NORMAL_PASSIVE” state. Timer 1 is deactivated when leaving “NORMAL_ACTIVE” state or “NORMAL_PASSIVE” state except for transitions between the two states.

Before reconfiguration of the timer, the timer has to be halted first by resetting bit T1RC to 0.

T1C
Timer 1 Configuration (0048_H)
Reset Value: 0002 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		T1MC													
r		rw													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														T1M	T1R
r														rw	rwh

Field	Bits	Type	Description
T1RC	0	rwh	Timer 1 Run Control 0 _B Timer 1 halted 1 _B Timer 1 running
T1MS	1	rw	Timer 1 Mode Select 0 _B Single-shot mode 1 _B Continuous mode
T1MC	[29:16]	rw	Timer 1 MacroTick Count When the configured MacroTick count is reached the timer 1 service request is generated. Valid values are: 2 _H ... 3FFF _H MacroTicks in continuous mode 1 _H ... 3FFF _H MacroTicks in single-shot mode
0	[15:2], [31:30]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Note: In case the Communication Controller leaves “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state, or if timer 1 is halted by Host command, output signal TINT1SR is reset to 0 immediately.

FlexRay™ Protocol Controller (E-Ray)
Stop Watch Register 1 (STPW1)

The stop watch is activated by a rising or falling edge on signal STPW, by a service request 0 or 1 event (rising edge on signal INT0SR or INT1SR) or by the Host by writing bit STPW1.SSWT to 1. With the MacroTICK counter increment following next to the stop watch activation the actual cycle counter and MacroTICK value are captured in the Stop Watch Register 1 STPW1 while the slot counter values for channel A and B are captured in the Stop Watch Register 2 STPW2.

STPW1
Stop Watch Register 1
(004C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SMTV													
r		rh													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SCCV						0	EINT	EINT	EET	SSW	EDG	SWM	ESW
r		rh						r	rw	rw	rw	rwh	rw	rw	rwh

Field	Bits	Type	Description
ESWT	0	rwh	Enable Stop Watch Trigger If enabled an edge on input STPW or a service request 0 or 1 event (rising edge on signal INT0SR or INT1SR) activates the stop watch. In single-shot mode this bit is reset to 0 after the actual cycle counter and MacroTICK value are stored in the Stop Watch register. 0 _B Stop watch trigger disabled 1 _B Stop watch trigger enabled
SWMS	1	rw	Stop Watch Mode Select It is not possible to change the Stop Watch Mode during enabled stop watch trigger (STPW1.ESWT) 0 _B Single-shot mode 1 _B Continuous mode
EDGE	2	rw	Stop Watch Trigger Edge Select 0 _B Falling Edge 1 _B Rising Edge

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
SSWT	3	rwh	Software Stop Watch Trigger When the Host writes this bit to 1 the stop watch is activated. After the actual cycle counter and Macrotick value are stored in the Stop Watch register this bit is reset to 0. The bit is only writeable while ESWT = 0. 0 _B Software trigger reset 1 _B Stop watch activated by software trigger
EETP	4	rw	Enable External Trigger Pin Enables stop watch trigger event via signal STPW if ESWT = 1. 0 _B Stop watch trigger via signal STPW disabled 1 _B Edge on signal STPW triggers stop watch
EINT0	5	rw	Enable Service Request 0 Trigger Enables stop watch trigger by service request 0 event if ESWT = 1. 0 _B Stop watch trigger by service request 0 disabled 1 _B Service Request 0 event triggers stop watch
EINT1	6	rw	Enable Service Request 1 Trigger Enables stop watch trigger by service request 1 event if ESWT = 1. 0 _B Stop watch trigger by service request 1 disabled 1 _B Service Request 1 event triggers stop watch
SCCV	[13:8]	rh	Stopped Cycle Counter Value State of the cycle counter when the stop watch event occurred. Valid values are: 0...3F _H Valid Values
SMTV	[29:16]	rh	Stopped Macrotick Value State of the Macrotick counter when the stop watch event occurred. Valid values are: 0...3F _H Valid Values
0	7, [15:14], [31:30]	r	Reserved Returns 0 if read; should be written with 0.

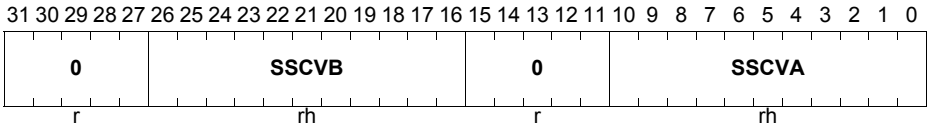
Note: Bits ESWT and SSWT cannot be set to 1 simultaneously. In this case the write access is ignored, and both bits keep their previous values. Therefore either the external stop watch triggers or the software stop watch trigger may be used.

FlexRay™ Protocol Controller (E-Ray)

Stop Watch Register 2 (STPW2)

STPW2

Stop Watch Register 2 (0050_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
SSCVA	[10:0]	rh	Stop Watch Captured Slot Counter Value Channel A State of the slot counter for channel A when the stop watch event occurred. Valid values are 0 to 2047 (0 _H to 7FF _H).
SSCVB	[26:16]	rh	Stop Watch Captured Slot Counter Value Channel B State of the slot counter for channel B when the stop watch event occurred. Valid values are 0 to 2047 (0 _H to 7FF _H).
0	[15:11], [31:27]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
CMD	[3:0]	rwh	<p>CHI Command Vector</p> <p>The host may write any CHI command at any time, but certain commands are only enabled in specific POC states. A disabled command will not be executed, the CHI command vector CMD will be reset to 0000_B = "COMMAND_NOT_ACCEPTED", and flag EIR.CNA in the Error Service Request register will be set to 1. In case the previous CHI command has not yet completed, EIR.CCL is set to 1 together with EIR.CNA; the CHI command needs to be repeated. Except for HALT state, POC state change command applied while the Communication Controller is already in the requested POC state will be ignored.</p> <p>0000_B COMMAND_NOT_ACCEPTED" 0001_B CONFIG 0010_B READY 0011_B WAKEUP 0100_B RUN 0101_B ALL_SLOTS 0110_B HALT 0111_B FREEZE 1000_B SEND_MTS 1001_B ALLOW_COLDSTART 1010_B RESET_STATUS_INDICATORS 1011_B MONITOR_MODE 1100_B CLEAR_RAMs 1101_B Reserved 1110_B Reserved 1111_B Reserved</p> <p>Reading SUCC1.CMD shows whether the last CHI command was accepted. CCSV.POCS monitors the actual POC state. The reserved CHI commands code hardware test functions.</p>
PBSY	7	rh	<p>POC Busy</p> <p>Signals that the POC is busy and cannot accept a command from the Host. SUCC1.CMD is locked against write accesses. Set to 1 after hard reset during initialization of internal RAM blocks.</p> <p>0_B POC not busy, SUCC1.CMD writable 1_B POC is busy, SUCC1.CMD locked</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TXST	8	rw	Transmit Startup Frame in Key Slot^{1) 2)} (pKeySlotUsedForStartup) Defines whether the key slot is used to transmit startup Frames. The bit can be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 0 _B No Startup Frame transmission in key slot, node is non-coldstarter 1 _B Key slot used to transmit startup Frame, node is leading or following coldstarter
TXSY	9	rw	Transmit SYNC Frame in Key Slot^{1) 2)} (pKeySlotUsedForSync) Defines whether the key slot is used to transmit SYNC Frames. The bit can be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 0 _B No SYNC Frame transmission in key slot, node is neither sync nor coldstart node 1 _B Key slot used to transmit SYNC Frames, node is sync node
CSA	[15:11]	rw	Cold Start Attempts¹⁾ (gColdStartAttempts) Configures the maximum number of attempts that a cold starting node is permitted to try to start up the network without receiving any valid response from another node. It can be modified in “DEFAULT_CONFIG” or “CONFIG” state only. Must be identical in all nodes of a cluster. Valid values are 2 to 31.
PTA	[20:16]	rw	Passive to Active¹⁾ (pAllowPassiveToActive) Defines the number of consecutive even / odd cycle pairs that must have valid clock correction terms before the Communication Controller is allowed to transit from “NORMAL_PASSIVE” to “NORMAL_ACTIVE” state. If set to 0000 _B the Communication Controller is not allowed to transit from “NORMAL_PASSIVE” to “NORMAL_ACTIVE” state. It can be modified in “DEFAULT_CONFIG” or “CONFIG” state only. Valid values are 0 to 31 even / odd cycle pairs.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
WUCS	21	rw	<p>Wakeup Channel Select¹⁾ (pWakeupChannel) With this bit the Host selects the channel on which the Communication Controller sends the Wakeup pattern. The Communication Controller ignores any attempt to change the status of this bit when not in “DEFAULT_CONFIG” or “CONFIG” state.</p> <p>0_B Send wakeup pattern on channel A 1_B Send wakeup pattern on channel B</p>
TSM	22	rw	<p>Transmission Slot Mode¹⁾ (pSingleSlotEnabled) Selects the initial transmission slot mode. In SINGLE slot mode the Communication Controller may only transmit in the preconfigured key slot. The key slot ID is configured in the Header Section of Message Buffer 0 respectively Message Buffers 0 and 1 depending on bit MRC.SPLM. In case SUCC1.TSM = 1, Message Buffer 0 respectively Message Buffers 0,1 can be (re)configured in “DEFAULT_CONFIG” or “CONFIG” state only. In ALL slot mode the Communication Controller may transmit in all slots. The bit can be written in “DEFAULT_CONFIG” or “CONFIG” state only. The communication controller changes to ALL slot mode when the Host successfully applied the ALL_SLOTS command by writing SUCC1.CMD = 0101_B in POC states “NORMAL_ACTIVE” or “NORMAL_PASSIVE”. The actual slot mode is monitored by CCSV.SLM.</p> <p>0_B ALL Slot Mode 1_B SINGLE Slot Mode (default after application reset)</p>
HCSE	23	rw	<p>Halt due to Clock Sync Error¹⁾ (pAllowHaltDueToClock) Controls the transition to “HALT” state due to a clock synchronization error. The bit can be modified in “DEFAULT_CONFIG” or “CONFIG” state only.</p> <p>0_B Communication Controller will enter / remain in “NORMAL_PASSIVE” 1_B Communication Controller will enter “HALT” state</p>
MTSA	24	rw	<p>Select Channel A for MTS Transmission^{1) 3)} The bit selects channel A for MTS symbol transmission. The flag is reset by default and may be modified only in “DEFAULT_CONFIG” or “CONFIG” state.</p> <p>0_B Channel A disabled for MTS transmission 1_B Channel A selected for MTS transmission</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
MTSB	25	rw	Select Channel B for MTS Transmission^{1) 3)} The bit selects channel B for MTS symbol transmission. The flag is reset by default and may be modified only in “DEFAULT_CONFIG” or “CONFIG” state. 0 _B Channel B disabled for MTS transmission 1 _B Channel B selected for MTS transmission
CCHA	26	rw	Connected to Channel A¹⁾ (pChannels) Configures whether the node is connected to channel A. 0 _B Not connected to channel A 1 _B Node connected to channel A (default after application reset)
CCHB	27	rw	Connected to Channel B¹⁾ (pChannels) Configures whether the node is connected to channel B. 0 _B Not connected to channel B 1 _B Node connected to channel B (default after application reset)
0	[6:4], 10, [31:28]	r	Reserved Returns 0 if read; should be written with 0.

1) This bit can be updated in “DEFAULT_CONFIG” or “CONFIG” state only!

2) The protocol requires that both bits TXST and TXSY are set for coldstart nodes.

3) MTSa and MTSB may also be changed outside “DEFAULT_CONFIG” or “CONFIG” state when the write to SUCC1 register is directly preceded by the unlock sequence as described in Lock Register (LCK). This may be combined with CHI command “SEND_MTS”. If both bits MTSa and MTSB are set to 1 an MTS symbol will be transmitted on both channels when requested by writing SUCC1.CMD = 1000_g.

COMMAND_NOT_ACCEPTED

SUCC1.CMD is reset to 0000_B due to one of the following conditions:

- Illegal command applied by the Host
- Host writes command_not_accepted
- Host applied new command while execution of the previous Host command has not completed

When SUCC1.CMD is reset to 0000_B, bit EIR.CNA in the Error Service Request register is set, and - if enabled - an service request is generated. Commands which are not accepted are not executed.

CONFIG

Go to POC state “CONFIG” when called in POC states “DEFAULT_CONFIG“, “READY“, or in “MONITOR_MODE“. When called in “HALT” state transits to POC state

FlexRay™ Protocol Controller (E-Ray)

“DEFAULT_CONFIG“. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED“.

READY

Go to POC state “READY” when called in POC states “CONFIG”, “NORMAL_ACTIVE”, “NORMAL_PASSIVE”, “STARTUP”, or “WAKEUP”. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED“.

WAKEUP

Go to POC state WAKEUP when called in POC state “READY”. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED“.

RUN

Go to POC state “STARTUP” when called in POC state “READY”. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED“.

ALL_SLOTS

Leave SINGLE slot mode after successful startup / integration at the next end of cycle when called in POC states “NORMAL_ACTIVE” or “NORMAL_PASSIVE”. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED“.

HALT

Set the halt request CCSV.HRQ bit in the Communication Controller Status Vector register and go to POC state “HALT” at the next end of cycle when called in POC states “NORMAL_ACTIVE” or “NORMAL_PASSIVE“. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED“.

FREEZE

Set the freeze status indicator CCSV.FSI and go to POC state “HALT” immediately. Can be called from any state.

SEND_MTS

Send single MTS symbol during the next following symbol window on the channel configured by SUCC1.MTSA, SUCC1.MTSB, when called in POC state “NORMAL_ACTIVE“. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED“.

FlexRay™ Protocol Controller (E-Ray)**ALLOW_COLDSTART**

The command resets bit CCSV.CSI to enable the node to become cold starter. When called in states “DEFAULT_CONFIG”, “CONFIG”, “HALT”, or “MONITOR_MODE”. SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”. To become leading coldstarter it is also required that both TXST and TXSY are set.

RESET_STATUS_INDICATORS

Resets status flags CCSV.CSNI, CCSV.CSAI, CCSV.WSV to their default values. May be called in POC state READY. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”.

MONITOR_MODE

Enter MONITOR_MODE when called in POC state CONFIG. In this mode the Communication Controller is able to receive FlexRay™ Frames and wakeup pattern. It is also able to detect coding errors. The temporal integrity of received Frames is not checked. This mode can be used for debugging purposes, e.g. in case that the startup of a FlexRay™ network fails. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”. For details see [“MONITOR_MODE” on Page 25-199](#).

CLEAR_RAMs

Sets bit MHDS.CRAM in the Message Handler Status register when called in “DEFAULT_CONFIG” or “CONFIG” state. When called in any other state, SUCC1.CMD will be reset to 0000_B = “COMMAND_NOT_ACCEPTED”. By setting MHDS.CRAM all internal RAM blocks are initialized to zero. Note that only the currently active IBF bank is cleared. To clear the 2nd bank as well, CUST1.IBF1PAG and CUST1.IBF2PAG need to be set and command CLEAR_RAMs need to be issued again. This is required in particular after an application reset. If the 2nd bank of IBF is left unused, this procedure is not required. During the initialization of the RAMs, SUCC1.PBSY will show POC busy. Access to the configuration and status registers is possible during execution of CHI command CLEAR_RAMs.

The initialization of the E-Ray internal RAM blocks requires 2048 f_{CLC_ERAY} cycles. There should be no Host access to IBF or OBF during initialization of the internal RAM blocks after application reset or after assertion of CHI command CLEAR_RAMs. Before asserting CHI command CLEAR_RAMs the Host should make sure that no transfer between Message RAM and IBF / OBF or the Transient Buffer RAMs is ongoing. This command also resets the Message Buffer Status registers MHDS, LDTS, FSR, MHDF, TXRQ1, TXRQ2, TXRQ3, TXRQ4, NDAT1, NDAT2, NDAT3, NDAT4, MBSC1, MBSC2, MBSC3, and MBSC4.

FlexRay™ Protocol Controller (E-Ray)

Note: All accepted commands with exception of CLEAR_RAMs and SEND_MTS will cause a change of register CCSV after at most 8 cycles of the slower of the two clocks f_{CLC_ERAY} and f_{SCLK} , assumed that POC was not busy when the command was applied and that no POC state change was forced by bus activity in that time Frame. Reading register CCSV will show data that is delayed by synchronization from f_{SCLK} to f_{CLC_ERAY} domain and by the Host-specific CPU interface.

Table 25-4 below references the CHI commands from the FlexRay™ Protocol Specification v2.1 (section 2.2.1.1, Table 2-2) to the E-Ray CHI command vector CMD.]

Table 25-4 Reference to CHI Host command summary from FlexRay™ protocol specification

CHI Command	Where processed (POC State)	CHI Command Vector CMD
ALL_SLOT	POC:NORMAL_ACTIVE, POC:NORMAL_PASSIVE	ALL_SLOTS
ALLOW_COLDSTART	All except POC:DEFAULT_CONFIG, POC:CONFIG, POC:HALT	ALLOW_COLDSTART
CONFIG	POC:DEFAULT_CONFIG, POC:READY	CONFIG
CONFIG_COMPLETE	POC:CONFIG	Unlock sequence & READY
DEFAULT_CONFIG	POC:HALT	CONFIG
FREEZE	All	FREEZE
HALT	POC:NORMAL_ACTIVE, POC:NORMAL_PASSIVE	HALT
READY	All except POC:DEFAULT_CONFIG, POC:CONFIG, POC:READY, POC:HALT	READY
RUN	POC:READY	RUN
WAKEUP	POC:READY	WAKEUP

FlexRay™ Protocol Controller (E-Ray)
SUC Configuration Register 2 (SUCC2)

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

SUCC2

SUC Configuration Register 2 (0084_H) Reset Value: 0100 0504_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		LTN				0		LT																							
r		rw				r		rw																							

Field	Bits	Type	Description
LT	[20:0]	rw	Listen Timeout¹⁾ (pdListenTimeout) Configures wakeup / startup listen timeout in Microticks. The range for wakeup / startup listen timeout (pdListenTimeout) is 1284 to 1283846 (504 _H to 139706 _H) Microticks
LTN	[27:24]	rw	Listen Time-out Noise¹⁾ (gListenNoise - 1) Configures the upper limit for startup and wakeup listen timeout in the presence of noise expressed as a multiple of the cluster constant pdListenTimeout. The range of pdListenTimeout 2 to 16. LTN must be configured identical in all nodes of a cluster. 1 _H Listen Time-out Noise is equal 2 2 _H Listen Time-out Noise is equal 3 ... _H ... F _H Listen Time-out Noise is equal 16
0	[23:21], [31:28]	r	Reserved Returns 0 if read; should be written with 0.

1) This bit can be updated in “DEFAULT_CONFIG” or “CONFIG” state only!

*Note: The wakeup / startup noise time-out is calculated as follows:
The wakeup / startup noise time-out = pdListenTimeout • gListenNoise
= LT • (LTN+ 1)*

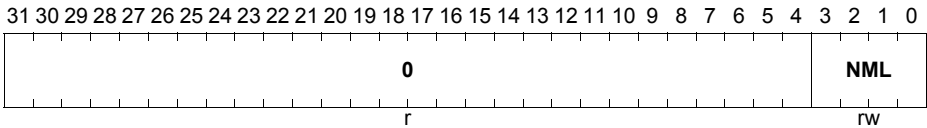
FlexRay™ Protocol Controller (E-Ray)

NEM Configuration Register (NEMC)

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

NEMC

NEM Configuration Register (008C_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
NML	[3:0]	rw	Network Management Vector Length¹⁾ (gNetworkManagementVectorLength) These bits configure the length of the NM Vector. The configured length must be identical in all nodes of a cluster. Valid values are 0 to 12 (0 _H to C _H) bytes.
0	[31:4]	r	Reserved Returns 0 if read; should be written with 0.

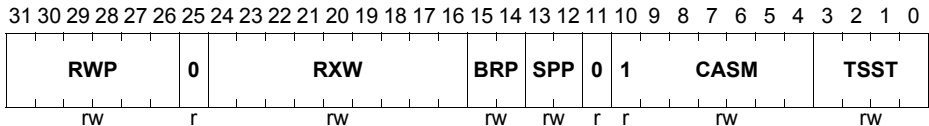
1) This bit can be updated in “DEFAULT_CONFIG” or “CONFIG” state only!

FlexRay™ Protocol Controller (E-Ray)
PRT Configuration Register 1 (PRTC1)

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

PRTC1

PRT Configuration Register 1 (0090_H) Reset Value: 084C 0633_H



Field	Bits	Type	Description
TSST	[3:0]	rw	Transmission Start Sequence Transmitter¹⁾ (gdTSSTransmitter) Configures the duration of the Transmission Start Sequence (TSS) in terms of Bit Times (1 bit time = 4 Microticks = 100ns at 10Mbps). Must be identical in all nodes of a cluster. Valid values are 3 to 15 (3 _H to F _H) Bit Times.
CASM	[10:4]	rw	Collision Avoidance Symbol Maximum¹⁾ (gdCASRxLowMax) Configures the upper limit of the acceptance window for a collision avoidance symbol (CAS). Valid values are 67 to 99 (43 _H to 63 _H). Most significant bit of CASM is hard wired to 1 and can not be modified.
SPP	[13:12]	rw	Strobe Point Position¹⁾ Defines the sample count value for strobing. The strobed bit value is set to the voted value when the sample count is incremented to the value configured by SPP. 00 _B Sample 5 (default) 01 _B Sample 4 10 _B Sample 6 11 _B Reserved; should not be used. <i>Note: The current revision 2.1 of the FlexRay™ protocol requires that SPP = 00_B. The alternate strobe point positions could be used to compensate for asymmetries in the physical layer.</i>

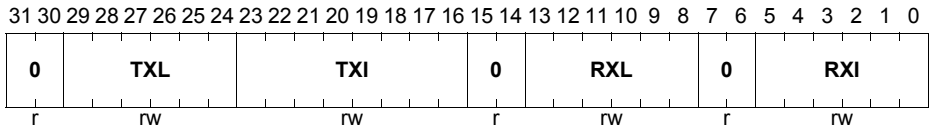
FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
BRP	[15:14]	rw	<p>Baud Rate Prescaler¹⁾ (gdSampleClockPeriod, pSamplePerMicrotick) The baud rate prescaler configures the baud rate on the FlexRay™ bus. The baud rates listed below are valid with a sample clock $f_{SCLK} = 80$ MHz. One bit time always consists of 8 samples independent of the configured baud rate.</p> <p>00_B 10 Mbit/s (1 Microtick= 25 ns; twice sampled with f_{SCLK}) gdSampleClockPeriod = 12.5 ns = $1 / f_{SCLK}$ pSamplesPerMicrotick = 2</p> <p>01_B 5 Mbit/s (1 Microtick= 25ns; single sampled with $f_{SCLK} / 2$) gdSampleClockPeriod = 25 ns = $2 / f_{SCLK}$ pSamplesPerMicrotick = 1</p> <p>10_B 2.5 Mbit/s (1 Microtick = 50ns; single sampled with $f_{SCLK} / 4$) gdSampleClockPeriod = 50 ns = $4 / f_{SCLK}$ pSamplesPerMicrotick = 1</p> <p>11_B Reserved; should not be used (2.5 Mbit/s (1 Microtick = 50 ns; single sampled with $f_{SCLK} / 4$) gdSampleClockPeriod = 50 ns = $4 / f_{SCLK}$ pSamplesPerMicrotick = 1</p>
RXW	[24:16]	rw	<p>Wakeup Symbol Receive Window Length¹⁾ (gdWakeupSymbolRxWindow) Configures the number of Bit Times used by the node to test the duration of the received wakeup pattern. Must be identical in all nodes of a cluster. Valid values are 76 to 301 (4C_H to 12D_H) Bit Times.</p>
RWP	[31:26]	rw	<p>Repetitions of Tx Wakeup Pattern¹⁾ (pWakeupPattern) Configures the number of repetitions (sequences) of the Tx wakeup symbol. Valid values are 2 to 63 (2_H to 3F_H).</p>
0	11, 25	r	<p>Reserved Returns 0 if read; should be written with 0.</p>

1) This bit can be updated in "DEFAULT_CONFIG" or "CONFIG" state only!

FlexRay™ Protocol Controller (E-Ray)
PRT Configuration Register 2 (PRTC2)

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

PRTC2
PRT Configuration Register 2 (0094_H) Reset Value: 0F2D 0A0E_H


Field	Bits	Type	Description
RXI	[5:0]	rw	Wakeup Symbol Receive Idle¹⁾ (gdWakeupSymbolRxIdle) Configures the number of Bit Times used by the node to test the duration of the idle phase of the received wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 14 to 59 (E _H to 3B _H) Bit Times.
RXL	[13:8]	rw	Wakeup Symbol Receive Low¹⁾ (gdWakeupSymbolRxLow) Configures the number of Bit Times used by the node to test the duration of the low phase of the received wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 10 to 55 (A _H to 37 _H) Bit Times.
TXI	[23:16]	rw	Wakeup Symbol Transmit Idle¹⁾ (gdWakeupSymbolTxIdle) Configures the number of Bit Times used by the node to transmit the idle phase of the wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 45 to 180 (2D _H to B4 _H) Bit Times.
TXL	[29:24]	rw	Wakeup Symbol Transmit Low¹⁾ (gdWakeupSymbolTxLow) Configures the number of Bit Times used by the node to transmit the low phase of the wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 15 to 60 (F _H to 3C _H) Bit Times.
0	[7:6], [15:14], [31:30]	r	Reserved Returns 0 if read; should be written with 0.

1) This bit can be updated in “DEFAULT_CONFIG” or “CONFIG” state only!

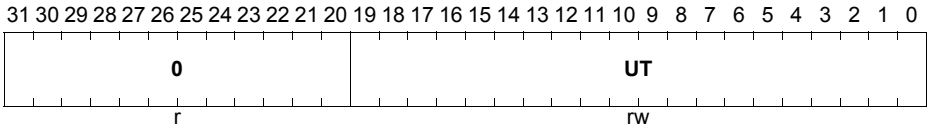
FlexRay™ Protocol Controller (E-Ray)

GTU Configuration Register 1 (GTUC01)

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC01

GTU Configuration Register 1 (00A0_H) Reset Value: 0000 0280_H



Field	Bits	Type	Description
UT	[19:0]	rw	Microtick per Cycle (pMicroPerCycle) ¹⁾ Configures the duration of the communication cycle in Microticks. Valid values are 640 to 640000 (280 _H to 9C400 _H) Microticks.
0	[31:20]	r	Reserved Returns 0 if read; should be written with 0.

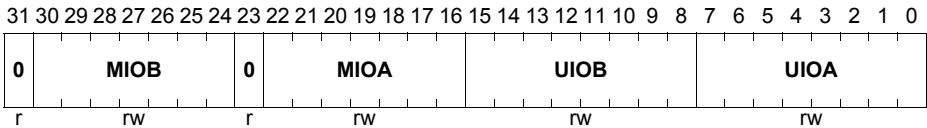
1) This bit can be updated in “DEFAULT_CONFIG” or “CONFIG” state only!

FlexRay™ Protocol Controller (E-Ray)
GTU Configuration Register 3 (GTUC03)

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC03

GTU Configuration Register 3 (00A8_H) **Reset Value: 0202 0000_H**



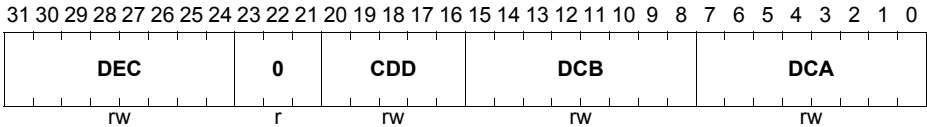
Field	Bits	Type	Description
UIOA	[7:0]	rw	Microtick Initial Offset Channel A¹⁾ (pMicroInitialOffset[A]) Configures the number of Microticks between the actual time reference point on channel A and the subsequent Macrotick boundary of the secondary time reference point. The parameter depends on pDelayCompensation[A] and therefore has to be set for each channel independently. Valid values are 0 to 240 (0 _H to F0 _H) Microticks.
UIOB	[15:8]	rw	Microtick Initial Offset Channel B¹⁾ (pMicroInitialOffset[B]) Configures the number of Microticks between the actual time reference point on channel B and the subsequent Macrotick boundary of the secondary time reference point. The parameter depends on pDelayCompensation[B] and therefore has to be set for each channel independently. Valid values are 0 to 240 (0 _H to F0 _H) Microticks.
MIOA	[22:16]	rw	Macrotick Initial Offset Channel A (gMacroInitialOffset[A]) ¹⁾ Configures the number of Macroticks between the static slot boundary and the subsequent Macrotick boundary of the secondary time reference point based on the nominal Macrotick duration. Must be identical in all nodes of a cluster. Valid values are 2 to 72 (2 _H to 48 _H) Macroticks.
MIOB	[30:24]	rw	Macrotick Initial Offset Channel B (gMacroInitialOffset[B]) ¹⁾ Configures the number of Macroticks between the static slot boundary and the subsequent Macrotick boundary of the secondary time reference point based on the nominal Macrotick duration. Must be identical in all nodes of a cluster. Valid values are 2 to 72 (2 _H to 48 _H) Macroticks.

FlexRay™ Protocol Controller (E-Ray)
GTU Configuration Register 5 (GTUC05)

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

GTUC05

GTU Configuration Register 5 (00B0_H) **Reset Value: 0E00 0000_H**



Field	Bits	Type	Description
DCA	[7:0]	rw	Delay Compensation Channel A¹ (pDelayCompensation[A]) Used to compensate for reception delays on channel A. This covers assumed propagation delay up to cPropagationDelayMax for Microticks in the range of 0.0125μs to 0.05μs. In practice, the minimum of the propagation delays of all sync nodes should be applied. Valid values are 0 to 200 (0 _H to C8 _H) Microticks.
DCB	[15:8]	rw	Delay Compensation Channel B¹ (pDelayCompensation[B]) Used to compensate for reception delays on channel B. This covers assumed propagation delay up to cPropagationDelayMax for Microticks in the range of 0.0125 to 0.05μs. In practice, the minimum of the propagation delays of all sync nodes should be applied. Valid values are 0 to 200 (0 _H to C8 _H) Microticks.
CDD	[20:16]	rw	Cluster Drift Damping (pClusterDriftDamping) ¹ Configures the cluster drift damping value used in clock synchronization to minimize accumulation of rounding errors. Valid values are 0 to 20 (0 _H to 14 _H) Microticks.
DEC	[31:24]	rw	Decoding Correction (pDecodingCorrection) ¹ Configures the decoding correction value used to determine the primary time reference point. Valid values are 14 to 143 (E _H to 8F _H) Microticks.
0	[23:21]	r	Reserved Returns 0 if read; should be written with 0.

¹⁾ This bit can be updated in “DEFAULT_CONFIG” or “CONFIG” state only!

FlexRay™ Protocol Controller (E-Ray)
GTU Configuration Register 9 (GTUC09)

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

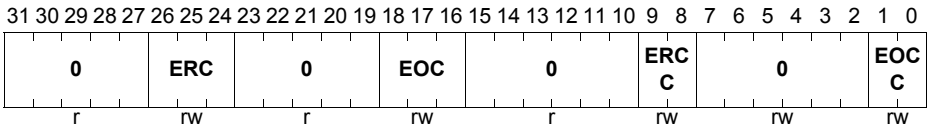
GTUC09

GTU Configuration Register 9 (00C0_H) **Reset Value: 0000 0101_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0																DSI	0	MAPO				0	APO								
r																rw	r	rw				r	rw								

Field	Bits	Type	Description
APO	[5:0]	rw	Action Point Offset (gdActionPointOffset) ¹⁾ Configures the action point offset in Macroticks within static slots and symbol window. Must be identical in all nodes of a cluster. Valid values are 1 to 63 (1 _H to 3F _H) Macroticks.
MAPO	[12:8]	rw	Minislot Action Point Offset ¹⁾ (gd Minislot Action Point Offset) Configures the action point offset in Macroticks within the minislots of the dynamic segment. Must be identical in all nodes of a cluster. Valid values are 1 to 31 (1 _H to 1F _H) Macroticks.
DSI	[17:16]	rw	Dynamic Slot Idle Phase ¹⁾ (gdDynamicSlotIdlePhase) The duration of the dynamic slot idle phase has to be greater or equal than the idle detection time. Must be identical in all nodes of a cluster. Valid values are 0 to 2 Minislot.
0	[7:6], [15:13], [31:18]	r	Reserved Returns 0 if read; should be written with 0.

1) This bit can be updated in “DEFAULT_CONFIG” or “CONFIG” state only!

FlexRay™ Protocol Controller (E-Ray)
GTU Configuration Register 11 (GTUC11)
GTUC11
GTU Configuration Register 11
(00C8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
EOCC	[1:0]	rw	<p>External Offset Correction Control (pExternOffsetControl) By writing to EOCC the external offset correction is enabled as specified below. Should be modified only outside network idle time (NIT).</p> <p>00_B No external clock correction 01_B No external clock correction 10_B External offset correction value subtracted from calculated offset correction value 11_B External offset correction value added to calculated offset correction value</p>
ERCC	[9:8]	rw	<p>External Rate Correction Control (pExternRateControl) By writing to ERCC the external rate correction is enabled as specified below. Should be modified only outside network idle time (NIT).</p> <p>00_B No external rate correction 01_B No external rate correction 10_B External rate correction value subtracted from calculated rate correction value 11_B External rate correction value added to calculated rate correction value</p>
EOC	[18:16]	rw	<p>External Offset Correction¹⁾ (pExternOffsetCorrection) Holds the external clock offset correction value in Microticks to be applied by the internal synchronization algorithm. The value is subtracted / added from / to the calculated offset correction value. The value is applied during network idle time (NIT). May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. Valid values are 0 to 7 Microticks.</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
ERC	[26:24]	rw	External Rate Correction¹⁾ (pExternRateCorrection) Holds the external rate correction value in Microticks to be applied by the internal clock synchronization algorithm. The value is subtracted / added from / to the calculated rate correction value. The value is applied during network idle time (NIT). May be modified in "DEFAULT_CONFIG" or "CONFIG" state only. Valid values are 0 to 7 Microticks.
0	[7:2], [15:10], [23:19], [31:27]	r	Reserved Returns 0 if read; should be written with 0.

1) This bit can be updated in "DEFAULT_CONFIG" or "CONFIG" state only!

FlexRay™ Protocol Controller (E-Ray)

25.5.2.5 Communication Controller Status Registers

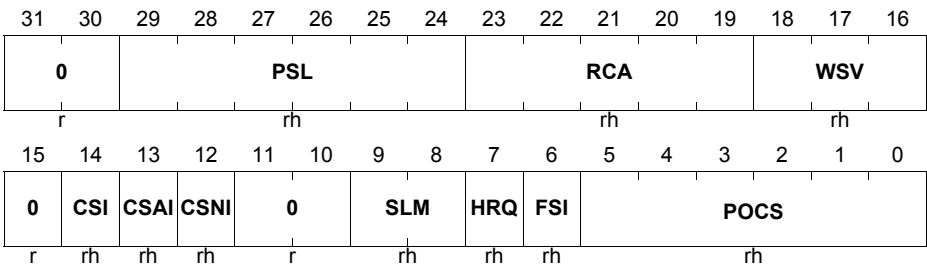
During 8/16-bit accesses to status variables coded with more than 8/16-bit, the variable might be updated by the Communication Controller between two accesses (non-atomic read accesses). The status vector may change faster than the Host can poll the status vector, depending on f_{CLC_ERAY} frequency.

Communication Controller Status Vector (CCSV)

CCSV

Communication Controller Status Vector(0100_H)

Reset Value: 0010 4000_H



FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
POCS	[5:0]	rh	<p>Protocol Operation Control Status</p> <p>Indicates the actual state of operation of the Communication Controller Protocol Operation Control</p> <p>000000_B “DEFAULT_CONFIG” state 000001_B “READY” state 000010_B “NORMAL_ACTIVE” state 000011_B “NORMAL_PASSIVE” state 000100_B “HALT” state 000101_B “MONITOR_MODE” state 000110_B ... 001110_B are reserved. 001111_B “CONFIG” state</p> <p>Indicates the actual state of operation of the POC in the wakeup path</p> <p>010000_B WAKEUP_STANDBY state 010001_B “WAKEUP_LISTEN” state 010010_B “WAKEUP_SEND” state 010011_B “WAKEUP_DETECT” state 010100_B ... 011111_B are reserved.</p> <p>Indicates the actual state of operation of the POC in the startup path</p> <p>100000_B “STARTUP_PREPARE” state 100001_B “COLDSTART_LISTEN” state 100010_B “COLDSTART_COLLISION_RESOLUTION state 100011_B “COLDSTART_CONSISTENCY_CHECK” state 100100_B “COLDSTART_GAP state 100101_B “COLDSTART_JOIN” State 100110_B “INTEGRATION_COLDSTART_CHECK” state 100111_B “INTEGRATION_LISTEN” state 101000_B “INTEGRATION_CONSISTENCY_CHECK” state 101001_B “INITIALIZE_SCHEDULE” state 101010_B “ABORT_STARTUP” state 101011_B “STARTUP_SUCCESS” state 101100_B ... 111111_B are reserved.</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
FSI	6	rh	Freeze Status Indicator (vPOC!Freeze) Indicates that the POC has entered the “HALT” state due to CHI command “FREEZE” or due to an error condition requiring an immediate POC halt. Reset by transition from “HALT” to “DEFAULT_CONFIG” state.
HRQ	7	rh	Halt Request (vPOC!CHIHaltRequest) Indicates that a request from the Host has been received to halt the POC at the end of the communication cycle. Reset by transition from “HALT” to “DEFAULT_CONFIG” state or when entering “READY” state.
SLM	[9:8]	rh	Slot Mode (vPOC!SlotMode) Indicates the actual slot mode of the POC in states READY, WAKEUP, STARTUP, NORMAL_ACTIVE, and NORMAL_PASSIVE. Default is “SINGLE”. Changes to “ALL”, depending on configuration bit SUCC1.TSM. In “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state the CHI command “ALL_SLOTS” will change the slot mode from “SINGLE” over “ALL_PENDING” to “ALL”. Set to SINGLE in all other states. 00 _B SINGLE 01 _B Reserved 10 _B ALL_PENDING 11 _B ALL
CSNI	12	rh	Coldstart Noise Indicator (vPOC!ColdstartNoise) Indicates that the cold start procedure occurred under noisy conditions. Reset by CHI command “RESET_STATUS_INDICATORS” or by transition from “HALT” to “DEFAULT_CONFIG” state or from “READY” to “STARTUP” state.
CSAI	13	rh	Coldstart Abort Indicator Coldstart aborted. Reset by CHI command “RESET_STATUS_INDICATORS” or by transition from “HALT” to “DEFAULT_CONFIG” state or from “READY” to “STARTUP” state.

FlexRay™ Protocol Controller (E-Ray)

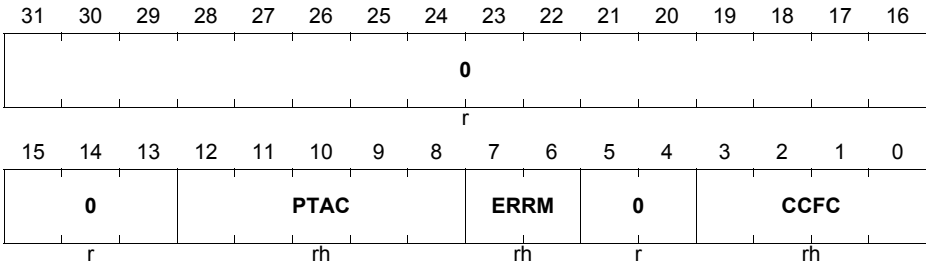
Field	Bits	Type	Description
CSI	14	rh	<p>Cold Start Inhibit (vColdStartInhibit)</p> <p>Indicates that the node is disabled from cold starting. The flag is set whenever the POC enters “READY” state due to CHI command “READY”. The flag has to be reset under control of the Host by CHI command “ALLOW_COLDSTART” (SUCC1.CMD = 1001_B).</p> <p>0_B Cold starting of node enabled 1_B Cold starting of node disabled</p>
WSV	[18:16]	rh	<p>Wakeup Status (vPOC!WakeupStatus)</p> <p>Indicates the status of the current wakeup attempt. Reset by CHI command “RESET_STATUS_INDICATORS” or by transition from “HALT” to “DEFAULT_CONFIG” state.</p> <p>000_B UNDEFINED. Wakeup not yet executed by the Communication Controller.</p> <p>001_B RECEIVED_HEADER. Set when the Communication Controller finishes wakeup due to the reception of a Frame Header without coding violation on either channel in “WAKEUP_LISTEN” state.</p> <p>010_B RECEIVED_WUP. Set when the Communication Controller finishes wakeup due to the reception of a valid wakeup pattern on the configured wakeup channel in “WAKEUP_LISTEN” state.</p> <p>011_B COLLISION_HEADER. Set when the Communication Controller stops wakeup due to a detected collision during wakeup pattern transmission by receiving a valid Header on either channel.</p> <p>100_B COLLISION_WUP. Set when the Communication Controller stops wakeup due to a detected collision during wakeup pattern transmission by receiving a valid wakeup pattern on the configured wakeup channel.</p> <p>101_B COLLISION_UNKNOWN. Set when the Communication Controller stops wakeup by leaving “WAKEUP_DETECT” state after expiration of the wakeup timer without receiving a valid wakeup pattern or a valid Frame Header.</p> <p>110_B TRANSMITTED. Set when the Communication Controller has successfully completed the transmission of the wakeup pattern.</p> <p>111_B Reserved</p>

FlexRay™ Protocol Controller (E-Ray)

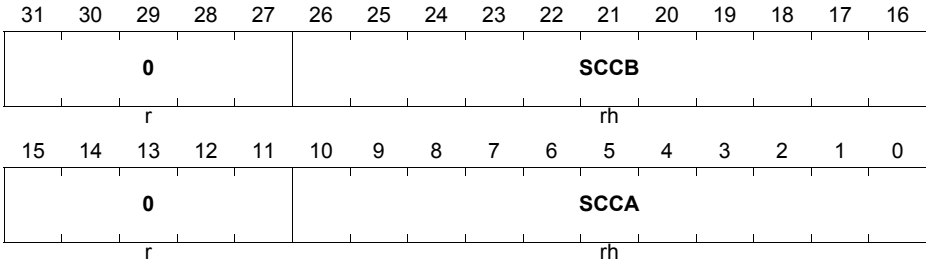
Field	Bits	Type	Description
RCA	[23:19]	rh	<p>Remaining Coldstart Attempts (vRemainingColdstartAttempts)</p> <p>Indicates the number of remaining coldstart attempts. The RUN command resets this counter to the maximum number of coldstart attempts as configured by SUCC1.CSA.</p>
PSL	[29:24]	rh	<p>POC Status Log</p> <p>Status of CCSV.POCS immediately before entering “HALT” state. Set when entering “HALT” state. Set to “HALT” when FREEZE command is applied during “HALT” state. Reset to 000000_B when leaving “HALT” state.</p>
0	[11:10], 15, [31:30]	rh	<p>Reserved</p> <p>Returns 0 if read; should be written with 0.</p>

FlexRay™ Protocol Controller (E-Ray)
Communication Controller Error Vector (CCEV)

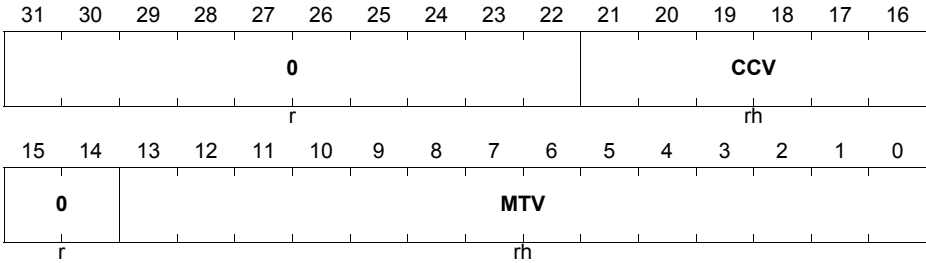
Reset by transition from “HALT” to “DEFAULT_CONFIG” state or when entering “READY” state.

CCEV
Communication Controller Error Vector (0104_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CCFC	[3:0]	rh	Clock Correction Failed Counter (vClockCorrectionFailed) The Clock Correction Failed Counter is incremented by one at the end of any odd communication cycle where either the missing offset correction error or missing rate correction error are active. The Clock Correction Failed Counter is reset to 0 at the end of an odd communication cycle if neither the offset correction failed nor the rate correction failed errors are active. The Clock Correction Failed Counter stops at 15.
ERRM	[7:6]	rh	Error Mode (vPOC!ErrorMode) Indicates the actual error mode of the POC. 00 _B “ACTIVE” (green) 01 _B “PASSIVE” (yellow) 10 _B “COMM_HALT” (red) 11 _B Reserved
PTAC	[12:8]	rh	Passive to Active Count (vAllowPassiveToActive) Indicates the number of consecutive even / odd cycle pairs that have passed with valid rate and offset correction terms, while the node is waiting to transit from “NORMAL_PASSIVE” state to “NORMAL_ACTIVE” state. The transition takes place when PTAC equals SUCC1.PTA.
0	[5:4], [31:13]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)
Slot Counter Value (SCV)
SCV
Slot Counter Value (0110_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
SCCA	[10:0]	rh	Slot Counter Channel A (vSlotCounter[A]) Current slot counter value on channel A. The value is incremented by the Communication Controller and reset at the start of a communication cycle. Valid values are 0 to 2047 (0 _H to 7FD _H).
SCCB	[26:16]	rh	Slot Counter Channel B (vSlotCounter[B]) Current slot counter value on channel B. The value is incremented by the Communication Controller and reset at the start of a communication cycle. Valid values are 0 to 2047 (0 _H to 7FD _H).
0	[15:11], [31:27]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)
Macrotick and Cycle Counter Value (MTCCV)
MTCCV
Macrotick and Cycle Counter Value (0114_H)
Reset Value: 0000 0000_H


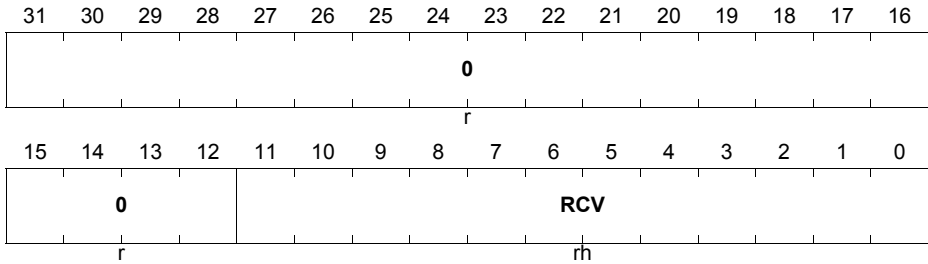
Field	Bits	Type	Description
MTV	[13:0]	rh	Macrotick Value (vMacrotick) Current Macrotick value. The value is incremented by the Communication Controller and reset at the start of a communication cycle. Valid values are 0 to 16000 (0 _H to 3E80 _H).
CCV	[21:16]	rh	Cycle Counter Value (vCycleCounter) Current cycle counter value. The value is incremented by the Communication Controller at the start of a communication cycle. Valid values are 0 to 63 (0 _H to 3F _H).
0	[15:14], [31:22]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Rate Correction Value (RCV)

RCV

Rate Correction Value (0118_H) Reset Value: 0000 0000_H



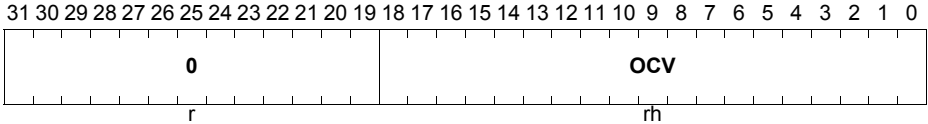
Field	Bits	Type	Description
RCV	[11:0]	rh	Rate Correction Value (vRateCorrection) Rate correction value (two's complement). Calculated internal rate correction value before limitation. If the RCV value exceeds the limits defined by GTUC10.MRC, flag SFS.RCLR is set to 1.
0	[31:12]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Offset Correction Value (OCV)

OCV

Offset Correction Value (011C_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
OCV	[18:0]	rh	Offset Correction Value (vOffsetCorrection) Offset correction value (two's complement). Calculated internal offset correction value before limitation. If the OCV value exceeds the limits defined by GTUC10.MOC flag SFS.OCLR is set to 1.
0	[31:19]	r	Reserved Returns 0 if read; should be written with 0.

Note: The external rate / offset correction value is added to the limited rate / offset correction value.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
VSBO	[15:12]	rh	Valid SYNC Frames Channel B, odd communication cycle Holds the number of valid SYNC Frames received on channel B in the odd communication cycle. If transmission of SYNC Frames is enabled by SUCC1.TXSY the value is incremented by one. The value is updated during the network idle time (NIT) of each odd communication cycle. This bit field is only valid if the channel B is assigned to the Communication Controller by SUCC1.CCHB.
MOCS	16	rh	Missing Offset Correction Signal The Missing Offset Correction flag signals to the Host, that no offset correction calculation can be performed because no SYNC Frames were received. The flag is updated by the Communication Controller at start of offset correction phase. 0 _B Offset correction signal valid 1 _B Missing offset correction signal
OCLR	17	rh	Offset Correction Limit Reached The Offset Correction Limit Reached flag signals to the Host, that the offset correction value has exceeded its limit as defined by GTUC10.MOC. The flag is updated by the Communication Controller at start of offset correction phase. 0 _B Offset correction below limit 1 _B Offset correction limit reached
MRCS	18	rh	Missing Rate Correction Signal The Missing Rate Correction Flag signals to the Host, that no rate correction calculation can be performed because no pairs of even / odd SYNC Frames were received. The flag is updated by the Communication Controller at start of offset correction phase. 0 _B Rate correction signal valid 1 _B Missing rate correction signal
RCLR	19	rh	Rate Correction Limit Reached The Rate Correction Limit Reached flag signals to the Host, that the rate correction value has exceeded its limit.as defined by GTUC10.MRC. The flag is updated by the Communication Controller at start of offset correction phase. 0 _B Rate correction below limit 1 _B Rate correction limit reached
0	[31:20]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Symbol Window and network idle time (NIT) Status (SWNIT)

Symbol window related status information. Updated by the Communication Controller at the end of the symbol window for each channel. During startup the status data is not updated.

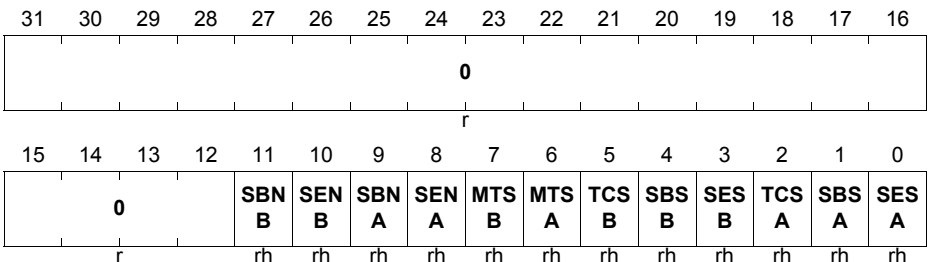
Note: MTS_A and MTS_B may be changed outside “DEFAULT_CONFIG” or “CONFIG” state when the write to SUC Configuration Register 1 (SUCC1) register is directly preceded by the unlock sequence as described in “Lock Register (LCK)” on Page 25-30. This may be combined with CHI command SEND_MTS. If both bits MTS_A and MTS_B are set to 1 an MTS symbol will be transmitted on both channels when requested by writing SUCC1.CMD = 1000_B

SWNIT

Symbol Window and Network Idle Time Status

(0124_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SESA	0	rh	Syntax Error in Symbol Window Channel A (vSS!SyntaxErrorA) 0 _B No syntax error detected 1 _B Syntax error during symbol window detected on channel A
SBSA	1	rh	Slot Boundary Violation in Symbol Window Channel A (vSS!BViolationA) 0 _B No slot boundary violation detected 1 _B Slot boundary violation during symbol window detected on channel A

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TCSA	2	rh	Transmission Conflict in Symbol Window Channel A (vSSI!TxConflictA) 0 _B No transmission conflict detected 1 _B Transmission conflict in symbol window detected on channel A
SESB	3	rh	Syntax Error in Symbol Window Channel B (vSSI!SyntaxErrorB) 0 _B No syntax error detected 1 _B Syntax error during symbol window detected on channel B
SBSB	4	rh	Slot Boundary Violation in Symbol Window Channel B (vSSI!BViolationB) 0 _B No slot boundary violation detected 1 _B Slot boundary violation during symbol window detected on channel B
TCSB	5	rh	Transmission Conflict in Symbol Window Channel B (vSSI!TxConflictB) 0 _B No transmission conflict detected 1 _B Transmission conflict in symbol window detected on channel B
MTSA	6	rh	MTS Received on Channel A (vSSI!ValidMTSA) ¹⁾ Media Access Test symbol received on channel A during the proceeding symbol window. Updated by the Communication Controller for each channel at the end of the symbol window. When this bit is set to 1, also interrupt flag SIR.MTSA is set to 1. 0 _B No MTS symbol received on channel A 1 _B MTS symbol received on channel A
MTSB	7	rh	MTS Received on Channel B (vSSI!ValidMTSB) ¹⁾ Media Access Test symbol received on channel B during the proceeding symbol window. Updated by the Communication Controller for each channel at the end of the symbol window. When this bit is set to 1, also interrupt flag SIR.MTSB is set to 1. 0 _B No MTS symbol received on channel B 1 _B MTS symbol received on channel B

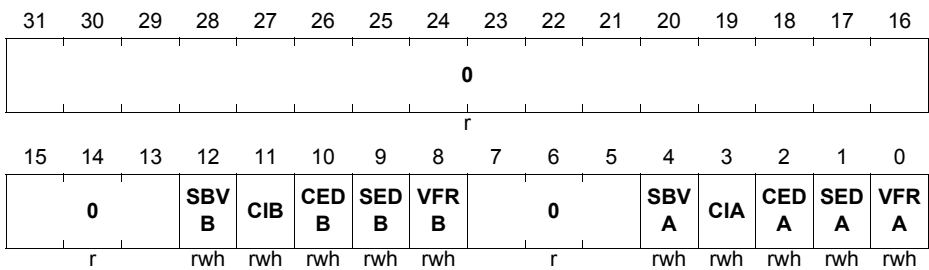
FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
SENA	8	rh	Syntax Error during network idle time (NIT) Channel A (vSSI!SyntaxErrorA) Updated by the Communication Controller channel A at the end of the NIT. 0 _B No syntax error detected 1 _B Syntax error during network idle time (NIT) detected on channel A
SBNA	9	rh	Slot Boundary Violation during network idle time (NIT) Channel A (vSSI!BViolationA) Updated by the Communication Controller channel A at the end of the NIT. 0 _B No slot boundary violation detected 1 _B Slot boundary violation during network idle time (NIT) detected on channel A
SENB	10	rh	Syntax Error during network idle time (NIT) Channel B (vSSI!SyntaxErrorB) Updated by the Communication Controller channel B at the end of the NIT. 0 _B No syntax error detected 1 _B Syntax error during network idle time (NIT) detected on channel B
SBNB	11	rh	Slot Boundary Violation during network idle time (NIT) Channel B (vSSI!BViolationB) Updated by the Communication Controller channel B at the end of the NIT. 0 _B No slot boundary violation detected 1 _B Slot boundary violation during network idle time (NIT) detected on channel B
0	[31:12]	r	Reserved Returns 0 if read; should be written with 0.

- 1) MTSA and MTSB may also be changed outside "DEFAULT_CONFIG" or "CONFIG" state when the write to SUCC1 register is directly preceded by the unlock sequence as described in "Lock Register (LCK)". This may be combined with CHI command SEND_MTS. If both bits MTSA and MTSB are set to 1 an MTS symbol will be transmitted on both channels when requested by writing SUCC1.CMD = 1000_g.

FlexRay™ Protocol Controller (E-Ray)
Aggregated Channel Status (ACS)

The aggregated channel status provides the Host with an accrued status of channel activity for all communication slots regardless of whether they are assigned for transmission or subscribed for reception. The aggregated channel status also includes status data from the symbol window and the network idle time. The status data is updated (set) after each slot and aggregated until it is reset by the Host. During startup the status data is not updated. A flag is cleared by writing a 1 to the corresponding bit position. Writing a 0 has no effect on the flag. An application reset will also clear the register.

ACS
Aggregated Channel Status (0128_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
VFRA	0	rwh	Valid Frame Received on Channel A (vSS!ValidFrameA) One or more valid Frames were received on channel A in any static or dynamic slot during the observation period. 0 _B No valid Frame received 1 _B Valid Frame(s) received on channel A
SEDA	1	rwh	Syntax Error Detected on Channel A (vSS!SyntaxErrorA) One or more syntax errors in static or dynamic slots, symbol window, and network idle time (NIT) were observed on channel A. 0 _B No syntax error observed 1 _B Syntax error(s) observed on channel A
CEDA	2	rwh	Content Error Detected on Channel A (vSS!ContentErrorA) One or more Frames with a content error were received on channel A in any static or dynamic slot during the observation period. 0 _B No Frame with content error received 1 _B Frame(s) with content error received on channel A

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
CIA	3	rwh	<p>Communication Indicator Channel A</p> <p>One or more valid Frames were received on channel A in slots that also contained any additional communication during the observation period, i.e. one or more slots received a valid Frame AND had any combination of either syntax error OR content error OR slot boundary violation.</p> <p>0_B No valid Frame(s) received in slots containing any additional communication</p> <p>1_B Valid Frame(s) received on channel A in slots containing any additional communication</p>
SBVA	4	rwh	<p>Slot Boundary Violation on Channel A (vSSI!BViolationA)</p> <p>One or more slot boundary violations were observed on channel A at any time during the observation period (static or dynamic slots, symbol window, and network idle time NIT).</p> <p>0_B No slot boundary violation observed</p> <p>1_B Slot boundary violation(s) observed on channel A</p>
VFRB	8	rwh	<p>Valid Frame Received on Channel B (vSSI!ValidFrameB)</p> <p>One or more valid Frames were received on channel B in any static or dynamic slot during the observation period.</p> <p>0_B No valid Frame received</p> <p>1_B Valid Frame(s) received on channel B</p>
SEDB	9	rwh	<p>Syntax Error Detected on Channel B (vSSI!SyntaxErrorB)</p> <p>One or more syntax errors in static or dynamic slots, symbol window, and network idle time (NIT) were observed on channel B.</p> <p>0_B No syntax error observed</p> <p>1_B Syntax error(s) observed on channel B</p>
CEDB	10	rwh	<p>Content Error Detected on Channel B (vSSI!ContentErrorB)</p> <p>One or more Frames with a content error were received on channel B in any static or dynamic slot during the observation period.</p> <p>0_B No Frame with content error received</p> <p>1_B Frame(s) with content error received on channel B</p>

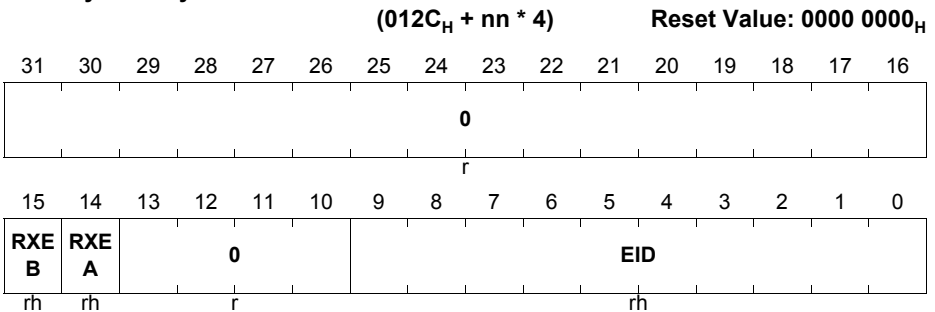
FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
CIB	11	rwh	Communication Indicator Channel B One or more valid Frames were received on channel B in slots that also contained any additional communication during the observation period, i.e. one or more slots received a valid Frame AND had any combination of either syntax error OR content error OR slot boundary violation. 0_B No valid Frame(s) received in slots containing any additional communication 1_B Valid Frame(s) received on channel B in slots containing any additional communication
SBVB	12	rwh	Slot Boundary Violation on Channel B (vSS!BViolationB) One or more slot boundary violations were observed on channel B at any time during the observation period (static or dynamic slots, symbol window, and network idle time NIT). 0_B No slot boundary violation observed 1_B Slot boundary violation(s) observed on channel B
0	[7:5], [31:13]	r	Reserved Returns 0 if read; should be written with 0.

Note: The set condition of flags CIA and CIB is also fulfilled if there is only one single Frame in the slot and the slot boundary at the end of the slot is reached during the Frames channel idle recognition phase. When one of the flags SEDB, CEDB, CIB, SBVB changes from 0 to 1, service request flag EIR.EDB is set to 1. When one of the flags SEDA, CEDA, CIA, SBVA changes from 0 to 1, service request flag EIR.EDA is set to 1.

FlexRay™ Protocol Controller (E-Ray)
Even Sync ID [01...15] (ESIDnn)

Registers Even Sync ID nn (ESIDnn, nn=01-15) hold the Frame IDs of the SYNC Frames received in **even** communication cycles, sorted in ascending order, with register ESID01 holding the lowest received SYNC Frame ID. If the node itself transmits a SYNC Frame in an even communication cycle, register ESID01 holds the respective SYNC Frame ID as configured in Message Buffer 0 and the flags RXEA, RXEB are set. The value is updated during the network idle time (NIT) of each even communication cycle.

ESIDnn (nn = 01-15)
Even Sync ID Symbol Window nn


Field	Bits	Type	Description
EID	[9:0]	rh	Even Sync ID (vsSyncIDListA,B even) SYNC Frame ID even communication cycle.
RXEA	14	rh	Received/Configured Even Sync ID on Channel A Signals that a SYNC Frame corresponding to the stored even sync ID was received on channel A or that the node is configured to be a sync node with key slot = EID (ESID1 only). 0 _B SYNC Frame not received on channel A / node configured to transmit SYNC Frames 1 _B SYNC Frame received on channel A / node not configured to transmit SYNC Frames
RXEB	15	rh	Received/Configured Even Sync ID on Channel B Signals that a SYNC Frame corresponding to the stored even sync ID was received on channel B or that the node is configured to be a sync node with key slot = EID (ESID1 only). 0 _B SYNC Frame not received on channel B / node configured to transmit SYNC Frames 1 _B SYNC Frame received on channel B / node not configured to transmit SYNC Frames

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
0	[13:10], [31:16]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
0	[13:10], [31:16]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Network Management Vector [1...3] (NMVx)

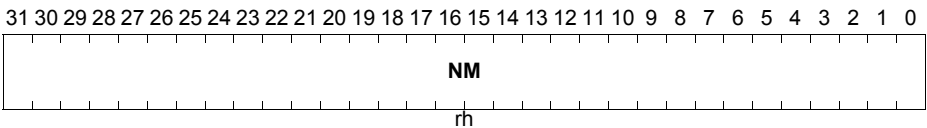
The three Network Management Vectors n (NMVx, x=1-3) registers hold the accrued Network Management (NM) vector (configurable 0 to 12 byte). The accrued Network Management (NM) vector is generated by the Communication Controller by bit-wise ORing each Network Management (NM) vector received (valid static Frames with PPI = 1) on each channel (see **“Network Management” on Page 25-213**). The Communication Controller updates the Network Management (NM) vector at the end of each communication cycle as long as the Communication Controller is either in “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state. NMVx-bytes exceeding the configured Network Management (NM) vector length are not valid.

NMVx (x = 1-3)

Network Management Vector x

$$(01AC_H + x * 4)$$

Reset Value: 0000 0000_H



Field	Bits	Type	Description
NM	[31:0]	rh	Network Management Vector

Table 25-5 below shows the assignment of the received payload's data byte to the Network Management vector.

Table 25-5 Assignment of Data Byte to Network Management Vector

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word	1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0										
NM1	Data3			Data2			Data1			Data0																						
NM2	Data7			Data6			Data5			Data4																						
NM3	Data11			Data10			Data9			Data8																						

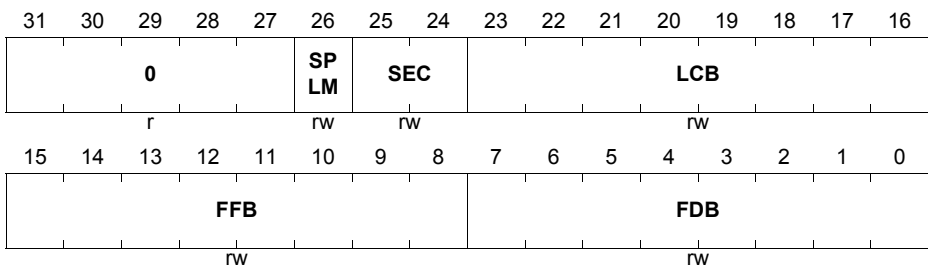
25.5.2.6 Message Buffer Control Registers

Message RAM Configuration (MRC)

The Message RAM Configuration register defines the number of Message Buffers assigned to the static segment, dynamic segment, and FIFO. The register can be written during “DEFAULT_CONFIG” or “CONFIG” state only.

MRC

Message RAM Configuration (0300_H) **Reset Value: 0180 0000_H**



Field	Bits	Type	Description
FDB	[7:0]	rw	First Dynamic Buffer May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 00 _H No group of Message Buffers exclusively for the static segment configured 01 _H ...7F _H Message Buffers 0 to FDB-1 reserved for static segment 80 _H ...FF _H No dynamic Message Buffers configured
FFB	[15:8]	rw	First Buffer of FIFO May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 00 _H ...7E _H Message Buffers from FFB to LCB assigned to the FIFO 7F _H All Message Buffers assigned to the FIFO 80 _H ...FF _H No Message Buffers assigned to the FIFO

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
LCB	[23:16]	rw	<p>Last Configured Buffer May be only modified in “DEFAULT_CONFIG” or “CONFIG” state.</p> <p>01_H ...7F_H Number of Message Buffers is LCB + 1</p> <p>80_H ...FF_H No Message Buffer configured</p>
SEC	[25:24]	rw	<p>Secure Buffers Not evaluated when the Communication Controller is in “DEFAULT_CONFIG” or “CONFIG” state. For temporary unlocking see “Host Handling of Errors” on Page 25-248.</p> <p>00_B Reconfiguration of Message Buffers enabled with numbers < FFB enabled.</p> <p><i>Note: In nodes configured for SYNC Frame transmission or for single slot mode operation Message Buffer 0 (and if SPLM = 1, also Message Buffer 1) Reconfiguration of all Message Buffers is always locked</i></p> <p>01_B Reconfiguration of Message Buffers with numbers < FDB and with numbers ≥ FFB locked and transmission of Message Buffers for static segment with numbers ≥ FDB disabled</p> <p>10_B Reconfiguration of all Message Buffers locked</p> <p>11_B Reconfiguration of all Message Buffers locked and transmission of Message Buffers for static segment with numbers ≥ FDB disabled</p>
SPLM	26	rw	<p>SYNC Frame Payload Multiplex This bit is only evaluated if the node is configured as sync node (SUCC1.TXSY = 1) or for single slot mode operation (SUCC1.TSM = 1). When this bit is set to 1 Message Buffers 0 and 1 are dedicated for SYNC Frame transmission with different payload data on channel A and B. When this bit is reset to 0, SYNC Frames are transmitted from Message Buffer 0 with the same payload data on both channels. Note that the channel filter configuration for Message Buffer 0 resp. Message Buffer 1 has to be chosen accordingly.</p> <p>0_B Only Message Buffer 0 locked against reconfiguration</p> <p>1_B Both Message Buffers 0 and 1 are locked against reconfiguration</p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
0	[31:27]	r	Reserved Returns 0 if read; should be written with 0.

Note: In case the node is configured as sync node (SUCC1.TXSY = 1) or for single slot mode operation (SUCC1.TSM = 1), Message Buffer 0 resp. 1 is reserved for SYNC Frames or single slot Frames and have to be configured with the node-specific key slot ID. In case the node is neither configured as sync node nor for single slot operation Message Buffer 0 resp. 1 is treated like all other Message Buffers.

Table 25-6 Usage of the three Message Buffer Pointer

Message Buffer 0	↓ Static Buffers		
Message Buffer 1			
	↓ Static + Dynamic Buffers	← FDB	
...			FIFO configured: FFB > FDB
	↓ FIFO	← FFB	No FIFO configured: FFB ≥ 128
Message Buffer N-1			LCB ≥ FDB, LCB ≥ FFB
Message Buffer N		← LCB	

The programmer has to ensure that the configuration defined by FDB, FFB, and LCB is valid. **The Communication Controller does not check for erroneous configurations!**

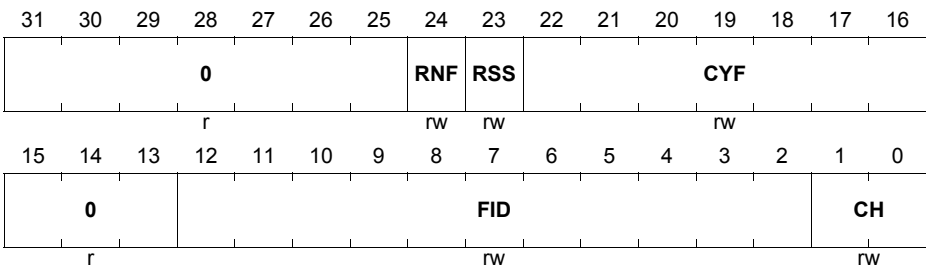
*Note: The maximum number of Header Sections is 128. This means a maximum of 128 Message Buffer can be configured. The maximum length of a Data Section is 254 byte. The length of the Data Section may be configured differently for each Message Buffer. For details see **“Message RAM” on Page 25-239**.*

In case two or more Message Buffers are assigned to slot 1 by use of cycle filtering, all of them must be located either in the “Static Buffers” or at the beginning of the “Static + Dynamic Buffers” section.

The payload length configured and the length of the Data Section need to be configured identically for all Message Buffers belonging to the FIFO via WRHS2.PLC and WRHS3.DP. When the Communication Controller is not in “DEFAULT_CONFIG” or “CONFIG” state reconfiguration of Message Buffers belonging to the FIFO is locked.

FlexRay™ Protocol Controller (E-Ray)
FIFO Rejection Filter (FRF)

The FIFO Rejection Filter defines a user specified sequence of bits to which channel, Frame ID, and cycle count of the incoming Frames are compared. Together with the FIFO Rejection Filter Mask this register determines whether a message is rejected by the FIFO. The FRF register can be written during “DEFAULT_CONFIG” or “CONFIG” state only.

FRF
FIFO Rejection Filter (0304_H) **Reset Value: 0180 0000_H**


Field	Bits	Type	Description
CH	[1:0]	rw	Channel Filter May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 00 _B receive on both channels ¹⁾ 01 _B receive only on channel B 10 _B receive only on channel A 11 _B no reception
FID	[12:2]	rw	Frame ID Filter Determines the Frame ID to be rejected by the FIFO. With the additional configuration of register FRFM, the corresponding Frame ID filter bits are ignored, which results in further rejected Frame IDs. When FRFM.MFID is zero, a Frame ID filter value of zero means that no Frame ID is rejected. 000 _H ... 7FF _H Frame ID filter values

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
CYF	[22:16]	rw	Cycle Counter Filter The 7-bit cycle counter filter determines the cycle set to which Frame ID and channel rejection filter are applied. In cycles not belonging to the cycle set specified by CYF, all Frames are rejected. For details about the configuration of the cycle counter filter see “Cycle Counter Filtering” on Page 25-215 . May be modified in “DEFAULT_CONFIG” or “CONFIG” state only.
RSS	23	rw	Reject in Static Segment If this bit is set, the FIFO is used only be used in dynamic segment. May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 0 _B FIFO also used in static segment 1 _B Reject messages for static segment
RNF	24	rw	Reject NULL Frames If this bit is set, received NULL Frames are not stored in the FIFO. May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 0 _B NULL Frames are stored in the FIFO 1 _B Reject all NULL Frames
0	[15:13], [31:25]	r	Reserved Returns 0 if read; should be written with 0.

- 1) If reception on both channels is configured, also in static segment always both Frames (from channel A and B) are stored in the FIFO, even if they are identical.

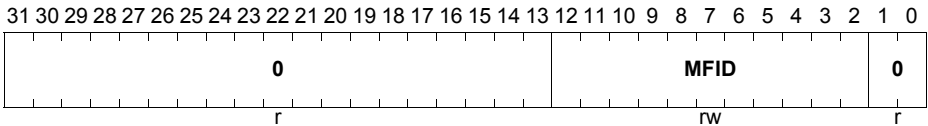
FlexRay™ Protocol Controller (E-Ray)

FIFO Rejection Filter Mask (FRFM)

The FIFO Rejection Filter Mask specifies which of the corresponding Frame ID filter bits are relevant for rejection filtering. If a bit is set, it indicates that the corresponding bit in the FRF register will not be considered for rejection filtering. The FRFM register can be written during “DEFAULT_CONFIG” or “CONFIG” state only.

FRFM

FIFO Rejection Filter Mask (0308_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
MFID	[12:2]	rw	Mask Frame ID Filter May be modified in “DEFAULT_CONFIG” or “CONFIG” state only. 0 _B Corresponding Frame ID filter bit is used for rejection filtering. 1 _B Ignore corresponding Frame ID filter bit.
0	[1:0], [31:13]	r	Reserved Returns 0 if read; should be written with 0.

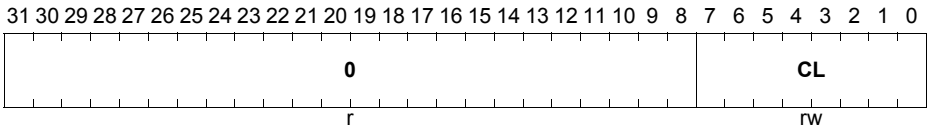
FlexRay™ Protocol Controller (E-Ray)

FIFO Critical Level (FCL)

The Communication Controller accepts modifications of the register in “DEFAULT_CONFIG” or “CONFIG” state only.

FCL

FIFO Critical Level (030C_H) Reset Value: 0000 0080_H



Field	Bits	Type	Description
CL	[7:0]	rw	<p>Critical Level</p> <p>When the receive FIFO fill level FSR.RFFL is equal or greater than the critical level configured by CL, the receive FIFO critical level flag FSR.RFCL is set. If CL is programmed to values > 128, bit FSR.RFCL is never set. When FSR.RFCL changes from 0 to 1 bit SIR.RFCL is set to 1, and if enabled, a service request is generated.</p>
0	[31:8]	r	<p>Reserved</p> <p>Returns 0 if read; should be written with 0.</p>

FlexRay™ Protocol Controller (E-Ray)

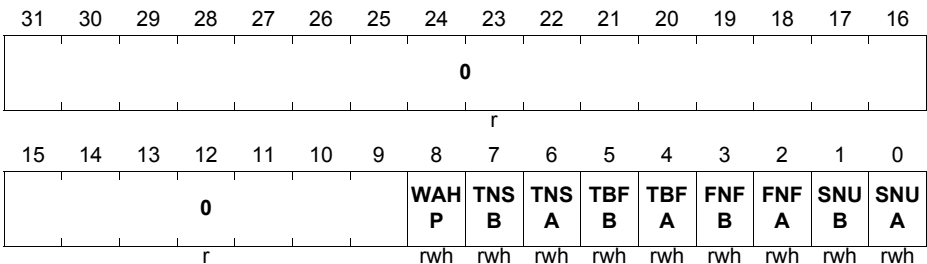
Field	Bits	Type	Description
FMBD	5	rwh	Faulty Message Buffer Detected 0 _B No faulty Message Buffer 1 _B Message Buffer referenced by MHDS.FMB holds faulty data due to a ECC error
MFMB	6	rwh	Multiple Faulty Message Buffers detected 0 _B No additional faulty Message Buffer 1 _B Another faulty Message Buffer was detected while flag MHDS.FMBD is set
GRAM	7	rh	Clear all internal RAM's Signals that execution of the CHI command CLEAR_RAMs is ongoing (all bits of all internal RAM blocks are written to 0). The bit is set by CHI command CLEAR_RAMs. 0 _B No execution of the CHI command CLEAR_RAMs 1 _B Execution of the CHI command CLEAR_RAMs ongoing
FMB	[14:8]	rh	Faulty Message Buffer ECC error occurred when reading from the Message Buffer or when transferring data from Input Buffer or Transient Buffer A or Transient Buffer B to the Message Buffer referenced by MHDS.FMB. Value only valid when one of the flags MHDS.EIBF, MHDS.EMR, MHDS.ETBF1, MHDS.ETBF2, and flag MHDS.FMBD is set. Updated only after the Host has reset flag MHDS.FMBD.
MBT	[22:16]	rh	Message Buffer Transmitted Number of last successfully transmitted Message Buffer. If the Message Buffer is configured for single-shot mode, the respective TXR flag in the Transmission Request Registers TXRQ1 to TXRQ4 was reset. MBT is reset when the Communication Controller leaves "CONFIG" state or enters "STARTUP" state.
MBU	[30:24]	rh	Message Buffer Updated Number of Message Buffer that was updated last. For this Message Buffer the respective NDn (n = 0-31) to NDn (n = 96-127) and / or MBCn (n = 0-31) to MBCn (n = 96-127) flag in the New Data Registers NDAT1 to NDAT4 and the Message Buffer Status Changed MBSC1 to MBSC4 registers are also set. MBU is reset when the Communication Controller leaves "CONFIG" state or enters "STARTUP" state.
0	15, 23, 31	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
RFO	2	rh	Receive FIFO Overrun The flag is set by the Communication Controller when a receive FIFO overrun is detected. When a receive FIFO overrun occurs, the oldest message is overwritten with the actual received message. In addition, service request flag EIR.RFO is set. The flag is cleared by the next FIFO read access issued by the Host. 0 _B No receive FIFO overrun detected 1 _B A receive FIFO overrun has been detected
RFFL	[15:8]	rh	Receive FIFO Fill Level Number of FIFO buffers filled up with new data not yet read by the Host. Maximum value is 128.
0	[7:3], [31:16]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)
Message Handler Constraints Flags (MHDF)

Some constraints exist for the Message Handler regarding f_{CLC_ERAY} frequency, Message RAM configuration, and FlexRay™ bus traffic. To simplify software development, constraints violations are reported by setting flags in the MHDF. The register is reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state. A flag is cleared by setting the corresponding bit position. Clearing has no effect on the flag. If any flag in MHDFL is set, interrupt flag EIR.MHF is set.

MHDF
Message Handler Constraints Flags (031C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SNUA	0	rwh	Status Not Updated Channel A This flag is set by the Communication Controller when the Message Handler, due to overload condition, was not able to update a Message Buffer's status MBS with respect to channel A. 0 _B No overload condition occurred when updating MBS for channel A 1 _B MBS for channel A not updated
SNUB	1	rwh	Status Not Updated Channel B This flag is set by the Communication Controller when the Message Handler, due to overload condition, was not able to update a Message Buffer's status MBS with respect to channel B. 0 _B No overload condition occurred when updating MBS for channel B 1 _B MBS for channel B not updated

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
FNFA	2	rwh	Find Sequence Not Finished Channel A This flag is set by the Communication Controller when the Message Handler, due to overload condition, was not able to finish a find sequence (scan of Message RAM for matching Message Buffer) with respect to channel A. 0 _B No find sequence not finished for channel A 1 _B Find sequence not finished for channel A
FNFB	3	rwh	Find Sequence Not Finished Channel B This flag is set by the Communication Controller when the Message Handler, due to overload condition, was not able to finish a find sequence (scan of Message RAM for matching Message Buffer) with respect to channel B. 0 _B No find sequence not finished for channel B 1 _B Find sequence not finished for channel B
TBFA	4	rwh	Transient Buffer Access Failure A This flag is set by the Communication Controller when a read or write access to Transient Buffer A requested by PRT A could not complete within the available time. 0 _B No TBF A access failure 1 _B TBF A access failure
TBFB	5	rwh	Transient Buffer Access Failure B This flag is set by the Communication Controller when a read or write access to Transient Buffer B requested by PRT B could not complete within the available time. 0 _B No Transient Buffer B access failure 1 _B Transient Buffer B access failure
TNSA	6	rwh	Transmission Not Started Channel A This flag is set by the CC when the Message Handler was not ready to start a scheduled transmission on channel A at the action point of the configured slot. 0 _B No transmission not started on channel A 1 _B Transmission not started on channel A
TNSB	7	rwh	Transmission Not Started Channel B This flag is set by the CC when the Message Handler was not ready to start a scheduled transmission on channel B at the action point of the configured slot. 0 _B No transmission not started on channel B 1 _B Transmission not started on channel B

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
WAHP	8	rwh	<p>Write Attempt to Header Partition</p> <p>Outside “DEFAULT_CONFIG” and “CONFIG” state this flag is set by the Communication Controller when the message handler tries to write message data into the Header Partition of the Message RAM due to faulty configuration of a Message Buffer. The write attempt is not executed, to protect the Header Partition from unintended write accesses.</p> <p>0_B No write attempt to Header Partition 1_B Write attempt to Header Partition</p>
0	[31:9]	r	<p>Reserved</p> <p>Returns 0 if read; should be written with 0.</p>

FlexRay™ Protocol Controller (E-Ray)
Transmission Request 1 (TXRQ1)

This register reflect the state of the TXR flags of the configured Message Buffers 0 to 31. The flags are evaluated for transmit buffers only. If the number of configured Message Buffers is less than 31, the remaining TXRn flags have no meaning and are read as 0.

TXRQ1
Transmission Request Register 1 (0320_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXR 31	TXR 30	TXR 29	TXR 28	TXR 27	TXR 26	TXR 25	TXR 24	TXR 23	TXR 22	TXR 21	TXR 20	TXR 19	TXR 18	TXR 17	TXR 16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXR 15	TXR 14	TXR 13	TXR 12	TXR 11	TXR 10	TXR 9	TXR 8	TXR 7	TXR 6	TXR 5	TXR 4	TXR 3	TXR 2	TXR 1	TXR 0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TXRn (n = 0-31)	n	rh	Transmission Request n (n = 0-31) If the flag is set, the respective Message Buffer 0 to 31 is ready for transmission respectively transmission of this Message Buffer is in progress. In single-shot mode the flags are reset after transmission has completed.

FlexRay™ Protocol Controller (E-Ray)
Transmission Request Register 4 (TXRQ4)

This register reflect the state of the TXR flags of the configured Message Buffers 96 to 127. The flags are evaluated for transmit buffers only. If the number of configured Message Buffers is less than 127, the remaining TXRn flags have no meaning and are read as 0.

TXRQ4
Transmission Request Register 4 (032C_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXR 127	TXR 126	TXR 125	TXR 124	TXR 123	TXR 122	TXR 121	TXR 120	TXR 119	TXR 118	TXR 117	TXR 116	TXR 115	TXR 114	TXR 113	TXR 112
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXR 111	TXR 110	TXR 109	TXR 108	TXR 107	TXR 106	TXR 105	TXR 104	TXR 103	TXR 102	TXR 101	TXR 100	TXR 99	TXR 98	TXR 97	TXR 96
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TXRn (n = 96-127)	n - 96	rh	Transmission Request n (n = 96-127) If the flag is set, the respective Message Buffer 96 to 127 is ready for transmission respectively transmission of this Message Buffer is in progress. In single-shot mode the flags are reset after transmission has completed.

FlexRay™ Protocol Controller (E-Ray)
New Data Register 1 (NDAT1)

This register reflect the state of the ND flags of all configured Message Buffers 0 to 31. ND flags assigned to transmit buffers are meaningless. If the number of configured Message Buffers is less than 31, the remaining NDn flags have no meaning. The registers are reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state.

NDAT1
New Data Register 1
(0330_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND 31	ND 30	ND 29	ND 28	ND 27	ND 26	ND 25	ND 24	ND 23	ND 22	ND 21	ND 20	ND 19	ND 18	ND 17	ND 16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND 15	ND 14	ND 13	ND 12	ND 11	ND 10	ND9	ND8	ND7	ND6	ND5	ND4	ND3	ND2	ND1	ND0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NDn (n = 0-31)	n	rh	New Data n (n = 0-31) The flags are set when a valid received Data Frame matches the Message Buffer’s filter configuration, independent of the payload length received or the payload length configured for that Message Buffer. The flags are not set after reception of NULL Frames except for Message Buffers belonging to the receive FIFO. An ND flag is reset when the Header Section of the corresponding Message Buffer is reconfigured or when the Data Section has been transferred to the Output Buffer.

FlexRay™ Protocol Controller (E-Ray)
New Data Register 2 (NDAT2)

This register reflect the state of the ND flags of all configured Message Buffers 32 to 63. ND flags assigned to transmit buffers are meaningless. If the number of configured Message Buffers is less than 63, the remaining NDn flags have no meaning. The registers are reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state.

NDAT2
New Data Register 2
(0334_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND 63	ND 62	ND 61	ND 60	ND 59	ND 58	ND 57	ND 56	ND 55	ND 54	ND 53	ND 52	ND 51	ND 50	ND 49	ND 48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND 47	ND 46	ND 45	ND 44	ND 43	ND 42	ND 41	ND 40	ND 39	ND 38	ND 37	ND 36	ND 35	ND 34	ND 33	ND 32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NDn (n = 32-63)	n - 32	rh	New Data n (n = 32-63) The flags are set when a valid received Data Frame matches the Message Buffer’s filter configuration, independent of the payload length received or the payload length configured for that Message Buffer. The flags are not set after reception of NULL Frames except for Message Buffers belonging to the receive FIFO. An ND flag is reset when the Header Section of the corresponding Message Buffer is reconfigured or when the Data Section has been transferred to the Output Buffer.

FlexRay™ Protocol Controller (E-Ray)
New Data Register 3 (NDAT3)

This register reflect the state of the ND flags of all configured Message Buffers 64 to 95. ND flags assigned to transmit buffers are meaningless. If the number of configured Message Buffers is less than 95, the remaining NDn flags have no meaning. The registers are reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state.

NDAT3
New Data Register 3
(0338_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND 95	ND 94	ND 93	ND 92	ND 91	ND 90	ND 89	ND 88	ND 87	ND 86	ND 85	ND 84	ND 83	ND 82	ND 81	ND 80
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND 79	ND 78	ND 77	ND 76	ND 75	ND 74	ND 73	ND 72	ND 71	ND 70	ND 69	ND 68	ND 67	ND 66	ND 65	ND 64
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NDn (n = 64-95)	n - 64	rh	New Data n (n = 64-95) The flags are set when a valid received Data Frame matches the Message Buffer’s filter configuration, independent of the payload length received or the payload length configured for that Message Buffer. The flags are not set after reception of NULL Frames except for Message Buffers belonging to the receive FIFO. An ND flag is reset when the Header Section of the corresponding Message Buffer is reconfigured or when the Data Section has been transferred to the Output Buffer.

FlexRay™ Protocol Controller (E-Ray)
New Data Register 4 (NDAT4)

This register reflect the state of the ND flags of all configured Message Buffers 96 to 127. ND flags assigned to transmit buffers are meaningless. If the number of configured Message Buffers is less than 127, the remaining NDn flags have no meaning. The registers are reset when the Communication Controller leaves “CONFIG” state or enters “STARTUP” state.

NDAT4
New Data Register 4
(033C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ND 127	ND 126	ND 125	ND 124	ND 123	ND 122	ND 121	ND 120	ND 119	ND 118	ND 117	ND 116	ND 115	ND 114	ND 113	ND 112
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ND 111	ND 110	ND 109	ND 108	ND 107	ND 106	ND 105	ND 104	ND 103	ND 102	ND 101	ND 100	ND 99	ND 98	ND 97	ND 96
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NDn (n = 96-127)	n - 96	rh	New Data n (n = 96-127) The flags are set when a valid received Data Frame matches the Message Buffer’s filter configuration, independent of the payload length received or the payload length configured for that Message Buffer. The flags are not set after reception of NULL Frames except for Message Buffers belonging to the receive FIFO. An ND flag is reset when the Header Section of the corresponding Message Buffer is reconfigured or when the Data Section has been transferred to the Output Buffer.

FlexRay™ Protocol Controller (E-Ray)

Message Buffer Status Changed 1 (MBSC1)

This register reflect the state of the MBC flags of all configured Message Buffers. If the number of configured Message Buffers is less than 31, the remaining MBCn flags have no meaning. The register is reset when the communication controller leaves “CONFIG” state or enters “STARTUP” state.

MBSC1

Message Buffer Status Changed 1 (0340_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MBC 31	MBC 30	MBC 29	MBC 28	MBC 27	MBC 26	MBC 25	MBC 24	MBC 23	MBC 22	MBC 21	MBC 20	MBC 19	MBC 18	MBC 17	MBC 16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBC 15	MBC 14	MBC 13	MBC 12	MBC 11	MBC 10	MBC 9	MBC 8	MBC 7	MBC 6	MBC 5	MBC 4	MBC 3	MBC 2	MBC 1	MBC 0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MBCn (n = 0-31)	n	rh	<p>Message Buffer Status Changed n (n = 0-31)</p> <p>An MBC flags is set whenever the Message Handler changes on of the status flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB in the Header Section (see “Message Buffer Status (MBS)” on Page 25-176) of the respective Message Buffer 0 to Message Buffer 31. The flags are reset when the Header Section of the Message Buffer is reconfigured or when it has been transferred to the Output Buffer.</p>

FlexRay™ Protocol Controller (E-Ray)

Message Buffer Status Changed 2 (MBSC2)

This register reflect the state of the MBC flags of all configured Message Buffers. If the number of configured Message Buffers is less than 63, the remaining MBCn flags have no meaning. The register is reset when the communication controller leaves “CONFIG” state or enters “STARTUP” state.

MBSC2

Message Buffer Status Changed 2 (0344_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MBC 63	MBC 62	MBC 61	MBC 60	MBC 59	MBC 58	MBC 57	MBC 56	MBC 55	MBC 54	MBC 53	MBC 52	MBC 51	MBC 50	MBC 49	MBC 48
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBC 47	MBC 46	MBC 45	MBC 44	MBC 43	MBC 42	MBC 41	MBC 40	MBC 39	MBC 38	MBC 37	MBC 36	MBC 35	MBC 34	MBC 33	MBC 32
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MBCn (n = 32-63)	n - 32	rh	<p>Message Buffer Status Changed n (n = 32-63)</p> <p>An MBC flags is set whenever the Message Handler changes on of the status flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB in the Header Section (see “Message Buffer Status (MBS)” on Page 25-176) of the respective Message Buffer 32 to Message Buffer 63. The flags are reset when the Header Section of the Message Buffer is reconfigured or when it has been transferred to the Output Buffer.</p>

FlexRay™ Protocol Controller (E-Ray)

Message Buffer Status Changed 3 (MBSC3)

This register reflect the state of the MBC flags of all configured Message Buffers. If the number of configured Message Buffers is less than 95, the remaining MBCn flags have no meaning. The register is reset when the communication controller leaves “CONFIG” state or enters “STARTUP” state.

MBSC3

Message Buffer Status Changed 3 (0348_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MBC 95	MBC 94	MBC 93	MBC 92	MBC 91	MBC 90	MBC 89	MBC 88	MBC 87	MBC 86	MBC 85	MBC 84	MBC 83	MBC 82	MBC 81	MBC 80
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBC 79	MBC 78	MBC 77	MBC 76	MBC 75	MBC 74	MBC 73	MBC 72	MBC 71	MBC 70	MBC 69	MBC 68	MBC 67	MBC 66	MBC 65	MBC 64
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MBCn (n = 64-95)	n - 64	rh	<p>Message Buffer Status Changed n (n = 64-95)</p> <p>An MBC flags is set whenever the Message Handler changes on of the status flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB in the Header Section (see “Message Buffer Status (MBS)” on Page 25-176) of the respective Message Buffer 64 to Message Buffer 95. The flags are reset when the Header Section of the Message Buffer is reconfigured or when it has been transferred to the Output Buffer.</p>

FlexRay™ Protocol Controller (E-Ray)
Message Buffer Status Changed 4 (MBSC4)

This register reflect the state of the MBC flags of all configured Message Buffers. If the number of configured Message Buffers is less than 127, the remaining MBCn flags have no meaning. The register is reset when the communication controller leaves “CONFIG” state or enters “STARTUP” state.

MBSC4
Message Buffer Status Changed 4 (034C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MBC 127	MBC 126	MBC 125	MBC 124	MBC 123	MBC 122	MBC 121	MBC 120	MBC 119	MBC 118	MBC 117	MBC 116	MBC 115	MBC 114	MBC 113	MBC 112
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBC 111	MBC 110	MBC 109	MBC 108	MBC 107	MBC 106	MBC 105	MBC 104	MBC 103	MBC 102	MBC 101	MBC 100	MBC 99	MBC 98	MBC 97	MBC 96
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MBCn (n = 96-127)	n - 96	rh	Message Buffer Status Changed n (n = 96-127) An MBC flags is set whenever the Message Handler changes on of the status flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB in the Header Section (see “Message Buffer Status (MBS)” on Page 25-176) of the respective Message Buffer 96 to Message Buffer 127. The flags are reset when the Header Section of the Message Buffer is reconfigured or when it has been transferred to the Output Buffer.

25.5.2.8 Identification Registers

Core Release Register (CREL)

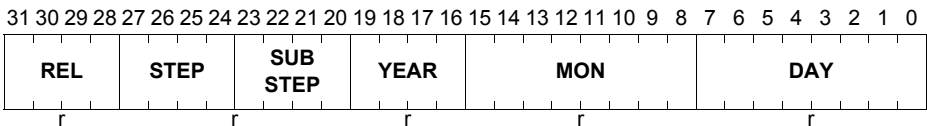
This register contains bit fields about the ERAY module identification. It is read only.

CREL

Core Release Register

(03F0_H)

Reset Value: XXXX XXXX_H



Field	Bits	Type	Description
DAY	[7:0]	r	Design Time Stamp, Day Two digits, BCD-coded.
MON	[15:8]	r	Design Time Stamp, Month Two digits, BCD-coded.
YEAR	[19:16]	r	Design Time Stamp, Year One digit, BCD-coded.
SUBSTEP	[23:20]	r	Sub-Step of Core Release One digits, BCD-coded. 0 _H Alpha, pre-Beta, pre-Beta-update, pre-Beta2, pre-Beta2-update, Beta, Beta2, Revision 1.0.0 1 _H Beta_ct, Beta-ct-fix1, Revision 1.0.1 2 _H Revision1.0RC1,Beta-ct-fix2, REVISION 1.0RC1
STEP	[27:24]	r	Step of Core Release One digits, BCD-coded. 0 _H Revision 1.0.0 1 _H Alpha 2 _H pre-Beta 3 _H pre-Beta-update 4 _H pre-Beta2 5 _H pre-Beta2-update 6 _H Beta 7 _H Beta2

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
REL	[31:28]	r	Core Release One digit, BCD-coded. 0 _B alpha...beta2ct 1 _B Revision 1.0

Table 25-7 Coding of releases

Release	Step	Sub-Step	Name	Release Date
0	1	0	Alpha	
0	2	0	pre-Beta	
0	3	0	pre-Beta-update	
0	4	0	pre-Beta2	
0	5	0	pre-Beta2-update	
0	6	0	Beta	
0	6	1	Beta-ct-fix1	14.10.2005
0	6	2	Beta-ct-fix2	14.12.2005
0	7	0	Beta2	03.02.2006
0	7	1	Beta2ct	24.03.2006
0	7	2	Revision 1.0RC1	07.04.2006
1	0	0	Release 1.0.0	19.05.2006
1	0	1	Release 1.0.1	2006
1	0	2	Release 1.0.2	31.10.2007

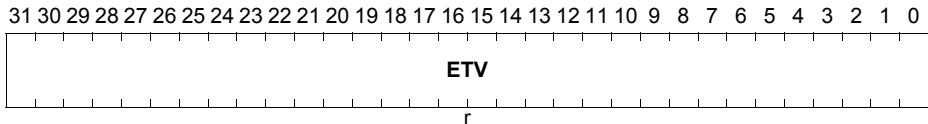
FlexRay™ Protocol Controller (E-Ray)

Endian Register (ENDN)

This register may be used to check, if the data of the E-Ray is handled by a host with the correct endian format. It is read only.

ENDN

Endian Register (003F4_H) Reset Value: 8765 4321_H



Field	Bits	Type	Description
ETV	[31:0]	r	Endianness Test Value The endianness test value.

25.5.2.9 Input Buffer

Double buffer structure consisting of Input Buffer Host and Input Buffer Shadow. While the Host can write to Input Buffer Host, the transfer to the Message RAM is done from Input Buffer Shadow. The Input Buffer holds the Header and Data Sections to be transferred to the selected Message Buffer in the Message RAM. It is used to configure the Message Buffers in the Message RAM and to update the Data Sections of transmit buffers.

When updating the Header Section of a Message Buffer in the Message RAM from the Input Buffer, the Message Buffer Status as described in **“Message Buffer Status (MBS)” on Page 25-176** is automatically reset to zero.

The Header Sections of Message Buffers belonging to the receive FIFO can only be (re)configured when the Communication Controller is in “DEFAULT_CONFIG” or “CONFIG” state. For those Message Buffers only the payload length configured and the data pointer need to be configured via WRHS2.PLC and WRHS2.DP. All information required for acceptance filtering is taken from the FIFO rejection filter and the FIFO rejection filter mask.

The data transfer between Input Buffer (IBF) and Message RAM is described in detail in **“Data Transfer from Input Buffer to Message RAM” on Page 25-224**.

Write Header Section 1 (WRHS1)
WRHS1
Write Header Section 1 (0500_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		MBI	TXM	PPIT	CFG	CHB	CHA	0		CYC					
r	rw		rw	rw	rw	rw	rw	r	rw						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				FID											
r				rw											

Field	Bits	Type	Description
FID	[10:0]	rw	Frame ID Frame ID of the selected Message Buffer. The Frame ID defines the slot number for transmission / reception of the respective message. Message Buffers with Frame ID = 0 are considered as not valid.
CYC	[22:16]	rw	Cycle Code The 7-bit cycle code determines the cycle set used for cycle counter filtering. For details about the configuration of the cycle code see Section 25.6.7.3 .
CHA	24	rw	Channel Filter Control A The channel filtering field A associated with the buffer serves of channel A as a filter for receive buffers, and as a control field for transmit buffers
CHB	25	rw	Channel Filter Control B The channel filtering field B associated with the buffer serves of channel B as a filter for receive buffers, and as a control field for transmit buffers
CFG	26	rw	Message Buffer Direction Configuration Bit This bit is used to configure the corresponding buffer as a transmit buffer or as a receive buffer. For Message Buffers belonging to the receive FIFO the bit is not evaluated. 0 _B The corresponding buffer is configured as Receive Buffer 1 _B The corresponding buffer is configured as Transmit Buffer

FlexRay™ Protocol Controller (E-Ray)

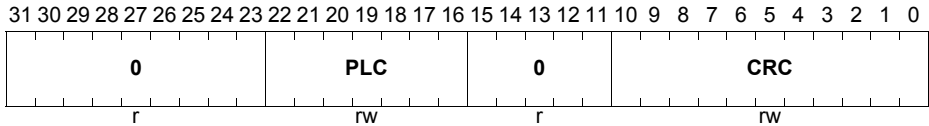
Field	Bits	Type	Description
PPIT	27	rw	Payload Preamble Indicator Transmit This bit is used to control the state of the Payload Preamble Indicator in transmit Frames. If the bit is set in a static Message Buffer, the respective Message Buffer holds Network Management information. If the bit is set in a dynamic Message Buffer the first two byte of the Payload Segment may be used for message ID filtering by the receiver. Message ID filtering of received FlexRay™ Frames is not supported by the E-Ray module, but can be done by the Host. 0 _B Payload Preamble Indicator not set 1 _B Payload Preamble Indicator set
TXM	28	rw	Transmission Mode This bit is used to select the transmission mode (see “Transmit Buffers” on Page 25-217). 0 _B Continuous mode 1 _B Single-shot mode
MBI	29	rw	Message Buffer Service Request This bit enables the receive / transmit service request for the corresponding Message Buffer. After a dedicated receive buffer has been updated by the Message Handler, flag SIR.RXI and /or SIR.MBSI in the Status Service Request register are set. After a transmission has completed flag SIR.TXI is set. 0 _B The corresponding Message Buffer service request is disabled 1 _B The corresponding Message Buffer service request is enabled
0	[15:11], 23, [31:30]	r	Reserved Returns 0 if read; should be written with 0.

Note: The Input Buffer RAMs are initialized to zero when leaving application reset or by CHI command CLEAR_RAMs. Note that only the currently active IBF bank is cleared. To clear the 2nd bank as well, CUST1.IBF1PAG and CUST1.IBF2PAG need to be set and command CLEAR_RAMs need to be issued again. This is required in particular after an application reset. If the 2nd bank of IBF is left unused, this procedure is not required.

Table 25-8 Channel Filter Control Bits

CHA	CHB	Transmit Buffer transmit Frame on	Receive Buffer store Frame received from
1 ¹⁾	1 ¹⁾	Both Channels (static segment only)	Channel A or B (store first semantically valid Frame, static segment only)
1	0	Channel A	Channel A
0	1	Channel B	Channel B
0	0	No Transmission	Ignore Frame

1) If a Message Buffer is configured for the dynamic segment and both bits of the channel filtering field are set to 1, no Frames are transmitted resp. received Frames are ignored (same function as CHA = CHB = 0)

FlexRay™ Protocol Controller (E-Ray)
Write Header Section 2 (WRHS2)
WRHS2
Write Header Section 2
(0504_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CRC	[10:0]	rw	Header CRC (vRF!Header!HeaderCRC) Receive Buffer: Configuration not required Transmit Buffer: Header CRC calculated and configured by the Host. For calculation of the Header CRC the payload length of the Frame send on the bus has to be considered. In static segment the payload length of all Frames is configured by MHDC.SFDL.
PLC	[22:16]	rw	Payload Length Configured Length of Data Section (number of 2-byte words) as configured by the Host. During static segment the static Frame payload length as configured by MHDC.SFDL in the MHD Configuration Register defines the payload length for all static Frames. If the payload length configured by PLC is shorter than this value padding byte are inserted to ensure that Frames have proper physical length. The padding pattern is logical zero.
0	[15:11], [31:23]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

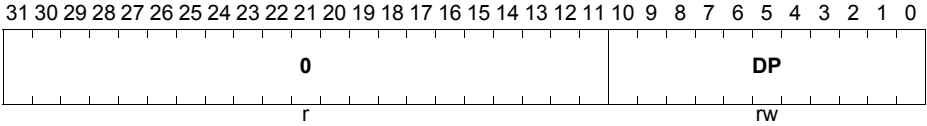
Write Header Section 3 (WRHS3)

WRHS3

Write Header Section 3

(0508_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
DP	[10:0]	rw	Data Pointer Pointer to the first 32-bit word of the Data Section of the addressed Message Buffer in the Message RAM.
0	[31:11]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
LHSS	16	rh	Load Header Section Shadow 0 _B Header Section is not updated 1 _B Header Section selected for transfer from Input Buffer to the Message RAM (transfer is ongoing of finalized)
LDSS	17	rh	Load Data Section Shadow 0 _B Data Section is not updated 1 _B Data Section selected for transfer from Input Buffer to the Message RAM (transfer is ongoing of finalized)
STXRS	18	rh	Transmission Request Shadow If this bit is set to 1, the Transmission Request flag TXRQ1.TXRn (n = 0-31) to TXRQ4.TXRn (n = 0-31) for the selected Message Buffer is set in the Transmission Request Registers to release the Message Buffer for transmission. In single-shot mode the flag is cleared by the Communication Controller after transmission has completed. TXRQ1.TXRn (n = 0-31) to TXRQ4.TXRn (n = 0-31) are evaluated for transmit buffer only. 0 _B Reset Transmission Request flag 1 _B Set Transmission Request flag, transmit buffer released for transmission (operation is ongoing of finalized)
0	[15:3], [31:19]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)
Input Buffer Command Request (IBCR)

When the Host writes the number of the target Message Buffer in the Message RAM to IBRH in the Input Buffer Command Request register, IBF Host and IBF Shadow are swapped. In addition the Message Buffer numbers stored under IBRH and IBRS are also swapped (see also [“Data Transfer from Input Buffer to Message RAM” on Page 25-224](#)).

With this write operation the IBSYS bit in the Input Buffer Command Request register is set to 1. The Message Handler then starts to transfer the contents of IBF Shadow to the Message Buffer in the Message RAM selected by IBRS.

While the Message Handler transfers the data from IBF Shadow to the target Message Buffer in the Message RAM, the Host may write the next message into the IBF Host. After the transfer between IBF Shadow and the Message RAM has completed, the IBSYS bit is set back to 0 and the next transfer to the Message RAM may be started by the Host by writing the respective target Message Buffer number to IBRH.

If a write access to IBRH occurs while IBSYS is 1, IBSYH is set to 1. After completion of the ongoing data transfer from IBF Shadow to the Message RAM, IBF Host and IBF Shadow are swapped, IBSYH is reset to 0. IBSYS remains set to 1, and the next transfer to the Message RAM is started. In addition the Message Buffer numbers stored under IBRH and IBRS are also swapped. Any write access to an Input Buffer register while both IBSYS and IBSYH are set will cause the error flag EIR.IIBA to be set. In this case the Input Buffer will not be changed.

IBCR
Input Buffer Command Request (0514_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IBSYS		0						IBRS							
rh		r						rh							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IBSYH		0						IBRH							
rh		r						rwh							

Field	Bits	Type	Description
IBRH	[6:0]	rwh	Input Buffer Request Host Selects the target Message Buffer in the Message RAM for data transfer from Input Buffer. Valid values are 00 _H to 7F _H (0...127).

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
IBSYH	15	rh	Input Buffer Busy Host Set to 1 by writing IBRH while IBSYS is still 1. After the ongoing transfer between IBF Shadow and the Message RAM has completed, the IBSYH is set back to 0. 0 _B No request pending 1 _B Request while transfer between IBF Shadow and Message RAM in progress
IBRS	[22:16]	rh	Input Buffer Request Shadow Number of the target Message Buffer actually updated/lately updated. Valid values are 00 _H to 7F _H (0...127).
IBSYS	31	rh	Input Buffer Busy Shadow Set to 1 after writing IBRH. When the transfer between IBF Shadow and the Message RAM has completed, IBSYS is set back to 0. 0 _B Transfer between IBF Shadow and Message RAM completed 1 _B Transfer between IBF Shadow and Message RAM in progress
0	[14:7], [30:23]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)
Read Header Section 1 (RDHS1)

Values as configured by the Host via WRHS1 Register:

RDHS1
Read Header Section 1
(0700_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		MBI	TXM	PPIT	CFG	CHB	CHA	0	CYC						
r		rh	rh	rh	rh	rh	rh	r	rh						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					FID										
r					rh										

Field	Bits	Type	Description
FID	[10:0]	rh	Frame ID
CYC	[22:16]	rh	Cycle Code
CHA	24	rh	Channel Filter Control A
CHB	25	rh	Channel Filter Control B
CFG	26	rh	Message Buffer Direction Configuration Bit
PPIT	27	rh	Payload Preamble Indicator Transmit
TXM	28	rh	Transmission Mode
MBI	29	rh	Message Buffer Service Request
0	[15:11], 23, [31:30]	r	Reserved Returns 0 if read; should be written with 0.

Note: In case that the Message Buffer read from the Message RAM belongs to the receive FIFO, FID holds the received Frame ID, while CYC, CHA, CHB, CFG, PPIT, TXM, and MBI are reset to zero.

Table 25-9 Channel Filter Control Bits

CHA	CHB	Transmit Buffer transmit Frame on	Receive Buffer store Frame received from
1 ¹⁾	1 ¹⁾	Both Channels (static segment only)	Channel A or B (store first semantically valid Frame, static segment only)
1	0	Channel A	Channel A
0	1	Channel B	Channel B
0	0	No Transmission	Ignore Frame

1) If a Message Buffer is configured for the dynamic segment and both bits of the channel filtering field are set to 1, no Frames are transmitted resp. received Frames are ignored (same function as CHA = CHB = 0)

FlexRay™ Protocol Controller (E-Ray)

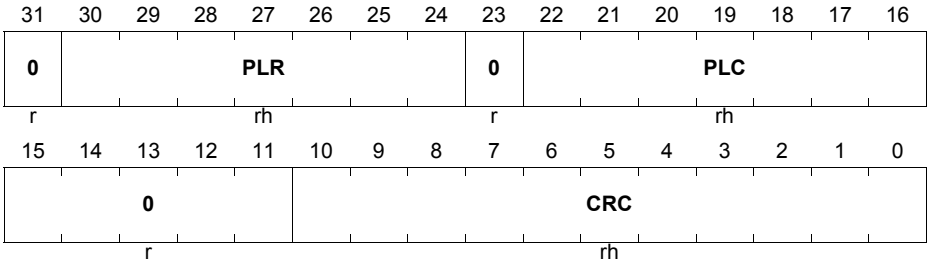
Read Header Section 2 (RDHS2)

RDHS2

Read Header Section 2

(0704_H)

Reset Value: 0000 0000_H

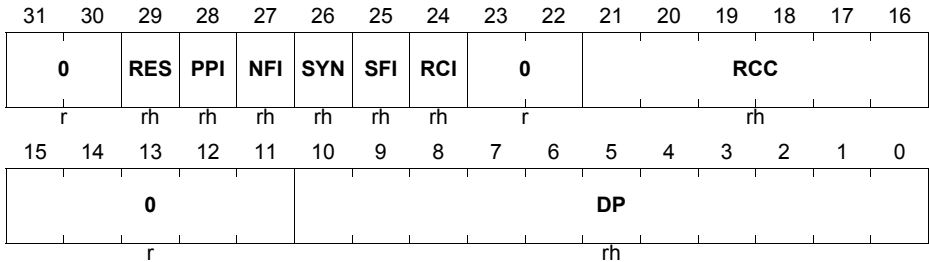


Field	Bits	Type	Description
CRC	[10:0]	rh	Header CRC (vRF!Header!HeaderCRC) Receive Buffer: Configuration not required. Header CRC updated from receive Data Frames. Transmit Buffer: Header CRC calculated and configured by the Host
PLC	[22:16]	rh	Payload Length Configured Length of Data Section (number of 2-byte words) as configured by the Host.

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
PLR	[30:24]	rh	<p>Payload Length Received (vRF!Header!Length) Payload length value updated from received Data Frame (exception: if Message Buffer belongs to the receive FIFO PLR is also updated from received NULL Frames). When a message is stored into a Message Buffer the following behavior with respect to payload length received and payload length configured is implemented:</p> <ul style="list-style-type: none"> • PLR > PLC: The payload data stored in the Message Buffer is truncated to the payload length configured for even PLC or else truncated to PLC + 1. • PLR ≤ PLC: The received payload data is stored into the Message Buffers Data Section. The remaining data bytes of the Data Section as configured by PLC are filled with undefined data. • PLR = 0: The Message Buffer's Data Section is filled with undefined data. • PLC = 0: Message Buffer has no Data Section configured. No data is stored into the Message Buffer's Data Section.
0	[15:11], 23, 31	r	<p>Reserved Returns 0 if read; should be written with 0.</p>

Note: The Message RAM is organized in 4-byte words. When received data is stored into a Message Buffer's Data Section, the number of 2-byte data words written into the Message Buffer is PLC rounded to the next even value. PLC should be configured identical for all Message Buffers belonging to the receive FIFO. Header 2 is updated from Data Frames only.

FlexRay™ Protocol Controller (E-Ray)
Read Header Section 3 (RDHS3)
RDHS3
Read Header Section 3
(0708_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
DP	[10:0]	rh	Data Pointer Pointer to the first 32-bit word of the Data Section of the addressed Message Buffer in the Message RAM.
RCC	[21:16]	rh	Receive Cycle Count (vRF!Header!CycleCount) Cycle counter value updated from received Data Frame.
RCI	24	rh	Received on Channel Indicator (vSS!Channel) Indicates the channel from which the received Data Frame was taken to update the respective receive buffer. 0 _B Frame received on channel B 1 _B Frame received on channel A
SFI	25	rh	Startup Frame Indicator (vRF!Header!SuFIndicator) A Startup Frame is marked by the Startup Frame indicator. 0 _B The received Frame is not a startup Frame 1 _B The received Frame is a startup Frame
SYN	26	rh	SYNC Frame Indicator (vRF!Header!SyFIndicator) A SYNC Frame is marked by the SYNC Frame indicator. 0 _B The received Frame is not a SYNC Frame 1 _B The received Frame is a SYNC Frame
NFI	27	rh	NULL Frame Indicator (vRF!Header!NFIndicator) Is set to 1 after storage of the first received Data Frame. 0 _B Up to now no Data Frame has been stored into the respective Message Buffer 1 _B At least one Data Frame has been stored into the respective Message Buffer

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
PPI	28	rh	Payload Preamble Indicator (vRF!Header!PPIndicator) The payload preamble indicator defines whether a Network Management vector or message ID is contained within the Payload Segment of the received Frame. 0 _B The Payload Segment of the received Frame does not contain a Network Management vector nor a message ID 1 _B Static segment: Network Management vector in the first part of the payload Dynamic segment: Message ID in the first part of the payload
RES	29	rh	Reserved Bit (vRF!Header!Reserved) Reflects the state of the received reserved bit. The reserved bit is transmitted as 0.
0	[15:11], [23:22], [31:30]	r	Reserved Returns 0 if read; should be written with 0.

Note: Header 3 is updated from Data Frames only.

FlexRay™ Protocol Controller (E-Ray)
Message Buffer Status (MBS)

The Message Buffer status is updated by the Communication Controller with respect to the assigned channel(s) latest at the end of the slot following the slot assigned to the Message Buffer. The flags are updated only when the Communication Controller is in “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state. If only one channel (A or B) is assigned to a Message Buffer, the channel-specific status flags of the other channel are written to zero. If both channels are assigned to a Message Buffer, the channel-specific status flags of both channels are updated. The Message Buffer status is updated only when the slot counter reached the configured Frame ID and when the cycle counter filter matched. When the Host updates a Message Buffer via Input Buffer, all MBS flags are reset to zero independent of which IBCM bits are set or not. For details about receive / transmit filtering see **“Filtering and Masking” on Page 25-213**, **“Transmit Process” on Page 25-217**, and **“Receive Process” on Page 25-220**.

Whenever the Message Handler changes one of the flags VFRA, VFRB, SEOA, SEOB, CEOA, CEOB, SVOA, SVOB, TCIA, TCIB, ESA, ESB, MLST, FTA, FTB the respective Message Buffer’s MBC flag in registers MBSC1 to MBSC4 is set

MBS
Message Buffer Status (070C_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	RES S	PPIS	NFIS	SYN S	SFIS	RCIS	0								
r	rh	rh	rh	rh	rh	rh	r						rh		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FTB	FTA	0	ML ST	ESB	ESA	TCIB	TCIA	SV OB	SV OA	CE OB	CE OA	SE OB	SE OA	VR FB	VR FA
rh	rh	r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
VFRA	0	rh	Valid Frame Received on Channel A (vSS!ValidFrameA) A valid Frame indication is set if a valid Frame was received on channel A. 0 _B No valid Frame received on channel A 1 _B Valid Frame received on channel A
VFRB	1	rh	Valid Frame Received on Channel B (vSS!ValidFrameB) A valid Frame indication is set if a valid Frame was received on channel B. 0 _B No valid Frame received on channel B 1 _B Valid Frame received on channel B

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
SEOA	2	rh	Syntax Error Observed on Channel A (vSS!SyntaxErrorA) A syntax error was observed in the assigned slot on channel A. 0 _B No syntax error observed on channel A 1 _B Syntax error observed on channel A
SEOB	3	rh	Syntax Error Observed on Channel B (vSS!SyntaxErrorB) A syntax error was observed in the assigned slot on channel B. 0 _B No syntax error observed on channel B 1 _B Syntax error observed on channel B
CEOA	4	rh	Content Error Observed on Channel A (vSS!ContentErrorA) A content error was observed in the assigned slot on channel A. 0 _B No content error observed on channel A 1 _B Content error observed on channel A
CEOB	5	rh	Content Error Observed on Channel B (vSS!ContentErrorB) A content error was observed in the assigned slot on channel B. 0 _B No content error observed on channel B 1 _B Content error observed on channel B
SVOA	6	rh	Slot Boundary Violation Observed on Channel A (vSS!BViolationA) A slot boundary violation (channel active at the start or at the end of the assigned slot) was observed on channel A. 0 _B No slot boundary violation observed on channel A 1 _B Slot boundary violation observed on channel A
SVOB	7	rh	Slot Boundary Violation Observed on Channel B (vSS!BViolationB) A slot boundary violation (channel active at the start or at the end of the assigned slot) was observed on channel B. 0 _B No slot boundary violation observed on channel B 1 _B Slot boundary violation observed on channel B
TCIA	8	rh	Transmission Conflict Indication Channel A (vSS!TxConflictA) A transmission conflict indication is set if a transmission conflict has occurred on channel A. 0 _B No transmission conflict occurred on channel A 1 _B Transmission conflict occurred on channel A

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
TCIB	9	rh	<p>Transmission Conflict Indication Channel B (vSS!TxConflictB)</p> <p>A transmission conflict indication is set if a transmission conflict has occurred on channel B.</p> <p>0_B No transmission conflict occurred on channel B 1_B Transmission conflict occurred on channel B</p>
ESA	10	rh	<p>Empty Slot Channel A</p> <p>In an empty slot there is no activity detected on the bus. The condition is checked in static and dynamic slots.</p> <p>0_B Bus activity detected in the assigned slot on channel A 1_B No bus activity detected in the assigned slot on channel A</p>
ESB	11	rh	<p>Empty Slot Channel B</p> <p>In an empty slot there is no activity detected on the bus. The condition is checked in static and dynamic slots.</p> <p>0_B Bus activity detected in the assigned slot on channel B 1_B No bus activity detected in the assigned slot on channel B</p>
MLST	12	rh	<p>Message Lost</p> <p>The flag is set in case the Host did not read the message before the Message Buffer was updated from a received Data Frame. Not affected by reception of NULL Frames except for Message Buffers belonging to the receive FIFO. The flag is reset by a Host write to the Message Buffer via IBF or when a new message is stored into the Message Buffer after the Message Buffers ND flag was reset by reading out the Message Buffer via OBF.</p> <p>0_B No message lost 1_B Unprocessed message was overwritten</p>
FTA	14	rh	<p>Frame Transmitted on Channel A</p> <p>Indicates that this node has transmitted a Data Frame in the assigned slot on channel A.</p> <p>0_B No transmission transmitted on channel A 1_B Data Frame transmitted on channel A in cycle defined by CCS bit field</p> <p><i>Note: The FlexRay™ protocol specification requires that FTA can only be reset by the Host. Therefore the Cycle Count Status CCS for these bits is only valid for the cycle where the bits are set to 1</i></p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
FTB	15	rh	<p>Frame Transmitted on Channel B Indicates that this node has transmitted a Data Frame in the assigned slot on channel B.</p> <p>0_B No transmission transmitted on channel B 1_B Data Frame transmitted on channel B in cycle defined by CCS bit field</p> <p><i>Note: The FlexRay™ protocol specification requires that FTB can only be reset by the Host. Therefore the Cycle Count Status CCS for these bits is only valid for the cycle where the bits are set to 1</i></p>
CCS	[21:16]	rh	<p>Cycle Count Status Cycle Count when status (MBS register) has been updated.</p>
RCIS	24	rh	<p>Received on Channel Indicator Status (vSS!Channel) Indicates the channel on which the Frame was received.</p> <p>0_B Frame received on channel B 1_B Frame received on channel A</p> <p><i>Note: For receive buffers (CFG = 0) the RCIS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.</i></p>
SFIS	25	rh	<p>Startup Frame Indicator Status (vRF!Header!SuFIndicator) A Startup Frame is marked by the Startup Frame indicator.</p> <p>0_B No Startup Frame received 1_B The received Frame is a startup Frame</p> <p><i>Note: For receive buffers (CFG = 0) the SFIS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.</i></p>
SYNS	26	rh	<p>SYNC Frame Indicator Status (vRF!Header!SyFIndicator) A Startup Frame is marked by the Startup Frame indicator.</p> <p>0_B No SYNC Frame received 1_B The received Frame is a SYNC Frame</p> <p><i>Note: For receive buffers (CFG = 0) the SYNS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.</i></p>

FlexRay™ Protocol Controller (E-Ray)

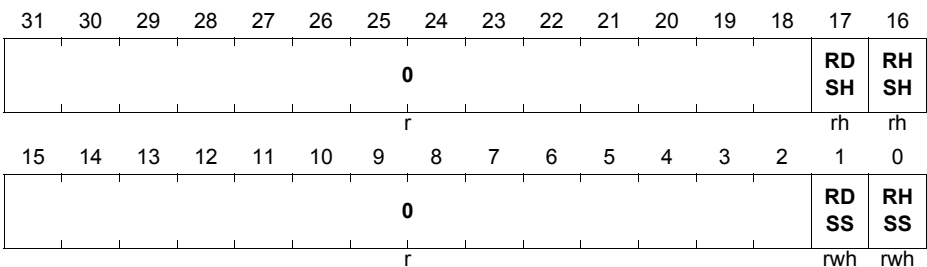
Field	Bits	Type	Description
NFIS	27	rh	<p>NULL Frame Indicator Status (vRF!Header!NFIndicator) If reset to 0 the Payload Segment of the received Frame contains no usable data.</p> <p>0_B Received Frame is a NULL Frame 1_B Received Frame is not a NULL Frame</p> <p><i>Note: For receive buffers (CFG = 0) the NFIS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.</i></p>
PPIS	28	rh	<p>Payload Preamble Indicator Status (vRF!Header!PPIIndicator) The payload preamble indicator defines whether a Network Management vector or message ID is contained within the Payload Segment of the received Frame.</p> <p>0_B Static Segment: The Payload Segment of the received Frame does not contain a Network Management vector or a message ID Dynamic Segment: The Payload Segment of the received Frame does not contain a Network Management vector or a message ID</p> <p>1_B Static Segment: Network Management vector at the beginning of the payload Dynamic Segment: Message ID at the beginning of the payload</p> <p><i>Note: For receive buffers (CFG = 0) the PPIS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.</i></p>
RESS	29	rh	<p>Reserved Bit Status (vRF!Header!Reserved) Reflects the state of the received reserved bit. The reserved bit is transmitted as 0.</p> <p><i>Note: For receive buffers (CFG = 0) the RESS is updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.</i></p>

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
0	13, [23:22], [31:30]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)
Output Buffer Command Mask (OBCM)

Configures how the Output Buffer is updated from the Message Buffer in the Message RAM selected by the Output Buffer Command Request register. If OBF Host and OBF Shadow are swapped, also mask bits OBCM.RDSH and OBCM.RHSH are swapped with bits OBCM.RDSS and OBCM.RHSS to keep them attached to the respective Output Buffer transfer.

OBCM
Output Buffer Command Mask (0710_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
RHSS	0	rwh	Read Header Section Shadow 0 _B Header Section is not read 1 _B Header Section selected for transfer from Message RAM to Output Buffer
RDSS	1	rwh	Read Data Section Shadow 0 _B Data Section is not read 1 _B Data Section selected for transfer from Message RAM to Output Buffer
RHSH	16	rh	Read Header Section Host 0 _B Header Section is not read 1 _B Header Section selected for transfer from Message RAM to Output Buffer
RDSH	17	rh	Read Data Section Host 0 _B Data Section is not read 1 _B Data Section selected for transfer from Message RAM to Output Buffer
0	[15:2], [31:18]	r	Reserved Returns 0 if read; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

Note: After the transfer of the Header Section from the Message RAM to OBF Shadow has completed, the Message Buffer status changed flag MBCn (n = 0-31) to MBCn (n = 96-127) of the selected Message Buffer in the Message Buffer Changed MBSC1 to MBSC4 registers is cleared. After the transfer of the Data Section from the Message RAM to OBF Shadow has completed, the New Data flag NDn (n = 0-31) to NDn (n = 96-127) of the selected Message Buffer in the New Data NDAT1 to NDAT4 registers is cleared.

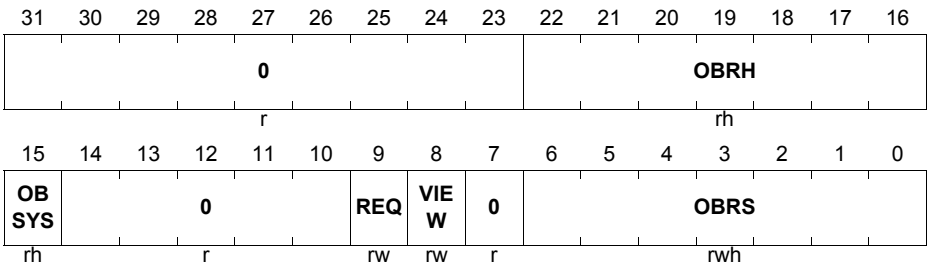
Output Buffer Command Request (OBCR)

The Message Buffer selected by OBCR.OBRS is transferred from the Message RAM to the Output Buffer as soon as the Host has set OBCR.REQ. Bit OBCR.REQ can only be set while OBCR.OBSYS is 0 (see also [“Data Transfer from Message RAM to Output Buffer” on Page 25-226](#)).

After setting OBCR.REQ, OBCR.OBSYS is automatically set, and the transfer of the Message Buffer selected by OBCR.OBRS from the Message RAM to Output Buffer Shadow is started. When the transfer between the Message RAM and OBF Shadow has completed, this is signalled by clearing OBCR.OBSYS. By setting OBCR.VIEW while OBCR.OBSYS is 0, OBF Host and OBF Shadow are swapped. When Output Buffer Host and Output Buffer Shadow are swapped, also mask bits OBCM.RDSH and OBCM.RHSH are swapped with bits OBCM.RDSS and OBCM.RHSS to keep them attached to the respective Output Buffer transfer. Now the Host can read the transferred Message Buffer from OBF Host. In parallel the Message Handler may transfer the next message from the Message RAM to OBF Shadow if OBCR.VIEW and OBCR.REQ are set at the same time.

Any write access to an Output Buffer register while OBCR.OBSYS is set will cause the error flag EIR.IOBA to be set. In this case the Output Buffer will not be changed.

FlexRay™ Protocol Controller (E-Ray)

OBCR
Output Buffer Command Request (0714_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
OBSRS	[6:0]	rw	Output Buffer Request Shadow Number of source Message Buffer to be transferred from the Message RAM to OBF Shadow. Valid values are 00 _H to 7F _H (0 to 127). If the number of the first Message Buffer of the receive FIFO is written to this register the Message Handler transfers the Message Buffer addressed by the GET Index Register (GIDX, “FIFO Function” on Page 25-221) to OBF Shadow.
VIEW	8	rw	View Shadow Buffer Toggles between OBF Shadow and OBF Host. Only writeable while OBCR.OBSYS = 0. 0 _B No action 1 _B Swap OBF Shadow and OBF Host
REQ	9	rw	Request Message RAM Transfer Requests transfer of Message Buffer addressed by OBCR.OBSRS from Message RAM to OBF Shadow. Only writeable while OBCR.OBSYS = 0. 0 _B No request 1 _B Transfer to OBF Shadow requested
OBSYS	15	rh	Output Buffer Busy Shadow Set to 1 after setting bit OBCR.REQ. When the transfer between the Message RAM and OBF Shadow has completed, OBCR.OBSYS is cleared again. 0 _B No transfer in progress 1 _B Transfer between Message RAM and OBF Shadow in progress

FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
OBRH	[22:16]	rh	<p>Output Buffer Request Host</p> <p>Number of Message Buffer currently accessible by the Host via RDHS1 to RDHS3, MBS, and RDDSnn (nn = 01-64). By setting OBCR.VIEW OBF Shadow and OBF Host are swapped and the transferred Message Buffer is accessible by the Host.</p> <p>Valid values are 00_H to 7F_H (01 to 27).</p>
0	7, [14:10], [31:23]	r	<p>Reserved</p> <p>Returns 0 if read; should be written with 0.</p>

25.6 Functional Description

This chapter describes the E-Ray implementation together with the related FlexRay™ protocol features. More information about the FlexRay™ protocol itself can be found in the FlexRay™ protocol specification v2.1.

Communication on FlexRay™ networks is based on Frames and symbols. The wakeup symbol (WUS) and the collision avoidance symbol (CAS) are transmitted outside the communication cycle to setup the time schedule. Frames and media access test symbols (MTS) are transmitted inside the communication cycle.

25.6.1 Communication Cycle

A communication cycle in FlexRay™ consists of the following elements:

- Static Segment
- Dynamic Segment
- Symbol Window
- Network Idle Time (NIT)

Static segment, dynamic segment, and symbol window form the Network Communication Time (NCT). For each communication channel the slot counter starts at 1 and counts up until the end of the dynamic segment is reached. Both channels share the same arbitration grid which means that they use the same synchronized MacroTICK.

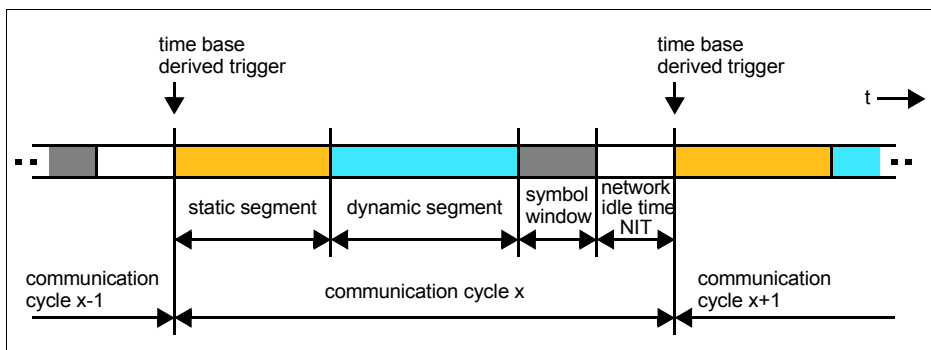


Figure 25-4 Structure of Communication Cycle

25.6.1.1 Static Segment

The Static Segment is characterized by the following features:

- Time slots of fixed length (optionally protected by bus guardian)
- Start of Frame transmission at action point of the respective static slot
- Payload length same for all Frames on both channel

FlexRay™ Protocol Controller (E-Ray)

Parameters: Number of Static Slots **GTUC07.NSS**, Static Slot Length **GTUC07.SSL**, Payload Length Static **MHDC.SFDL**, Action Point Offset **GTUC09.APO**.

25.6.1.2 Dynamic Segment

The Dynamic Segment is characterized by the following features:

- All controllers have bus access (no bus guardian protection possible)
- Variable payload length and duration of slots, different for both channels
- Start of transmission at minislot action point

Parameters: Number of Minislots **GTUC08.NMS**, Minislot Length **GTUC08.MSL**, Minislot Action Point Offset **GTUC09.MAPO**, Start of Latest Transmit (last minislot) **MHDC.SLT**.

25.6.1.3 Symbol Window

During the symbol window only one media access test symbol (MTS) may be transmitted per channel. MTS symbols are sent in "NORMAL_ACTIVE" state to test the bus guardian.

The symbol window is characterized by the following features:

- Send single symbol
- Transmission of the MTS symbol starts at the symbol windows action point

Parameters: Symbol Window Action Point Offset **GTUC09.APO** (same as for static slots), Network Idle Time Start **GTUC04.NIT**.

25.6.1.4 Network Idle Time (NIT)

During network idle time the Communication Controller has to perform the following tasks:

- Calculate clock correction terms (offset and rate)
- Distribute offset correction over multiple MacroTicks
- Perform cluster cycle related tasks

Parameters: Network Idle Time Start **GTUC04.NIT**, Offset Correction Start **GTUC04.OCS**.

25.6.1.5 Configuration of Network Idle Time (NIT) Start and Offset Correction Start

The number of MacroTicks per cycle (gMacroPerCycle) is assumed to be m . It is configured by programming **GTUC02.MPC** = m .

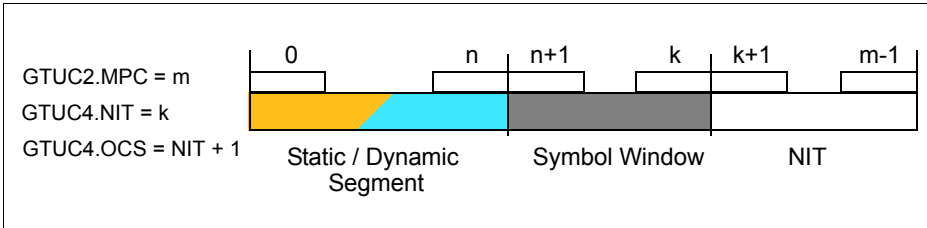


Figure 25-5 Configuration of network idle time (NIT) start and offset correction start

The static / dynamic segment starts with Macrotick 0 and ends with Macrotick n:
 $n = \text{static segment length} + \text{dynamic segment offset} + \text{dynamic segment length} - 1 \text{ Macrotick}$

$$n = g\text{NumberOfStaticSlots} \cdot g\text{dStaticSlot} + \text{dynamic segment offset} + g\text{NumberOfMinislots} \cdot g\text{dMinislot} - 1 \text{ Macroticks}$$

The static segment length is configured by [GTUC07.SSL](#) and [GTUC07.NSS](#).
 The dynamic segment length is configured by [GTUC08.MSL](#) and [GTUC08.NMS](#).

The dynamic segment offset is:

If $g\text{dActionPointOffset} \leq g\text{dMinislotActionPointOffset}$:

$$\text{dynamic segment offset} = 0 \text{ MT}$$

Else if $g\text{dActionPointOffset} > g\text{dMinislotActionPointOffset}$:

$$\text{dynamic segment offset} = g\text{dActionPointOffset} - g\text{dMinislotActionPointOffset}$$

The network idle time (NIT) starts with Macrotick k+1 and ends with the last Macrotick of cycle m-1. It has to be configured by setting [GTUC04.NIT](#) = k.

For the E-Ray the offset correction start is required to be

$$\text{GTUC04.OCS} \geq \text{GTUC04.NIT} + 1 = k+1.$$

The length of symbol window results from the number of Macroticks between the end of the static / dynamic segment and the beginning of the NIT. It can be calculated by $k - n$.

25.6.2 Communication Modes

The FlexRay™ Protocol Specification v2.1 defines the Time-Triggered Distributed (TT-D) mode.

Time-triggered Distributed (TT-D)

In TT-D mode the following configurations are possible:

- **Pure static:** minimum 2 static slots + symbol window (optional)
- **Mixed static/dynamic:** minimum 2 static slots + dynamic segment + symbol window (optional)

A minimum of two coldstart nodes need to be configured for distributed time-triggered operation. Two fault-free coldstart nodes are necessary for the cluster startup. Each Startup Frame must be a SYNC Frame, therefore all coldstart nodes are sync nodes.

25.6.3 Clock Synchronization

In TT-D mode a distributed clock synchronization is used. Each node individually synchronizes itself to the cluster by observing the timing of received SYNC Frames from other nodes.

25.6.3.1 Global Time

Activities in a FlexRay™ node, including communication, are based on the concept of a global time, even though each individual node maintains its own view of it. It is the clock synchronization mechanism that differentiates the FlexRay™ cluster from other node collections with independent clock mechanisms. The global time is a vector of two values; the cycle (cycle counter) and the cycle time (Macrotick counter).

Cluster specific:

- Macrotick = basic unit of time measurement in a FlexRay™ network, a Macrotick consists of an integer number of Microticks
- Cycle length = duration of a communication cycle in units of Macroticks

25.6.3.2 Local Time

Internally, nodes time their behavior with Microtick resolution. Microticks are time units derived from the oscillator clock tick of the specific node. Therefore Microticks are controller-specific units. They may have different duration in different controllers. The precision of a node's local time difference measurements is a Microtick.

FlexRay™ Protocol Controller (E-Ray)

Node specific:

- Oscillator clock → prescaler → Microtick
- Microtick = basic unit of time measurement in a Communication Controller, clock correction is done in units of Microticks
- Cycle counter + Macrotick counter = nodes local view of the global time

25.6.3.3 Synchronization Process

Clock synchronization is performed by means of SYNC Frames. Only preconfigured nodes (sync nodes) are allowed to send SYNC Frames. In a two-channel cluster a sync node has to send its SYNC Frame on both channels.

For synchronization in FlexRay™ the following constraints have to be considered:

- Max. one SYNC Frame per node in one communication cycle
- Max. 15 SYNC Frames per cluster in one communication cycle
- Every node has to use all available SYNC Frames for clock synchronization
- Minimum of two sync nodes required for clock synchronization and startup

For clock synchronization the time difference between expected and observed arrival time of SYNC Frames received during the static segment, valid on both channels (two-channel cluster), is measured. The calculation of correction terms is done during network idle time (NIT) (offset: every cycle, rate: odd cycle) by using a FTA / FTM algorithm. For details see FlexRay™ protocol specification v2.1, chapter 8.

Offset (phase) Correction

- Only deviation values measured and stored in the current cycle used
- For a two channel node the smaller value will be taken
- Calculation during network idle time (NIT) of **every** communication cycle, value may be negative
- Offset correction value calculated in even cycles used for error checking only
- Checked against limit values (violation: "NORMAL_ACTIVE" → "NORMAL_PASSIVE" → "HALT")
- Correction value is an integer number of Microticks
- Correction done in **odd** numbered cycles, distributed over the Macroticks beginning at offset correction start up to cycle end (end of network idle time (NIT)) to shift nodes next start of cycle (Macroticks lengthened / shortened)

Rate (frequency) Correction

- Pairs of deviation values measured and stored in even / odd cycle pair used
- For a two channel node the average of the differences from the two channels is used
- Calculated during network idle time (NIT) of **odd** numbered cycles, value may be negative
- Cluster drift damping is performed using global damping value

FlexRay™ Protocol Controller (E-Ray)

- Checked against limit values
- Correction value is a signed integer number of Microticks
- Distributed over Macroticks comprising the next **even / odd** cycle pair (Macroticks lengthened / shortened)

Synchronization Process

Clock synchronization is performed by means of SYNC Frames. Only preconfigured nodes (sync nodes) are allowed to send SYNC Frames. In a two-channel cluster a sync node has to send its SYNC Frame on both channels.

For synchronization in FlexRay™ the following constraints have to be considered:

- Max. one SYNC Frame per node in one communication cycle
- Max. 15 SYNC Frames per cluster in one communication cycle
- Every node has to use all available SYNC Frames for clock synchronization
- Minimum of two sync nodes required for clock synchronization and startup

For clock synchronization the time difference between expected and observed arrival time of SYNC Frames received during the static segment, valid on both channels (two-channel cluster), is measured. The calculation of correction terms is done during network idle time (NIT) (offset: every cycle, rate: odd cycle) by using a FTA / FTM algorithm. For details see FlexRay™ protocol specification v2.1, chapter 8.

SYNC Frame Transmission

SYNC Frame transmission is only possible from buffer 0 and 1. Message Buffer 1 may be used for SYNC Frame transmission in case that SYNC Frames should have different payloads on the two channels. In this case bit **MRC.SPLM** has to be programmed to 1.

Message Buffers used for SYNC Frame transmission have to be configured with the key slot ID and can be (re)configured in “DEFAULT_CONFIG” or “CONFIG” state only. For nodes transmitting SYNC Frames **SUCC1.TXSY** must be set to 1.

25.6.3.4 External Clock Synchronization

During normal operation, independent clusters can drift significantly. If synchronous operation across independent clusters is desired, external synchronization is necessary; even though the nodes within each cluster are synchronized. This can be accomplished with synchronous application of host-deduced rate and offset correction terms to the clusters.

- External offset / rate correction value is a signed integer
- External offset / rate correction value is added to calculated offset / rate correction value
- Aggregated offset / rate correction term (external + internal) is not checked against configured limits

25.6.4 Error Handling

The implemented error handling concept is intended to ensure that in case of a lower layer protocol error in a single node communication between non-affected nodes can be maintained. In some cases, higher layer program command activity is required for the Communication Controller to resume normal operation. A change of the error handling state will set bit **EIR.PEMC** in the Error Service Request Register and may trigger an service request to the Host if enabled. The actual error mode is signalled by **CCEV.ERRM** in the Communication Controller Error Vector register.

Table 25-10 Error Modes of the POC (Degradation Model)

Error Mode	Activity
ACTIVE (green)	Full operation , State: "NORMAL_ACTIVE" The Communication Controller is fully synchronized and supports the cluster wide clock synchronization. The host is informed of any error condition(s) or status change by interrupt (if enabled) or by reading the error and status interrupt flags from registers EIR and SIR .
PASSIVE (yellow)	Reduced operation , State: "NORMAL_PASSIVE", Communication Controller self rescue allowed The Communication Controller stops transmitting Frames and symbols, but received Frames are still processed. Clock synchronization mechanisms are continued based on received Frames. No active contribution to the cluster wide clock synchronization. The host is informed of any error condition(s) or status change by interrupt (if enabled) or by reading the error and status interrupt flags from registers EIR and SIR .
COMM_HALT (red)	Operation halted , State: "HALT", Communication Controller self rescue not allowed The Communication Controller stops Frame and symbol processing, clock synchronization processing, and the Macro-tick generation. The host has still access to error and status information by reading the error and status interrupt flags from registers EIR and SIR . The bus drivers are disabled.

25.6.4.1 Clock Correction Failed Counter

When the Clock Correction Failed Counter reaches the maximum "without clock correction passive" limit defined by **SUCC3.WCP**, the POC transits from "NORMAL_ACTIVE" to "NORMAL_PASSIVE" state. When it reaches the "maximum without clock correction fatal" limit defined by **SUCC3.WCF**, it transits "NORMAL_ACTIVE" or "NORMAL_PASSIVE" to the "HALT" state. Both limits are defined in the SUC Configuration Register 3.

FlexRay™ Protocol Controller (E-Ray)

The Clock Correction Failed Counter **CCEV.CCFC** allows the Host to monitor the duration of the inability of a node to compute clock correction terms after the Communication Controller passed protocol startup phase. It will be incremented by one at the end of any **odd** numbered communication cycle where either the Missing Offset Correction signal **SFS.MOCS** nor the Missing Rate Correction signal **SFS.MRCS** flag is set. The two flags are located in the SYNC Frame Status register, while the Clock Correction Failed Counter is located in the Communication Controller Error Vector register.

The Clock Correction Failed Counter is reset to zero at the end of an **odd** communication cycle if neither the Missing Offset Correction signal **SFS.MOCS** nor the Missing Rate Correction signal **SFS.MRCS** flag is set.

The Clock Correction Failed Counter stops incrementing when the “maximum without clock correction fatal” value **SUCC3.WCF** as defined in the SUC Configuration Register 3 is reached (i.e. incrementing the counter at its maximum value will not cause it to “wraparound” back to zero). The Clock Correction Failed Counter is initialized to zero when the Communication Controller enters “READY” state or when “NORMAL_ACTIVE” state is entered.

25.6.4.2 Passive to Active Counter

The passive to active counter controls the transition of the POC from “NORMAL_PASSIVE” to “NORMAL_ACTIVE” state. **SUCC1.PTA** in the SUC Configuration Register 1 defines the number of consecutive even / odd cycle pairs that must have valid clock correction terms before the Communication Controller is allowed to transit from “NORMAL_PASSIVE” to “NORMAL_ACTIVE” state. If **SUCC1.PTA** is reset to zero the Communication Controller is not allowed to transit from “NORMAL_PASSIVE” to “NORMAL_ACTIVE” state.

25.6.4.3 HALT Command

In case the Host wants to stop FlexRay™ communication of the local node it can bring the Communication Controller into “HALT” state by asserting the HALT command. This can be done by writing **SUCC1.CMD** = 0110_B in the SUC Configuration Register 1. When called in “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state the POC transits to “HALT” state at the end of the current cycle. When called in any other state **SUCC1.CMD** will be reset to 0000_B = “COMMAND_NOT_ACCEPTED” and bit **EIR.CNA** in the Error Service Request Register is set to 1. If enabled an service request to the Host is generated.

25.6.4.4 FREEZE Command

In case the Host detects a severe error condition it can bring the Communication Controller into “HALT” state by asserting the FREEZE command. This can be done by writing **SUCC1.CMD** = 0111_B in the SUC Configuration Register 1. The FREEZE

FlexRay™ Protocol Controller (E-Ray)

command triggers the entry of the "HALT" state immediately regardless of the actual POC state.

The POC state from which the transition to HALT state took place can be read from **CCSV.PSL**.

25.6.5 Communication Controller States

This chapter introduces the states of the Communication Controller.

25.6.5.1 Communication Controller State Diagram

State transitions are controlled by external the application reset or RXDA/B, by the POC state machine, and by the CHI Command Vector **SUCC1.CMD** located in the SUC Configuration Register 1.

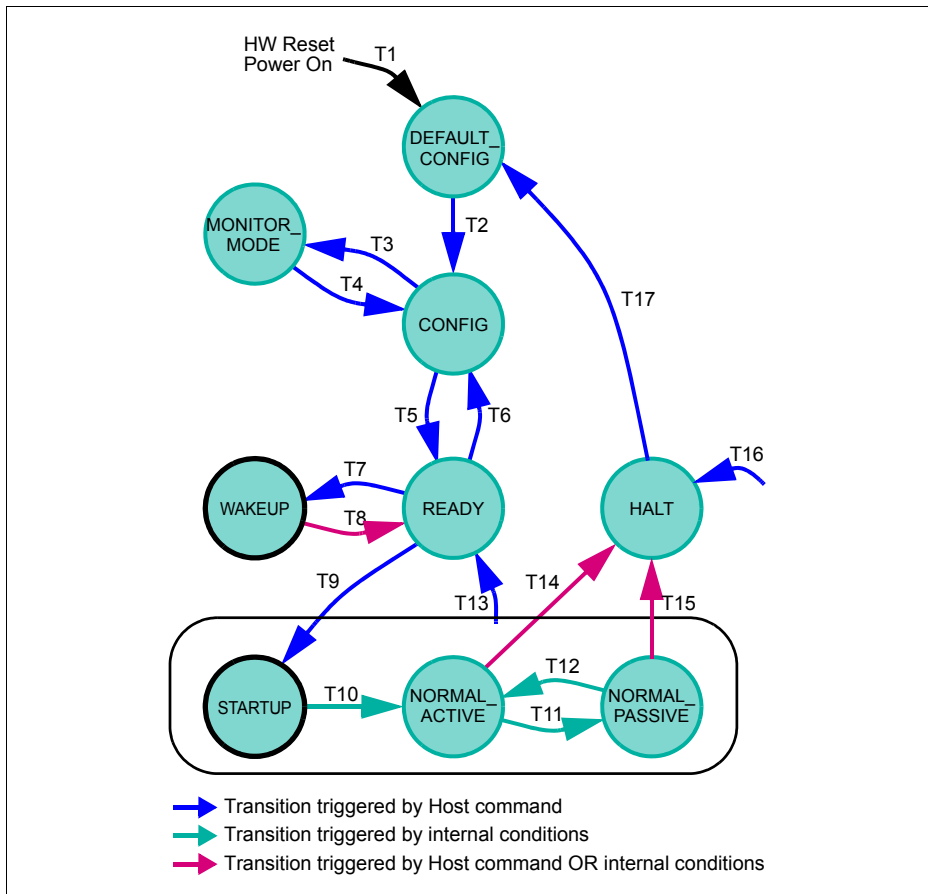


Figure 25-6 Overall State Diagram of E-Ray Communication Controller

The Communication Controller exits from all states to “HALT” state after application of the FREEZE command (SUCC1.CMD = 0111_B).

Table 25-11 State Transitions of E-Ray Overall State Machine

T#	Condition	From	To
1	application reset	HW Reset	DEFAULT_CONFIG
2	Command CONFIG, SUCC1.CMD = 0001 _B	DEFAULT_CONFIG	CONFIG
3	Unlock sequence followed by command MONITOR_MODE, SUCC1.CMD = 1011 _B	CONFIG	MONITOR_MODE
4	Command CONFIG, SUCC1.CMD = 0001 _B	MONITOR_MODE	CONFIG
5	Unlock sequence followed by command READY, SUCC1.CMD = 0010 _B	CONFIG	READY
6	Command CONFIG, SUCC1.CMD = 0001 _B	READY	CONFIG
7	Command WAKEUP, SUCC1.CMD = 0011 _B	READY	WAKEUP
8	Complete, non-aborted transmission of wakeup pattern OR received WUP OR received Frame Header OR command READY, SUCC1.CMD = 0010 _B	WAKEUP	READY
9	Command RUN, SUCC1.CMD = 0100 _B	READY	STARTUP
10	Successful startup	STARTUP	NORMAL_ACTIVE
11	Clock Correction Failed counter reached Maximum Without Clock Correction Passive limit configured by WCP in SUC Configuration Register 3	NORMAL_ACTIVE	NORMAL_PASSIVE
12	Number of valid correction terms reached the Passive to Active limit configured by PTA in SUC Configuration Register 1	NORMAL_PASSIVE	NORMAL_ACTIVE
13	Command READY, SUCC1.CMD = 0010 _B	STARTUP, NORMAL_ACTIVE, NORMAL_PASSIVE	READY

FlexRay™ Protocol Controller (E-Ray)
Table 25-11 State Transitions of E-Ray Overall State Machine (cont'd)

T#	Condition	From	To
14	Clock Correction Failed counter reached Maximum Without Clock Correction Fatal limit configured by WCF in SUC Configuration Register 3 AND bit HCSE in the SUC Configuration Register 1 set to 1 OR command HALT, SUCC1.CMD = 0110 _B	NORMAL_ACTIVE	HALT
15	Clock Correction Failed counter reached Maximum Without Clock Correction Fatal limit configured by WCF in SUC Configuration Register 3 AND bit HCSE in the SUC Configuration Register 1 set to 1 OR command HALT, SUCC1.CMD = 0110 _B	NORMAL_PASSIVE	HALT
16	Command FREEZE, SUCC1.CMD = 0111 _B	All States	HALT
17	Command CONFIG, SUCC1.CMD = 0001 _B	HALT	DEFAULT_CONFIG

25.6.5.2 DEFAULT_CONFIG State

In “DEFAULT_CONFIG” state, the Communication Controller is stopped. All configuration registers are accessible and the pins to the physical layer are in their inactive state.

The Communication Controller enters this state

- When leaving application reset
- When exiting from “HALT” state

To leave “DEFAULT_CONFIG” state the Host has to write SUCC1.CMD = 0001_B in the SUC Configuration Register 1. The Communication Controller transits to “CONFIG” state.

CONFIG State

In “CONFIG” state, the Communication Controller is stopped. All configuration registers are accessible and the pins to the physical layer are in their inactive state. This state is used to initialize the Communication Controller configuration.

FlexRay™ Protocol Controller (E-Ray)

The Communication Controller enters this state

- When exiting from “DEFAULT_CONFIG” state
- When exiting from “MONITOR_MODE” or “READY” state

When the state has been entered via “HALT” and “DEFAULT_CONFIG” state, the Host can analyze status information and configuration. Before leaving “CONFIG” state the Host has to assure that the configuration is fault-free.

To leave “CONFIG” state, the Host has to perform the unlock sequence as described on **“LCK” on Page 25-30**. Directly after unlocking the “CONFIG” state the Host has to write **SUCC1.CMD** in the SUC Configuration Register 1 to enter the next state.

Internal counters and the Communication Controller status flags are reset when the Communication Controller leaves “CONFIG”.

*Note: The Message Buffer Status Registers (**MHDS**, **TXRQ1** to **TXRQ4**, **NDAT1** to **NDAT4**, **MBSC1** to **MBSC4**) and status data stored in the Message RAM and are not affected by the transition of the POC from “CONFIG” to “READY” state.*

When the Communication Controller is in “CONFIG” state it is also possible to bring the Communication Controller into a power saving mode by halting the module clocks (f_{SCLK} , f_{CLC_ERAY}). To do this the Host has to assure that all Message RAM transfers have finished before turning off the clocks.

25.6.5.3 MONITOR_MODE

After unlocking “CONFIG” state and writing **SUCC1.CMD** = 0011_B the Communication Controller enters “MONITOR_MODE”. In this mode the Communication Controller is able to receive FlexRay™ Frames and to detect wakeup pattern. The temporal integrity of received Frames is not checked, and therefore cycle counter filtering is not supported. It is not possible to distinguish between static and dynamic frames, because limited functions in Monitor Mode (FRF.RSS will be ignored, filtering not functional). This mode can be used for debugging purposes in case e.g. that startup of a FlexRay™ network fails. After writing **SUCC1.CMD** = 0001_B the Communication Controller transits back to “CONFIG” state.

In MONITOR_MODE the pick first valid mechanism is disabled. This means that a receive Message Buffer may only be configured to receive on one channel. Received Frames are stored into Message Buffers according to Frame ID and receive channel. NULL Frames are handled like Data Frames. After Frame reception only status bits **MBS.VFRA**, **MBS**, **MBS.MLST**, **MBS.RCIS**, **MBS.SFIS**, **MBS.SYNS**, **MBS.NFIS**, **MBS.PPIS**, **MBS.RESS** have valid value.

In “MONITOR_MODE” the Communication Controller is not able to distinguish between CAS and MTS symbols. In case one of these symbols is received on one or both of the two channels, the flags **SIR.MTSA** resp. **SIR.MTSB** are set. **SIR.CAS** has no function in “MONITOR_MODE”.

25.6.5.4 READY State

After unlocking “CONFIG” state and writing **SUCC1.CMD** = 0010_B the Communication Controller enters “READY” state. From this state the Communication Controller can transit to WAKEUP state and perform a cluster wakeup or to “STARTUP” state to perform a coldstart or to integrate into a running communication.

The Communication Controller enters this state

- When exiting from “CONFIG”, “WAKEUP”, “STARTUP”, “NORMAL_ACTIVE”, or “NORMAL_PASSIVE” state by writing **SUCC1.CMD** = 0010_B (READY command).

The Communication Controller exits from this state

- To “CONFIG” state by writing **SUCC1.CMD** = 0001_B (CONFIG command)
- To “WAKEUP” state by writing **SUCC1.CMD** = 0011_B (WAKEUP command)
- To “STARTUP” state by writing **SUCC1.CMD** = 0100_B (RUN command)

Internal counters and the Communication Controller status flags are reset when the Communication Controller enters “STARTUP” state.

*Note: Status bits **MHDS**, registers **TXRQ1** to **TXRQ4**, and status data stored in the Message RAM are not affected by the transition of the POC from “READY” to “STARTUP” state.*

25.6.5.5 WAKEUP State

The description below is intended to help configuring wakeup for the E-Ray IP-module. A detailed description of the wakeup procedure together with the respective SDL diagrams can be found in the FlexRay™ protocol specification v2.1, section 7.1.

The Communication Controller enters this state

- When exiting from “READY” state by writing **SUCC1.CMD** = 0011_B (WAKEUP command).

The Communication Controller exits from this state to “READY” state

- After complete non-aborted transmission of wakeup pattern
- After WUP reception
- After detecting a WUP collision
- After reception of a Frame Header
- By writing **SUCC1.CMD** = 0010_B (READY command)

The cluster wakeup must precede the communication startup in order to ensure that all mechanisms defined for the startup work properly. The minimum requirement for a cluster wakeup is that all bus drivers are supplied with power. A bus driver has the ability to wake up the other components of its node when it receives a wakeup pattern on its channel. At least one node in the cluster needs an **external** wakeup source.

The Host completely controls the wakeup procedure. It is informed about the state of the cluster by the bus driver and the Communication Controller and configures bus guardian

FlexRay™ Protocol Controller (E-Ray)

(if available) and Communication Controller to perform the cluster wakeup. The Communication Controller provides to the Host the ability to transmit a special wakeup pattern on each of its available channels separately. The Communication Controller needs to recognize the wakeup pattern only during “WAKEUP” state.

Wakeup may be performed on only one channel at a time. The Host has to configure the wakeup channel while the Communication Controller is in “CONFIG” state by writing bit **SUCC1.WUCS** in the SUC Configuration Register 1. The Communication Controller ensures that ongoing communication on this channel is not disturbed. The Communication Controller cannot guarantee that all nodes connected to the configured channel awake upon the transmission of the wakeup pattern, since these nodes cannot give feedback until the startup phase. The wakeup procedure enables single-channel devices in a two-channel system to trigger the wakeup, by only transmitting the wakeup pattern on the single channel to which they are connected. Any coldstart node that deems a system startup necessary will then wake the remaining channel before initiating communication startup.

The wakeup procedure tolerates any number of nodes simultaneously trying to wakeup a single channel and resolves this situation such that only one node transmits the pattern. Additionally the wakeup pattern is collision resilient, so even in the presence of a fault causing two nodes to simultaneously transmit a wakeup pattern, the resulting collided signal can still wake the other nodes.

After wakeup the Communication Controller returns to “READY” state and signals the change of the wakeup status to the Host by setting bit **SIR.WST** in the Status Service Request Register. The wakeup status vector can be read from the Communication Controller Status Vector register **CCSV.WSV**. If a valid wakeup pattern was received also either flag **SIR.WUPA** or flag **SIR.WUPB** in the Status Service Request Register is set.

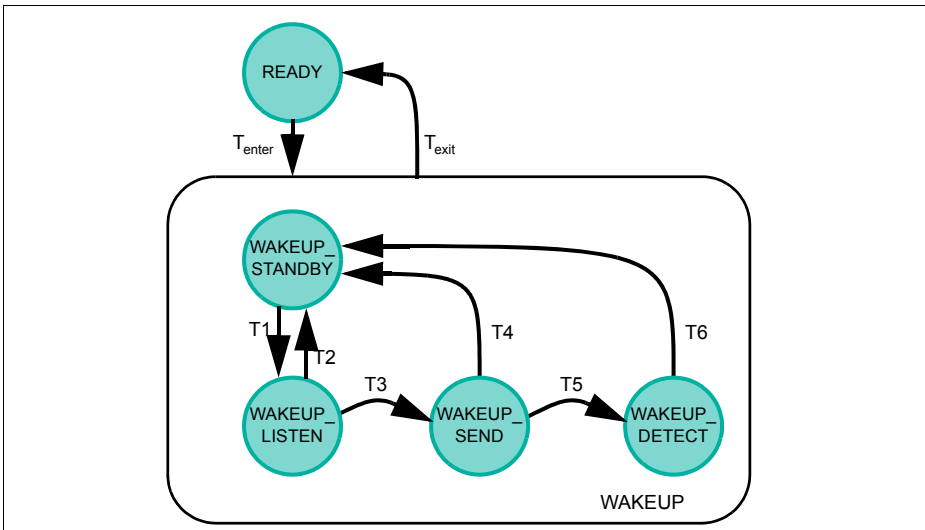


Figure 25-7 Structure of POC State WAKEUP

Table 25-12 State Transitions WAKEUP

T#	Condition	From	To
enter	Host commands change to "WAKEUP" state by writing SUCC1.CMD = 0011 _B (WAKEUP command)	READY	WAKEUP
1	CHI command WAKEUP triggers wakeup FSM to transit to "WAKEUP_LISTEN" state	WAKEUP_STANDBY	WAKEUP_LISTEN
2	Received WUP on wakeup channel selected by flag SUCC1.WUCS in the SUC Configuration Register 1 OR Frame Header on either available channel	WAKEUP_LISTEN	WAKEUP_STANDBY
3	Timer event	WAKEUP_LISTEN	WAKEUP_SEND
4	Complete, non-aborted transmission of wakeup pattern	WAKEUP_SEND	WAKEUP_STANDBY

FlexRay™ Protocol Controller (E-Ray)
Table 25-12 State Transitions WAKEUP (cont'd)

T#	Condition	From	To
5	Collision detected	WAKEUP_SEND	WAKEUP_DETECT
6	Wakeup timer expired OR WUP detected on wakeup channel selected by flag SUCC1.WUCS in the SUC Configuration Register 1 OR Frame Header received on either available channel	WAKEUP_DETECT	WAKEUP_STANDBY
exit	Wakeup completed (after T2 or T4 or T6) OR Host commands change to “READY” state by writing SUCC1.CMD = 0010 _B (READY command). This command also resets the wakeup FSM to “WAKEUP_STANDBY” state	WAKEUP	READY

The “WAKEUP_LISTEN” state is controlled by the wakeup timer and the wakeup noise timer. The two timers are controlled by the parameters listen time-out **SUCC2.LT** and listen time-out noise **SUCC2.LTN**. Both values can be configured in the SUC Configuration Register 2. listen time-out enables a fast cluster wakeup in case of a noise free environment, while listen time-out noise enables wakeup under more difficult conditions regarding noise interference.

In “WAKEUP_SEND” state the Communication Controller transmits the wakeup pattern on the configured channel and checks for collisions. After return from wakeup the Host has to bring the Communication Controller into “STARTUP” state by CHI command RUN.

In “WAKEUP_DETECT” state the Communication Controller attempts to identify the reason for the wakeup collision detected in “WAKEUP_SEND” state. The monitoring is bounded by the expiration of listen time-out as configured by **SUCC2.LT** in the SUC Configuration Register 2. Either the detection of a wakeup pattern indicating a wakeup attempt by another node or the reception of a Frame Header indication existing communication, causes the direct transition to “READY” state. Otherwise WAKEUP_DETECT is left after expiration of listen time-out; in this case the reason for wakeup collision is unknown.

The Host has to be aware of possible failures of the wakeup and act accordingly. It is advisable to delay any potential startup attempt of the node having instigated the wakeup by the minimal time it takes another coldstart node to become awake and to be configured.

The FlexRay™ Protocol Specification v2.1 recommends that two different Communication Controllers shall awake the two channels.

Host activities

The host must coordinate the wakeup of the two channels and must decide whether, or not, to wake a specific channel. The sending of the wakeup pattern is initiated by the Host and generated by the Communication Controller. The wakeup pattern is detected by the remote BDs and signalled to their local Hosts.

Wakeup procedure controlled by Host (single-channel wakeup):

- Configure the Communication Controller in “CONFIG” state
 - Select wakeup channel by programming bit **SUCC1.WUCS**
- Check local BDs whether a WUP was received
- Activate BD of selected wakeup channel
- Command Communication Controller to start wakeup on the configured channel by writing **SUCC1.CMD** = 0011_B
 - Communication Controller enters “WAKEUP”
 - Communication Controller returns to “READY” state and signals status of wakeup attempt to Host
- Wait predefined time to allow the other nodes to wakeup and configure themselves
- Coldstart node: wait for WUP on the other channel
 - In a dual channel cluster wait for WUP on the other channel
 - Reset coldstart inhibit flag **CCSV.CSI** by writing **SUCC1.CMD** = 1001_B (ALLOW_COLDSTART command))
- Reset Coldstart Inhibit flag **CCSV.CSI** in the CCSV register by writing **SUCC1.CMD** = 1001_B (ALLOW_COLDSTART command), coldstart node only
- Command Communication Controller to enter startup by writing **SUCC1.CMD** = 0100_B (RUN command)

Wakeup procedure triggered by BD:

- Wakeup recognized by BD
- BD triggers power-up of Host (if required)
- BD signals wakeup event to Host
- Host configures its local Communication Controller
- If necessary Host commands wakeup of second channel and waits predefined time to allow the other nodes to wakeup and configure themselves
- Host commands Communication Controller to enter “STARTUP” state by writing **SUCC1.CMD** = 0100_B (RUN command)

Wakeup pattern (WUP)

The wakeup pattern is composed of at least two wakeup symbols (WUS). Wakeup symbol and wakeup pattern are configured by the PRT Configuration Registers **PRTC1** and **PRTC2**.

- Single channel wakeup, wakeup symbol may not be sent on both channels at the same time

FlexRay™ Protocol Controller (E-Ray)

- Wakeup symbol collision resilient for up to two sending nodes (two overlapping wakeup symbols still recognizable)
- Wakeup symbol must be configured identical in all nodes of a cluster
- Wakeup symbol transmit low time configured by **PRTC2.TXL**
- Wakeup symbol idle time used to listen for activity on the bus, configured by **PRTC2.TXI**
- A wakeup pattern composed of at least two Tx-wakeup symbols needed for wakeup
- Number of repetitions configurable by **PRTC1.RWP** (2 to 63 repetitions)
- Wakeup symbol receive window length configured by **PRTC1.RXW**
- Wakeup symbol receive low time configured by **PRTC2.RXL**
- Wakeup symbol receive idle time configured by **PRTC2.RXI**

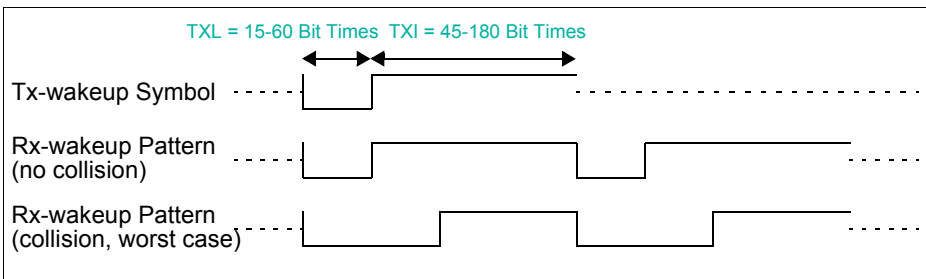


Figure 25-8 Timing of Wakeup Pattern

25.6.5.6 STARTUP State

The description below is intended to help configuring startup for the E-Ray IP-module. A detailed description of the startup procedure together with the respective SDL diagrams can be found in the FlexRay™ protocol specification v2.1, section 7.2.

Any node entering “STARTUP” state that has coldstart capability should assure that both channels attached have been awakened before initiating coldstart.

It cannot be assumed that all nodes and stars need the same amount of time to become completely awake and to be configured. Since at least two nodes are necessary to start up the cluster communication, it is advisable to delay any potential startup attempt of the node having instigated the wakeup by the minimal amount of time it takes another coldstart node to become awake, to be configured and to enter startup. It may require several hundred milliseconds (depending on the hardware used) before all nodes and stars are completely awakened and configured.

Startup is performed on all channels synchronously. During startup, a node only transmits startup Frames.

A fault-tolerant, distributed startup strategy is specified for initial synchronization of all nodes. In general, a node may enter “NORMAL_ACTIVE” state via (see **Figure 25-9**):

FlexRay™ Protocol Controller (E-Ray)

- Coldstart path initiating the schedule synchronization (leading coldstart node)
- Coldstart path joining other coldstart nodes (following coldstart node)
- Integration path integrating into an existing communication schedule (all other nodes)

A coldstart attempt begins with the transmission of a collision avoidance symbol (CAS). Only a coldstart node that had transmitted the CAS transmits Frames in the first four cycles after the CAS, it is then joined firstly by the other coldstart nodes and afterwards by all other nodes.

A coldstart node has the Transmit SYNC Frame in Key Slot bits **SUCC1.TXST** and **SUCC1.TXSY** in the SUC Configuration Register 1 set to 1. The Message Buffer 0 holds the key slot ID which defines the slot number where the Startup Frame is send. In the Frame Header of the Startup Frame the Startup Frame indicator bit is set.

In clusters consisting of three or more nodes, at least three nodes shall be configured to be coldstart nodes. In clusters consisting of two nodes, both nodes must be coldstart nodes. At least two fault-free coldstart nodes are necessary for the cluster to startup.

Each Startup Frame must also be a SYNC Frame; therefore each coldstart node will also be a sync node. The number of coldstart attempts is configured by **SUCC1.CSA** in the SUC Configuration Register 1.

A non-coldstart node requires at least two startup Frames from distinct nodes for integration. It may start integration before the coldstart nodes have finished their startup. It will not finish its startup until at least two coldstart nodes have finished their startup.

Both non-coldstart nodes and coldstart nodes start passive integration via the integration path as soon as they receive SYNC Frames from which to derive the TDMA schedule information. During integration the node has to adapt its own clock to the global clock (rate and offset) and has to make its cycle time consistent with the global schedule observable at the network. Afterwards, these settings are checked for consistency with all available network nodes. The node can only leave the integration phase and actively participate in communication when these checks are passed.

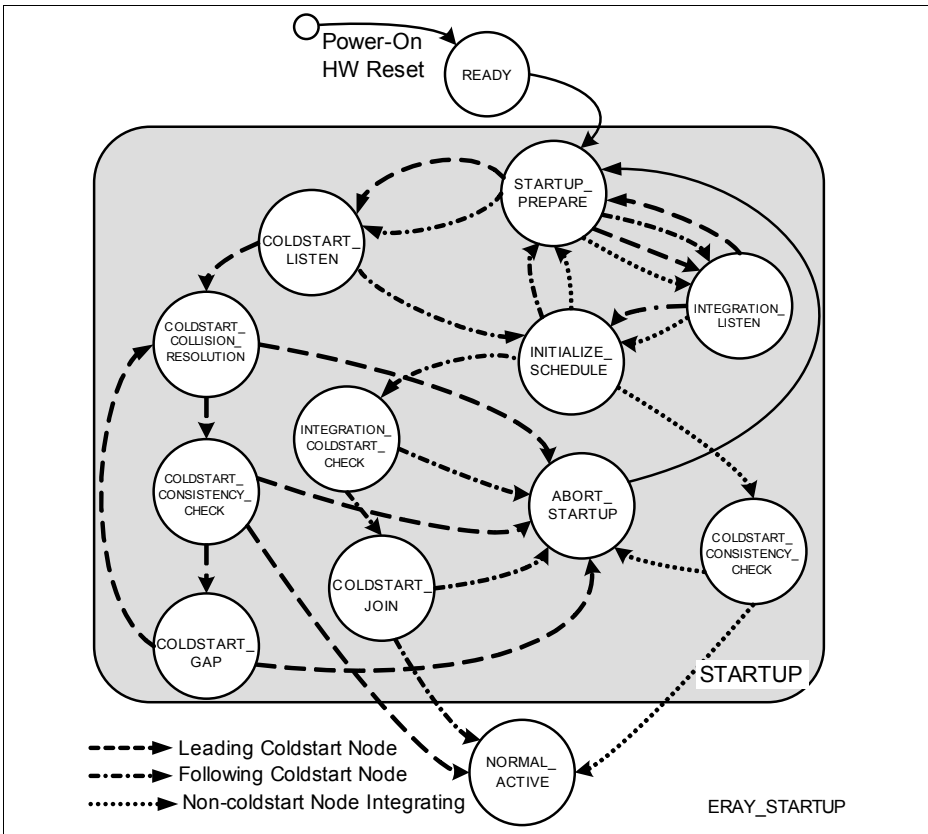


Figure 25-9 State Diagram Time-Triggered Startup

Coldstart Inhibit Mode

In coldstart inhibit mode the node is prevented from initializing the TDMA communication schedule. If bit **CCSV.CSI** in the Communication Controller Status Vector register is set, the node is not allowed to initialize the cluster communication, i.e. entering the coldstart path is prohibited. The node is allowed to integrate to a running cluster or to transmit startup Frames after another coldstart node started the initialization of the cluster communication.

The coldstart inhibit bit **CCSV.CSI** is set whenever the POC enters “READY” state. The bit has to be cleared under control of the Host by CHI command **ALLOW_COLDSTART (SUCC1.CMD = 1001_B)**

25.6.5.7 Startup Time-outs

The Communication Controller supplies two different Microtick timers supporting two time-out values, startup time-out and startup noise time-out. The two timers are reset when the Communication Controller enters the “COLDSTART_LISTEN” state. The expiration of either of these timers causes the node to leave the initial sensing phase (“COLDSTART_LISTEN” state) with the intention of starting up communication.

*Note: The startup and startup noise timers are identical with the wakeup and wakeup noise timers and use the same configuration values **SUCC2.LT** and **SUCC2.LTN** from the SUC Configuration Register 2.*

Startup Time-out

The startup time-out limits the listen time used by a node to determine if there is already communication between other nodes or at least one coldstart node actively requesting the integration of others.

The startup timer is configured by programming **SUCC2.LT** (pdListenTimeout) in the SUC Configuration Register 2.

The startup timer is restarted upon:

- Entering the “COLDSTART_LISTEN” state
- Both channels reaching idle state while in “COLDSTART_LISTEN” state

The startup timer is stopped:

- If communication channel activity is detected on one of the configured channels while the node is in the “COLDSTART_LISTEN” state
- When the “COLDSTART_LISTEN” state is left

Once the startup time-out expires, neither an overflow nor a cyclic restart of the timer is performed. The timer status is kept for further processing by the startup state machine.

Startup Noise Time-out

At the same time the startup timer is started for the first time (transition from “STARTUP_PREPARE” state to “COLDSTART_LISTEN” state), the startup noise timer is started. This additional time-out is used to improve reliability of the startup procedure in the presence of noise.

The startup noise timer is configured by programming **SUCC2.LTN** (gListenNoise - 1) in the SUC Configuration Register 2 (see “**SUC Configuration Register 2 (SUCC2)**” on **Page 25-87**).

The startup noise time-out is:

$\text{pdListenTimeout} \cdot \text{gListenNoise} = \text{SUCC2.LT} \cdot (\text{SUCC2.LTN} + 1)$

The startup noise timer is restarted upon:

- Entering the “COLDSTART_LISTEN” state
- Reception of correctly decoded Headers or CAS symbols while the node is in “COLDSTART_LISTEN” state

The startup noise timer is stopped when the “COLDSTART_LISTEN” state is left.

Once the startup noise time-out expires, neither an overflow nor a cyclic restart of the timer is performed. The status is kept for further processing by the startup state machine. Since the startup noise timer won't be restarted when random channel activity is sensed, this time-out defines the fall-back solution that guarantees that a node will try to start up the communication cluster even in the presence of noise.

25.6.5.8 Path of leading Coldstart Node (initiating coldstart)

When a coldstart node enters “COLDSTART_LISTEN”, it listens to its attached channels.

If no communication is detected, the node enters the “COLDSTART_COLLISION_RESOLUTION” state and commences a coldstart attempt. The initial transmission of a CAS symbol is succeeded by the first regular cycle. This cycle has the number zero.

From cycle zero on, the node transmits its startup Frame. Since each coldstart node is allowed to perform a coldstart attempt, it may occur that several nodes simultaneously transmit the CAS symbol and enter the coldstart path. This situation is resolved during the first four cycles after CAS transmission.

As soon as a node that initiates a coldstart attempt receives a CAS symbol or a Frame Header during these four cycles, it re-enters the “COLDSTART_LISTEN” state. Thereby, only one node remains in this path. In cycle four, other coldstart nodes begin to transmit their startup Frames.

After four cycles in “COLDSTART_COLLISION_RESOLUTION” state, the node that initiated the coldstart enters the “COLDSTART_CONSISTENCY_CHECK” state. It collects all startup Frames from cycle four and five and performs the clock correction. If the clock correction does not deliver any errors and it has received at least one valid Startup Frame pair, the node leaves “COLDSTART_CONSISTENCY_CHECK” and enters “NORMAL_ACTIVE” state.

The number of coldstart attempts that a node is allowed to perform is configured by **SUCC1.CSA** in the SUC Configuration Register 1. The number of remaining coldstarts attempts **CCSV.RCA** can be read from Communication Controller Status Vector register. The number of remaining attempts is reduced by one for each attempted coldstart. A node may enter the “COLDSTART_LISTEN” state only if this value is larger than one and it may enter the “COLDSTART_COLLISION_RESOLUTION” state only if this value is larger than zero. If the number of coldstart attempts is one, coldstart is inhibited but integration is still possible.

FlexRay™ Protocol Controller (E-Ray)**Path of following Coldstart Node (responding to leading Coldstart Node)**

When a coldstart node enters the “COLDSTART_LISTEN” state, it tries to receive a valid pair of startup Frames to derive its schedule and clock correction from the leading coldstart node.

As soon as a valid Startup Frame has been received the “INITIALIZE_SCHEDULE” state is entered. If the clock synchronization can successfully receive a matching second valid Startup Frame and can derive a schedule from this startup Frames, the “INTEGRATION_COLDSTART_CHECK” state is entered.

In “INTEGRATION_COLDSTART_CHECK” state it is assured that the clock correction can be performed correctly and that the coldstart node from which this node has initialized its schedule is still available. The node collects all SYNC Frames and performs clock correction in the following double-cycle. If clock correction does not signal any errors and if the node continues to receive sufficient Frames from the same node it has integrated on, the “COLDSTART_JOIN” state is entered.

In “COLDSTART_JOIN” state integrating coldstart nodes begin to transmit their own startup Frames. Thereby the node that initiated the coldstart and the nodes joining it can check if their schedules agree to each other. If for the following three cycles the clock correction does not signal errors and at least one other coldstart node is visible, the node leaves “COLDSTART_JOIN” state and enters “NORMAL_ACTIVE” state. Thereby it leaves “STARTUP” at least one cycle after the node that initiated the coldstart.

Path of Non-coldstart Node

When a non-coldstart node enters the INTEGRATION_LISTEN state, it listens to its attached channels and tries to receive FlexRay™ Frames.

As soon as a valid Startup Frame has been received the “INITIALIZE_SCHEDULE” state is entered. If the clock synchronization can successfully receive a matching second valid Startup Frame and derive a schedule from this, the INTEGRATION_CONSISTENCY_CHECK state is entered.

In “INTEGRATION_CONSISTENCY_CHECK” state it is verified that the clock correction can be performed correctly and that enough coldstart nodes (at least 2) send startup Frames that agree to the nodes own schedule. Clock correction is activated, and if any errors are signalled, the integration attempt is aborted.

During the first even cycle in this state, either two valid startup Frames or the Startup Frame of the node that this node has integrated on must be received; otherwise the node aborts the integration attempt.

During the first double-cycle in this state, either two valid Startup Frame pairs or the Startup Frame pair of the node that this node has integrated on must be received; otherwise the node aborts the integration attempt.

FlexRay™ Protocol Controller (E-Ray)

If after the first double-cycle less than two valid startup Frames are received within an even cycle, or less than two valid Startup Frame pairs are received within a double-cycle, the startup attempt is aborted.

Nodes in this state need to see two valid Startup Frame pairs for two consecutive double-cycles each to be allowed to leave STARTUP and enter NORMAL_OPERATION. Consequently, they leave startup at least one double-cycle after the node that initiated the coldstart and only at the end of a cycle with an odd cycle number.

25.6.5.9 NORMAL_ACTIVE State

As soon as the node that transmitted the first CAS symbol (resolving the potential access conflict and entering “STARTUP” via coldstart path) and one additional node have entered the “NORMAL_ACTIVE” state, the startup phase for the cluster has finished. In the “NORMAL_ACTIVE” state, all configured messages are scheduled for transmission. This includes all Data Frames as well as the SYNC Frames. Rate and offset measurement is started in all even cycles (even/odd cycle pairs required).

In “NORMAL_ACTIVE” state the Communication Controller supports regular communication functions

- The Communication Controller performs transmissions and reception on the FlexRay™ bus as configured
- Clock synchronization is running
- The Host interface is operational

The Communication Controller exits from that state to

- “HALT” state by writing **SUCC1.CMD** = 0110_B (HALT command, at the end of the current cycle)
- “HALT” state by writing **SUCC1.CMD** = 0111_B (FREEZE command, immediately)
- “HALT” state due to change of the error state from “ACTIVE” to “COMM_HALT”
- “NORMAL_PASSIVE” state due to change of the error state from “ACTIVE” to “PASSIVE”
- “READY” state by writing **SUCC1.CMD** = 0010_B (READY command)

25.6.5.10 NORMAL_PASSIVE State

“NORMAL_PASSIVE” state is entered from “NORMAL_ACTIVE” state when the error state changes from ACTIVE (green) to PASSIVE (yellow).

In “NORMAL_PASSIVE” state, the node is able to receive all Frames (node is fully synchronized and performs clock synchronization). In comparison to the “NORMAL_ACTIVE” state the node does not actively participate in communication, i.e. neither symbols nor Frames are transmitted.

FlexRay™ Protocol Controller (E-Ray)

In “NORMAL_PASSIVE” state

- The Communication Controller performs reception on the FlexRay™ bus
- The Communication Controller does not transmit any Frames or symbols on the FlexRay™ bus
- Clock synchronization is running
- The Host interface is operational

The Communication Controller exits from this state to

- “HALT” state by writing **SUCC1.CMD** = 0110_B (HALT command, at the end of the current cycle)
- “HALT” state by writing **SUCC1.CMD** = 0111_B (FREEZE command, immediately)
- “HALT” state due to change of the error state from “PASSIVE” to “COMM_HALT”
- “NORMAL_ACTIVE” state due to change of the error state from “PASSIVE” to “ACTIVE”. The transition takes place when **CCEV.PTAC** from the Communication Controller Error Vector register equals **SUCC1.PTA** - 1.
- “READY” state by writing **SUCC1.CMD** = 0010_B (READY command)

25.6.5.11 HALT State

In this state all communication (reception and transmission) is stopped.

The Communication Controller enters this state

- By writing **SUCC1.CMD** = 0110_B (HALT command) while the Communication Controller is in “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state
- By writing **SUCC1.CMD** = 0111_B (FREEZE command) from all states
- When exiting from “NORMAL_ACTIVE” state because the clock correction failed counter reached the “maximum without clock correction fatal” limit
- When exiting from “NORMAL_PASSIVE” state because the clock correction failed counter reached the “maximum without clock correction fatal” limit

The Communication Controller exits from this state to “CONFIG” state

- By writing **SUCC1.CMD** = 0001_B (DEFAULT_CONFIG command)

When the Communication Controller enters “HALT” state, all configuration and status data is maintained for analyzing purposes.

When the Host writes **SUCC1.CMD** = 0110_B (HALT command) in the SUC Configuration Register 1 to 1, the Communication Controller sets bit **CCSV.HRQ** in the Communication Controller Status Vector register and enters “HALT” state after the current communication cycle has finished.

When the Host writes **SUCC1.CMD** = 0111_B (FREEZE command) in the SUC Configuration Register to 1, the Communication Controller enters “HALT” state immediately and sets the **CCSV.FSI** bit in the Communication Controller Status Vector register.

FlexRay™ Protocol Controller (E-Ray)

The POC state from which the transition to HALT state took place can be read from **CCSV.PSL**.

25.6.6 Network Management

The accrued Network Management (NM) vector is located in the Network Management Register 1 to Network Management Register 3 (**NMV_x (x = 1-3)**). The Communication Controller performs a logical OR operation over all Network Management (NM) vectors out of all received valid Network Management (NM) Frames with the Payload Preamble Indicator (PPI) bit set. Only a static Frame may be configured to hold Network Management (NM) information. The Communication Controller updates the Network Management (NM) vector at the end of each cycle.

The length of the Network Management (NM) vector can be configured from 0 to 12 byte by NML in the NEM Configuration Register. The Network Management (NM) vector length must be configured identically in all nodes of a cluster.

To configure a transmit buffer to send FlexRay™ Frames with the PPI bit set, the PPIT bit in the Header Section of the respective transmit buffer has to be set via **WRHS1.PPIT**. In addition the Host has to write the Network Management (NM) information to the Data Section of the respective transmit buffer.

The evaluation of the Network Management (NM) vector has to be done by the application running on the Host.

*Note: In case a Message Buffer is configured for transmission / reception of Network Management Frames, the payload length configured in Header 2 of that Message Buffer should be equal or greater than the length of the NM Vector configured by **NEMC.NML**.*

When the Communication Controller transits to “HALT” state, the cycle count is not incremented and therefore the NM Vector is not updated. In this case NMV1 to NMV3 holds the value from the cycle before.

25.6.7 Filtering and Masking

Filtering is done by checking specific fields in a received Frame against the corresponding configuration constants of the valid Message Buffers and the actual slot and cycle counter values (acceptance filtering), or by comparing the configuration constants of the valid Message Buffers against the actual slot and cycle counter values (transmit filtering). A Message Buffer is only updated / transmitted if the required matches occur.

Filtering is done on the following fields:

- Channel ID
- Frame ID
- Cycle Counter

The following filter combinations for acceptance / transmit filtering are allowed:

FlexRay™ Protocol Controller (E-Ray)

- Frame ID + Channel ID
- Frame ID + Channel ID + Cycle Counter

In order to store a received message in a Message Buffer all configured filters must match.

Note: For the FIFO the acceptance filter is configured by the FIFO Rejection Filter and the FIFO Rejection Filter Mask.

A message will be transmitted in the time slot corresponding to the configured Frame ID on the configured channel(s). If cycle counter filtering is enabled the configured cycle filter value must also match.

25.6.7.1 Frame ID Filtering

Every transmit and receive buffer contains a Frame ID stored in the Header Section. This Frame ID is used differently for receive and transmit buffers.

Receive Buffers

A received message is stored in the first receive buffer where the received Frame ID matches the configured Frame ID, provided channel ID and cycle counter criteria are also met.

Transmit Buffers

For transmit buffers the configured Frame ID is used to determine the appropriate slot for message transmission. The Frame will be transmitted in the time slot corresponding to the configured Frame ID, provided channel ID and cycle counter criteria are also met.

25.6.7.2 Channel ID Filtering

There is a 2-bit channel filtering field (CHA, CHB) located in the Header Section of each Message Buffer in the Message RAM. It serves as a filter for receive buffers, and as a control field for transmit buffers (see [Table 25-13](#)).

Table 25-13 Channel Filtering Configuration

CHA	CHB	Transmit Buffer transmit Frame	Receive Buffer store valid receive Frame
1	1	on both channels (static segment only)	received on channel A or B (store first semantically valid Frame, static segment only)
1	0	on channel A	received on channel A
0	1	on channel B	received on channel B
0	0	no transmission	ignore Frame

FlexRay™ Protocol Controller (E-Ray)

Note: If a Message Buffer is configured for the dynamic segment and both bits of the channel filtering field are set to 1, no Frames are transmitted resp. received Frames are ignored (same function as CHA = CHB = 0)

Receive Buffers

Valid received Frames are stored if they are received on the channels specified in the channel filtering field. Only in static segment a receive buffer may be setup for reception on both channels (CHA and CHB set). Other filtering criteria must also be met.

If a valid Header Segment was stored, the respective MBC flag in the Message Buffer Status Changed register is set. If a valid Payload Segment was stored, the respective **NDn (n = 0-31)** to **NDn (n = 96-127)** flag in the New Data **NDAT1** to **NDAT4** register is set. In both cases, if bit **RDHS1.MBI** in the Header Section of the respective Message Buffer is set, the RXI flag in the Status Service Request Register is set to 1. If enabled an service request is generated.

Transmit Buffers

The content of the buffer is transmitted only on the channels specified in the channel filtering field when the Frame ID filtering and cycle counter filtering criteria are also met. Only in static segment a transmit buffer may be setup for transmission on both channels (CHA and CHB set). After transmission has completed, and if bit **WRHS1.MBI** in the Header Section of the respective Message Buffer is set, the TXI flag in the Status Service Request Register is set to 1. If enabled an service request is generated.

25.6.7.3 Cycle Counter Filtering

Cycle counter filtering is based on the notion of a cycle set. For filtering purposes, a match is detected if any one of the elements of the cycle set is matched. The cycle set is defined by the cycle code field in the Header Section of each Message Buffer.

If Message Buffer 0 is configured to hold the startup / SYNC Frame or the single slot Frame by bits TXST, TXSY, and TSM in the SUC Configuration Register 1, cycle counter filtering for Message Buffer 0 should be disabled.

*Note: Sharing of a static time slot via cycle counter filtering between different nodes of a FlexRay™ network is **not** allowed.*

The set of cycle numbers belonging to a cycle set is determined as described in [Table 25-14](#).

Table 25-14 Definition of Cycle Set

Cycle Code	Matching Cycle Counter Values		
000000 _B	all Cycles		
000001 _B	every second Cycle	at (Cycle Count)mod2	= c
00001 _B	every fourth Cycle	at (Cycle Count)mod4	= cc
0001 _B	every eighth Cycle	at (Cycle Count)mod8	= ccc
001 _B	every sixteenth Cycle	at (Cycle Count)mod16	= cccc
01 _B	every thirty-second Cycle	at (Cycle Count)mod32	= ccccc
1 _B	every sixty-fourth Cycle	at (Cycle Count)mod64	= cccccc

Table 25-15 below gives some examples for valid cycle sets to be used for cycle counter filtering:

Table 25-15 Examples for Valid Cycle Sets

Cycle Code	Matching Cycle Counter Values
0000011 _B	1-3-5-7--63 ↓
0000100 _B	0-4-8-12--60 ↓
0001110 _B	6-14-22-30--62 ↓
0011000 _B	8-24-40-56 ↓
0100011 _B	3-35 ↓
1001001 _B	9 ↓

Receive Buffers

The received message is stored only if the received cycle counter matches an element of the receive buffer's cycle set. Channel ID and Frame ID criteria must also be met.

Transmit Buffers

The content of the buffer is transmitted on the configured channels when an element of the cycle set matches the current cycle counter value and the Frame ID matches the slot counter value.

25.6.7.4 FIFO Filtering

For FIFO filtering there is one rejection filter and one rejection filter mask available. The FIFO rejection filter consists of 20 bits for **Channel** (2 bits), **Frame ID** (11 bits), and **Cycle Code** (7 bits). Rejection filter and rejection filter mask can be configured in

FlexRay™ Protocol Controller (E-Ray)

DEFAULT_CONGIF or “CONFIG” state only. The filter configuration in the Header Sections of Message Buffers belonging to the FIFO is ignored.

A valid received Frame is stored in the FIFO if channel ID, Frame ID, and cycle counter are not rejected by the configured rejection filter and rejection filter mask, and if there is no matching dedicated receive buffer.

25.6.8 Transmit Process

The transmit process is described in the following sections.

25.6.8.1 Static Segment

For the static segment, if there are several messages pending for transmission, the message with the Frame ID corresponding to the next sending slot is selected for transmission.

The Data Section of transmit buffers assigned to the static segment can be updated until the end of the preceding time slot. This means that a transfer from the Input Buffer has to be started by writing to the Input Buffer Command Request register latest at this time.

25.6.8.2 Dynamic Segment

In the dynamic segment, if several messages are pending, the message with the highest priority (lowest Frame ID) is selected next. Only Frame ID's which are higher than the largest static Frame ID are allowed for the dynamic segment.

In the dynamic segment different slot counter sequences are possible (concurrent sending of different Frame ID's on both channels). Therefore pending messages are selected according to their Frame ID and their channel configuration bit.

The Data Section of transmit buffers assigned to the dynamic segment can be updated until the end of the preceding slot. This means that a transfer from the Input Buffer has to be started by writing to the Input Buffer Command Request register latest at this time.

The start of latest transmit configured by SLT in the MHD Configuration Register 1 defines the maximum minislot value allowed before inhibiting new Frame transmission in the dynamic segment of the current cycle.

25.6.8.3 Transmit Buffers

A portion of the E-Ray Message Buffers can be configured as transmit buffers by programming bit CFG in the Header Section of the respective Message Buffer to 1. This can be done via the Write Header Section 1 register.

There exist the following possibilities to assign a transmit buffer to the Communication Controller channels:

- Static segment: channel A **or** channel B, channel A **and** channel B

FlexRay™ Protocol Controller (E-Ray)

- Dynamic segment: channel A **or** channel B

Message Buffer 0 is dedicated to hold the startup Frame, the SYNC Frame, or the designated single slot Frame as configured by TXST, TXSY, and TSM in the SUC Configuration Register 1. In this case it can be reconfigured in “DEFAULT_CONFIG” or “CONFIG” state only. This ensures that any node transmits at most one startup / SYNC Frame per communication cycle. Transmission of startup / SYNC Frames from other Message Buffers is not possible.

All other Message Buffers configured for transmission in static or dynamic segment are reconfigurable during runtime. Due to the organization of the Data Partition in the Message RAM (reference by data pointer), reconfiguration of the configured payload length and the data pointer in the Header Section of a Message Buffer may lead to erroneous configurations. If a Message Buffer is reconfigured during runtime it may happen that this Message Buffer is not send out in the respective communication cycle.

The Communication Controller does not have the capability to calculate the Header CRC. The Host is supposed to provide the Header CRCs for all transmit buffers. If Network Management is required the Host has to set the PPIT bit in the Header Section of the respective Message Buffer to 1 and write the Network Management information to the Data Section of the Message Buffer (see [Section 25.6.6](#)).

The payload length field configures the data payload length in 2-byte words. If the configured payload length of a static transmit buffer is shorter than the payload length configured for the static segment by SFDL in the Message Handler Configuration Register 1, the Communication Controller generates padding byte to ensure that Frames have proper physical length. The padding pattern is logical zero.

Each transmit buffer provides a transmission mode flag TXM that allows the Host to configure the transmission mode for the transmit buffer in the static segment. If this bit is set, the transmitter operates in the single-shot mode. If this bit is cleared, the transmitter operates in the continuous mode. In dynamic segment the transmitter always works in single-shot mode.

If a Message Buffer is configured in the continuous mode, the Communication Controller does not reset the transmission request flag TXR after successful transmission. In this case a Frame is sent out each time the Frame ID and cycle counter filter match. The TXR flag can be reset by the Host by writing the respective Message Buffer number to the Input Buffer Command Request register while bit **STXRH** in the Input Buffer Command Mask register is reset to 0.

If two or more transmit buffers are configured with the same Frame ID **and** cycle counter filter value, the transmit buffer with the lowest Message Buffer number will be transmitted in the respective slot.

25.6.8.4 Frame Transmission

To prepare a transmit buffer for transmission the following steps are required:

FlexRay™ Protocol Controller (E-Ray)

- Configure the Message Buffer as transmit buffer by writing bit CFG = 1 in the Write Header Section 1 register
- Write transmit message (Header and Data Section) to the Input Buffer.
- To transfer a transmit message from Input Buffer to the Message RAM proceed as described on **“Data Transfer from Input Buffer to Message RAM” on Page 25-224**.
- If configured in the Input Buffer Command Mask register the Transmission Request flag for the respective Message Buffer will be set as soon as the transfer has completed, and the Message Buffer is ready for transmission.
- Check whether the Message Buffer has been transmitted by checking the TXR bits (TXR = 0) in the Transmission Request 1,2 registers (single-shot mode only).

In single-shot mode the Communication Controller resets the TXR flag after transmission has been completed. Now the Host may update the transmit buffer with the next message. The Communication Controller does not transmit the message before the Host has indicated that the update is completed by setting the Transmission Request flag TXR again. The Host can check the actual state of the TXR flags of all Message Buffers by reading the Transmission Request registers. After successful transmission, if bit **WRHS1.MBI** in the Header Section of the respective Message Buffer is set, the transmit service request flag in the Status Service Request Register is set (TXI = 1). If enabled an service request is generated.

25.6.8.5 NULL Frame Transmission

If in static segment the Host does not set the transmission request flag before transmit time, and if there is no other transmit buffer with matching filter criteria (matching Frame ID and cycle counter filter), the Communication Controller transmits a NULL Frame with the NULL Frame indication bit reset to 0 and the payload data reset to zero.

In the following cases the Communication Controller transmits a NULL Frame with the NULL Frame indication bit reset to 0, and the rest of the Frame Header and the Frame length unchanged (payload data is reset to zero):

- All transmit buffers configured for the slot have cycle counter filters that do not match the current cycle
- There are matching Frame ID's and cycle counter filters, but none of these transmit buffers has the transmission request flag TXR set

NULL Frames are not transmitted in the dynamic segment.

25.6.9 Receive Process

The receive process is described in the following sections.

25.6.9.1 Frame Reception

To prepare or change a Message Buffer for reception the following steps are required:

- Configure the Message Buffer as receive buffer by writing bit CFG = 0 in the Write Header Section 1 register
- Configure the receive buffer by writing the configuration data (Header Section) to the Input Buffer
- Transfer the configuration from Input Buffer to the Message RAM by writing the number of the target Message Buffer to the Input Buffer Command Request register.

Once these steps are performed, the Message Buffer functions as an active receive buffer and participates in the internal acceptance filtering process, which takes place every time the Communication Controller receives a message. The first matching receive buffer is updated from the received message. If the Message Buffer holds an unprocessed Data Section (ND = 1) it is overwritten with the new message and the MLST bit in the respective Message Buffer Status register is set.

If the payload length of a received Frame PLC is longer than the value programmed by PLC in the Header Section of the respective Message Buffer, the data field stored in the Message Buffer is truncated to that length.

If no Frame, a NULL Frame, or a corrupted Frame is received in a slot, the Data Section of the Message Buffer configured for this slot is not updated. In this case only the flags in the Message Buffer Status register are updated to signal the cause of the problem. In addition the respective MBC flag in the Message Buffer Status Changed 1,2,3,4 registers is set.

When the Data Section of a receive buffer has been updated from a received Frame, the respective New Data **ND_n (n = 0-31)** to **ND_n (n = 96-127)** flag in the New Data **NDAT1** to **NDAT4** registers is set. When the Message Handler has updated the Message Buffer status, the respective MBC flag in the Message Buffer Status Changed 1,2,3,4 registers is set. If bit **RDHS1.MBI** in the Header Section of the respective Message Buffer is set, the receive service request flag in the Status Service Request Register is set (RXI = 1). If enabled an service request is generated.

To read a receive buffer from the Message RAM via the Output Buffer proceed as described on **“Data Transfer from Message RAM to Output Buffer” on Page 25-226**.

Note: The ND and MBC flags are automatically cleared by the Message Handler when the received message has been transferred to the Output Buffer.

25.6.9.2 NULL Frame reception

The Payload Segment of a received NULL Frame is **not** copied into the matching receive buffer. If a NULL Frame has been received, the Header Section of the matching Message Buffer is updated from the received NULL Frame. The NULL Frame indication bit in the Header Section 3 of the respective Message Buffer is reset (NFI = 0) and the respective MBC flag in the Message Buffer Status Changed 1,2,3,4 registers is set.

In case that bit ND and / or MBC were already set before this event because the Host did not read the last received message, bit MLST in the Message Buffer Status register of the respective Message Buffer is also set.

25.6.10 FIFO Function

A group of the Message Buffers can be configured as a cyclic First-In-First-Out (FIFO). The group of Message Buffers belonging to the FIFO is contiguous in the register map starting with the Message Buffer referenced by FFB and ending with the Message Buffer referenced by LCB in the Message RAM Configuration register. Up to 128 Message Buffers can be assigned to the FIFO.

25.6.10.1 Description

Every valid incoming message not matching with any dedicated receive buffer but passing the programmable FIFO filter is stored into the FIFO. In this case Frame ID, payload length, receive cycle count, and the status bits of the addressed FIFO Message Buffer are overwritten with Frame ID, payload length, receive cycle count, and the status from the received message and can be read by the Host for message identification. Bit RFNE in the Status Service Request Register shows that the FIFO is not empty, bit RFF in the Status Service Request Register is set when the last available Message Buffer belonging to the FIFO is written, bit RFO in the Error Service Request Register shows that a FIFO overrun has been detected. If enabled, service requests are generated.

There are two index registers associated with the FIFO. The PUT Index Register (PIDX) is an index to the next available location in the FIFO. When a new message has been received it is written into the Message Buffer addressed by the PIDX register. The PIDX register is then incremented and addresses the next available Message Buffer. If the PIDX register is incremented past the highest numbered Message Buffer of the FIFO, the PIDX register is loaded with the number of the first (lowest numbered) Message Buffer in the FIFO chain. The GET Index Register (GIDX) is used to address the next Message Buffer of the FIFO to be read. The GIDX register is incremented after transfer of the contents of a Message Buffer belonging to the FIFO to the Output Buffer. The PUT Index Register and the GET Index Register are not accessible by the Host.

The FIFO is completely filled when the PUT index (PIDX) reaches the value of the GET index (GIDX). When the next message is written to the FIFO before the oldest message has been read, both PUT index and GET index are incremented and the new message

FlexRay™ Protocol Controller (E-Ray)

overwrites the oldest message in the FIFO. This will set FIFO overrun flag RFO in the Error Service Request Register.

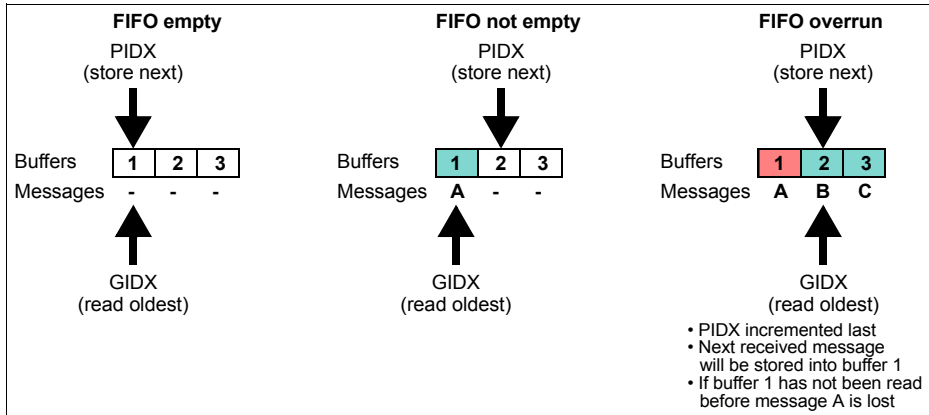


Figure 25-10 FIFO Status: Empty, Not Empty, Overrun

A FIFO non empty status is detected when the PUT index (PIDX) differs from the GET index (GIDX). In this case flag RFNE is set. This indicates that there is at least one received message in the FIFO. The FIFO empty, FIFO not empty, and the FIFO overrun states are explained in [Figure 25-10](#) for a three Message Buffer FIFO.

There is a programmable FIFO rejection filter for the FIFO. The FIFO Rejection Filter register (FRF) defines a filter pattern for messages to be rejected. The FIFO rejection filter consists of channel filter, Frame ID filter, and cycle counter filter. If bit RSS is set to 1 (default), all messages received in the static segment are rejected by the FIFO. If bit RNF is set to 1 (default), received NULL Frames are not stored in the FIFO.

The FIFO Rejection Filter Mask register (FRFM) specifies which bits of the Frame ID filter in the FIFO Rejection Filter register are marked “don’t care” for rejection filtering.

25.6.10.2 Configuration of the FIFO

For all Message Buffers belonging to the FIFO the data pointer to the first 32-bit word of the Data Section of the respective Message Buffer in the Message RAM has to be configured via the Write Header Section 3 register. All information required for acceptance filtering is taken from the FIFO rejection filter and the FIFO rejection filter mask and needs not be configured in the Header Sections of the Message Buffers belonging to the FIFO.

When programming the data pointers for the Message Buffers belonging to the FIFO, the payload length of all Message Buffers should be programmed to the same value.

FlexRay™ Protocol Controller (E-Ray)

*Note: It is recommended to program the MBI bits of the Message Buffers belonging to the FIFO to 0 via **WRHS1.MBI** to avoid generation of RX interrupts.*

*If the payload length of a received Frame is longer than the value programmed by **WRHS2.PLC** in the Header Section of the respective Message Buffer, the data field stored in a Message Buffer of the FIFO is truncated to that length.*

25.6.10.3 Access to the FIFO

To read from the FIFO the Host has to trigger a transfer from the Message RAM to the Output Buffer by writing the number of the first Message Buffer of the FIFO (referenced by FFB) to the Output Buffer Command Request register. The Message Handler then transfers the Message Buffer addressed by the GET Index Register (GIDX) to the Output Buffer. After this transfer the GET Index Register (GIDX) is incremented.

25.6.11 Message Handling

The Message Handler controls data transfers between the Input / Output Buffer and the Message RAM and between the Message RAM and the two Transient Buffer RAMs. All accesses to the internal RAM's are 32 bit accesses.

Access to the Message Buffers stored in the Message RAM is done under control of the Message Handler state machine. This avoids conflicts between accesses of the two protocol controllers and the Host to the Message RAM.

Frame IDs of Message Buffers assigned to the static segment have to be in the range from 1 to NSS as configured in the GTU Configuration Register 7. Frame IDs of Message Buffers assigned to the dynamic segment have to be in the range from NSS + 1 to 2047.

Received messages with no matching dedicated receive buffer (static or dynamic segment) are stored in the receive FIFO (if configured) if they pass the FIFO rejection filter.

25.6.11.1 Host access to Message RAM

The message transfer between Input Buffer and Message RAM as well as between Message RAM and Output Buffer is triggered by the Host by writing the number of the target / source Message Buffer to be accessed to the Input or Output Buffer Command Request register.

The Input / Output Buffer Command Mask registers can be used to write / read Header and Data Section of the selected Message Buffer separately. If bit **STXRS** in the Input Buffer Command Mask register is set (**STXRS** = 1), the transmission request flag TXR of the selected Message Buffer is automatically set after the Message Buffer has been updated.

FlexRay™ Protocol Controller (E-Ray)

If bit **STXRS** in the Input Buffer Command Mask register is reset (**STXRS** = 0), the transmission request flag TXR of the selected Message Buffer is reset. This can be used to stop transmission from Message Buffers operated in continuous mode.

Input Buffer (IBF) and the Output Buffer (OBF) are build up as a double buffer structure. One half of this double buffer structure is accessible by the Host (IBF Host / OBF Host), while the other half (IBF Shadow / OBF Shadow) is accessed by the Message Handler for data transfers between IBF / OBF and Message RAM.

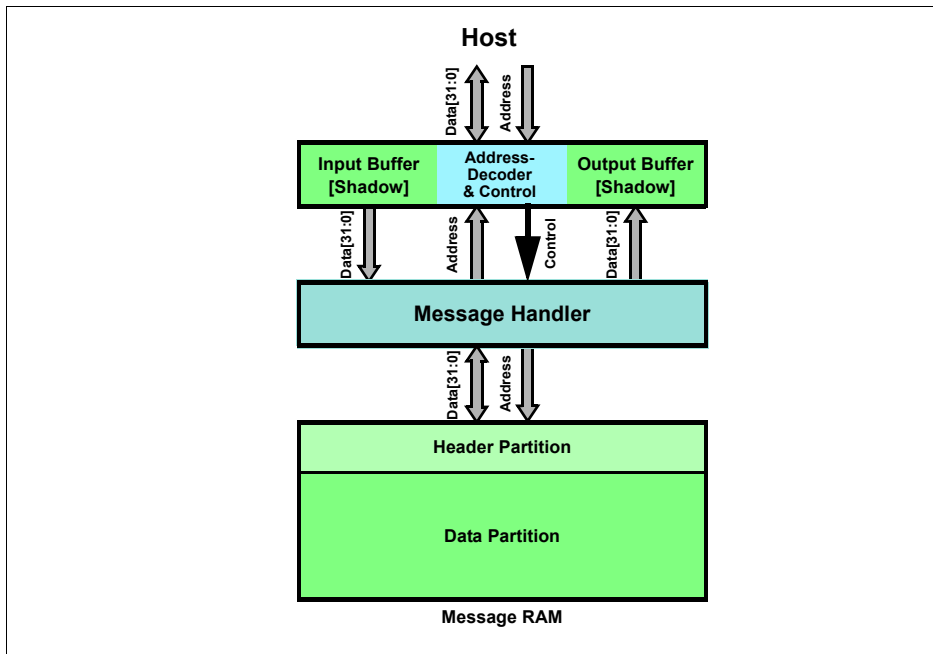


Figure 25-11 Host Access to Message RAM

Data Transfer from Input Buffer to Message RAM

To configure / update a Message Buffer in the Message RAM, the Host has to write the data to **WRDSnn** (**nn = 01-64**) and the Header to **WRHS1**, **WRHS2**, **WRHS3**. Two sets of **WRDSnn** (**nn = 01-64**) are available in parallel and selected by **CUST1.IBF1PAG** and **CUST1.IBF2PAG**. **CUST1.IBFS** shows which Input Buffer is currently used as Input Shadow Buffer and which as Input Host Buffer. **WRHS1**, **WRHS2**, and **WRHS3** does only exist once. The specific action is selected by configuring the Input Buffer Command Mask **IBCM**.

FlexRay™ Protocol Controller (E-Ray)

When the Host writes the number of the target Message Buffer in the Message RAM to IBRH in the Input Buffer Command Request register **IBCR**, IBF Host and IBF Shadow are swapped (see **Figure 25-12**).

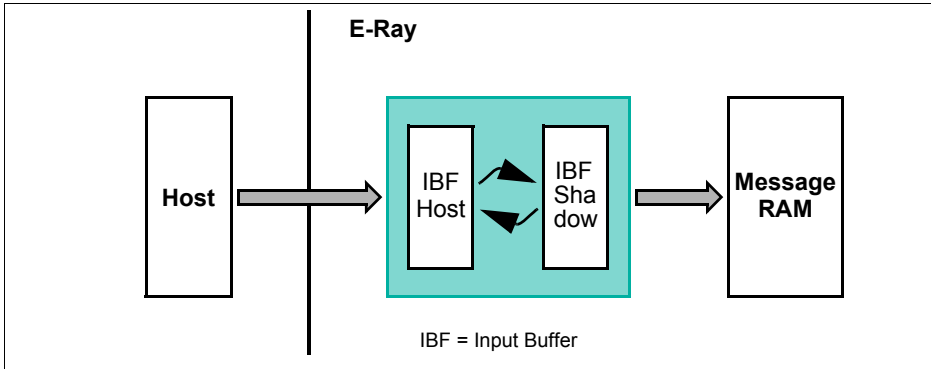


Figure 25-12 Double Buffer Structure Input Buffer

In addition the bits in the Input Buffer Command Mask and Input Buffer Command Request registers are also swapped to keep them attached to the respective IBF section (see **Figure 25-13**).

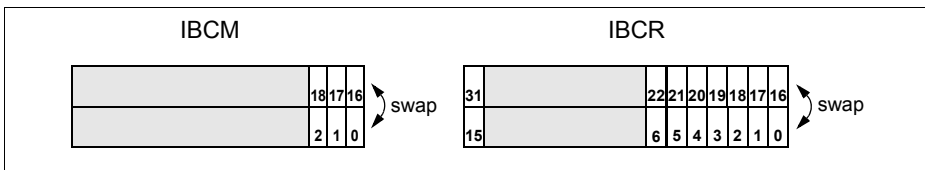


Figure 25-13 Swapping of IBCM and IBCR Bit

With this write operation the IBSYS bit in the Input Buffer Command Request register is set to 1. The Message Handler then starts to transfer the contents of IBF Shadow to the Message Buffer in the Message RAM selected by IBRS.

While the Message Handler transfers the data from IBF Shadow to the target Message Buffer in the Message RAM, the Host may write the next message to IBF Host. After the transfer between IBF Shadow and the Message RAM has completed, the IBSYS bit is set back to 0 and the next transfer to the Message RAM may be started by the Host by writing the respective target Message Buffer number to IBRH in the Input Buffer Command Request register.

If a write access to IBRH occurs while IBSYS is 1, IBSYH is set to 1. After completion of the ongoing data transfer from IBF Shadow to the Message RAM, IBF Host and IBF Shadow are swapped, IBSYH is reset to 0, IBSYS remains set to 1, and the next transfer

FlexRay™ Protocol Controller (E-Ray)

to the Message RAM is started. In addition the Message Buffer numbers stored under IBRH and IBRS and the Command Mask flags are also swapped.

Table 25-16 Assignment of Input Buffer Command Mask Bit

Pos.	Access	Bit	Function
18	rh	STXRS	Set Transmission Request Shadow
17	rh	LDSS	Load Data Section Shadow
16	rh	LHSS	Load Header Section Shadow
2	rw	STXRH	Set Transmission Request Host
1	rw	LDSH	Load Data Section Host
0	rw	LHSH	Load Header Section Host

Table 25-17 Assignment of Input Buffer Command Request Bit

Pos.	Access	Bit	Function
31	rh	IBSYS	IBF Busy Shadow , signals ongoing transfer from IBF Shadow to Message RAM
21–16	rh	IBRS	IBF Request Shadow , number of Message Buffer currently / last updated
15	rh	IBSYH	IBF Busy Host , transfer request pending for Message Buffer referenced by IBRH
5-0	rwh	IBRH	IBF Request Host , number of Message Buffer to be updated next

Data Transfer from Message RAM to Output Buffer

To read a Message Buffer from the Message RAM, the Host has to write to Command Request register **OBCR** to trigger the data transfer as configured in Output Buffer Command Mask **OBCM** register. After the transfer has completed, the Host can read the transferred data from **RDDSnn (nn = 01-64)**, **RDHS1**, **RDHS2**, **RDHS2**, and **MBS**.

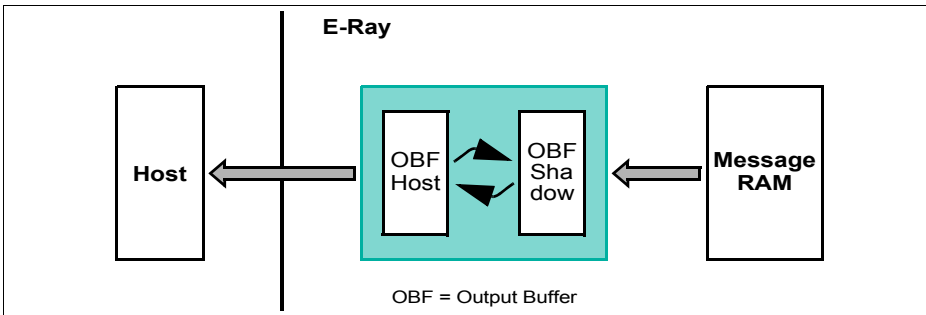


Figure 25-14 Double Buffer Structure Output Buffer

OBF Host and OBF Shadow as well as bits **OBCM.RHSS**, **OBCM.RDSS**, **OBCM.RHSH**, **OBCM.RDSH** and bits **OBCR.OBRS**, **OBCR.OBRH** are swapped under control of bits **OBCR.VIEW** and **OBCR.REQ**.

Writing bit **OBCR.REQ** to 1 copies bits **OBCM.RHSS**, **OBCM.RDSS** and bits **OBCR.OBRS** to an internal storage (see **Figure 25-15**).

After setting **OBCR.REQ** to 1, **OBCR.OBSYS** is set to 1, and the transfer of the Message Buffer selected by **OBCR.OBRS** from the Message RAM to OBF Shadow is started. After the transfer between the Message RAM and OBF Shadow has completed, the **OBCR.OBSYS** bit is set back to 0. Bits **OBCR.REQ** and **OBCR.VIEW** can only be set to 1 while **OBCR.OBSYS** is 0.

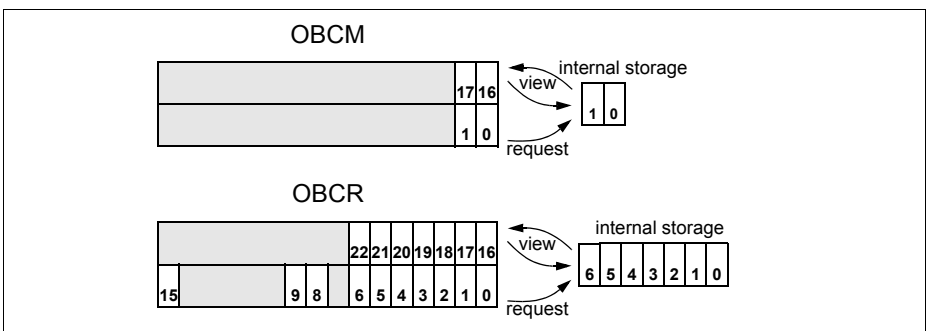


Figure 25-15 Swapping of OBCM and OBCR Bit

OBF Host and OBF Shadow are swapped by setting bit **OBCR.VIEW** to 1 while bit **OBCR.OBSYS** is 0 (see **Figure 25-14**).

In addition bits **OBCR.OBRH** and bits **OBCM.RHSH**, **OBCM.RDSH** are swapped with the registers internal storage thus assuring that the Message Buffer number stored in

FlexRay™ Protocol Controller (E-Ray)

OBCR.OBRH and the mask configuration stored in **OBCM.RHSH**, **OBCM.RDSH** matches the transferred data stored in OBF Host (see **Figure 25-15**).

Now the Host can read the transferred Message Buffer from OBF Host while the Message Handler may transfer the next message from the Message RAM to OBF Shadow.

Table 25-18 Assignment of Output Buffer Command Mask Bit

Pos.	Access	Bit	Function
17	rh	RDSH	Data Section available for Host access
16	rh	RHSH	Header Section available for Host access
1	rw	RDSS	Read Data Section Shadow
0	rw	RHSS	Read Header Section Shadow

Table 25-19 Assignment of Output Buffer Command Request Bit

Pos.	Access	Bit	Function
22–16	rh	OBRH	OBF Request Host , number of Message Buffer available for Host access
15	rh	OBSYS	OBF Busy Shadow , signals ongoing transfer from Message RAM to OBF Shadow
9	rw	REQ	Request Transfer from Message RAM to OBF Shadow
8	rwh	VIEW	View OBF Shadow, swap OBF Shadow, and OBF Host
6–0	rwh	OBRS	OBF Request Shadow , number of Message Buffer for next request

25.6.11.2 Data Transfers between IBF / OBF and Message RAM

This document uses the following terms and abbreviations:

Table 25-20 Terms and Abbreviations

Term	Meaning
MHD	Message Handler
IBF	Input Buffer 1 or 2 RAM
OBF	Output Buffer 1 or 2 RAM
MBF	Message Buffer RAM
TBF	Transient Buffer RAM Channel A (TBF1) or Channel B (TBF2)

FlexRay™ Protocol Controller (E-Ray)

Table 25-20 Terms and Abbreviations (cont'd)

IBF ⇒ MBF	Transfer from IBF to MBF
MBF ⇒ OBF	Transfer from MBF to OBF
MBF ⇒ TBF	Transfer from MBF to TBF
TBF ⇒ MBF	Transfer from TBF to MBF
SS	Slot Status
SS ⇒ MBF	Transfer SS to MBF

Message Handler functionality

The MHD controls the access to the MBF. It manages data-transfer between MBF and IBF, OBF, TBF1, TBF2. The data-path are shown in Figure 25-16.

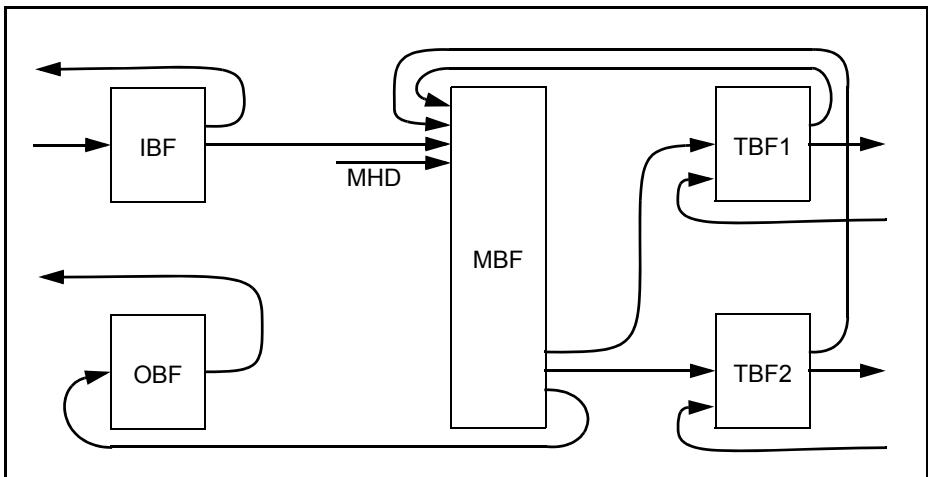


Figure 25-16 Interconnection of RAMs

Furthermore a search-algorithm allows to find the next valid message object in the MBF for transmission or reception.

Each transfer consists of a setup-time, four time steps to transfer the Header-section and a payload-length-dependent number of time steps to transfer the data-section. The internal data-busses have a width of 32 bits. Thereby it is possible to transfer two 2-byte words in one time step. If the payload consists of an odd number of 2-byte words the last time step of the data-section contains only 16 bit of valid data. If the Payload-Length (PL) is e.g. 7, the data-section consists of 4 time steps.

The maximum length for the data-section is 64 time steps, the minimum length is zero time steps.

FlexRay™ Protocol Controller (E-Ray)

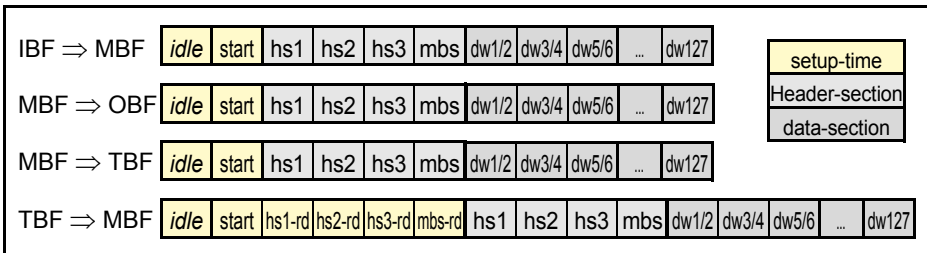


Figure 25-17 Different Possible Buffer Transfers

The update of the Slot-Status consists of a setup-time and one time-step to write the new Slot-Status.

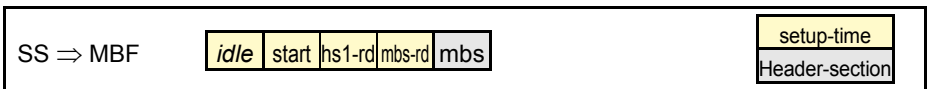


Figure 25-18 Update of Slot Status

The length of a time step depends on the number of concurrent tasks.

The following concurrent tasks are executed under control of the Message Handler:

- Data transfer between IBF or OBF and MBF
- Data transfer between TBF1 and MBF, search next TX / RX Message Buffer CHA
- Data transfer between TBF2 and MBF, search next TX / RX Message Buffer CHB

Thereby the time step length can vary between one and three f_{CLC_ERAY} periods.

Under certain conditions it is possible that a transfer is stopped or interrupted for a number of time steps until it is continued.

When a IB F \Rightarrow MBF is started short after a TBF \Rightarrow MBF or SS \Rightarrow MBF the transfer from IBF has to wait until the setup-time of the internal transfer has finished (see Figure 25-19)

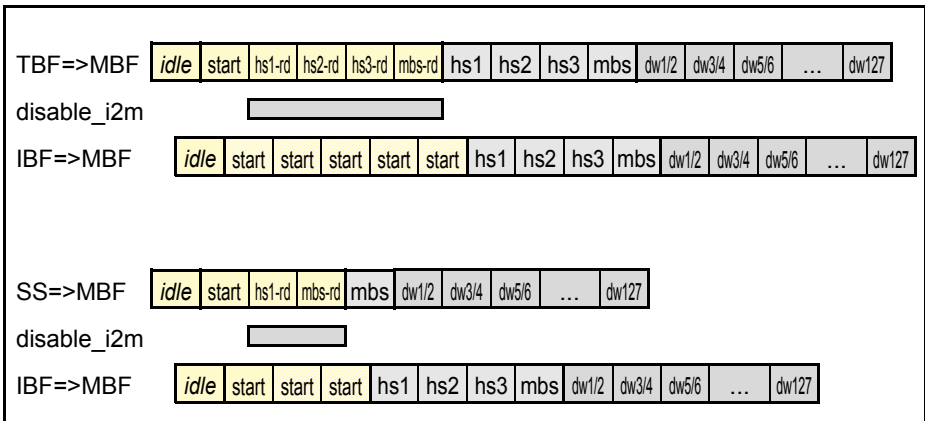


Figure 25-19 Delay start of IBF=>MBF

The internal signal “disable_i2m” is always active when the TBF ⇒ MBF is in state “hs1-rd”, “hs2-rd”, “hs3-rd” or “mbs-rd” and when the SS ⇒ MBF is in state “hs1-rd” or “mbs-rd”.

The IBF ⇒ MBF is hold in state “start” until the internal signal “disable_i2m” gets inactive.

These additional time-steps are independent of any address-counter-values. This means, the IBF ⇒ MBF has to wait even if it writes to another buffer than the internal transfer.

Multiple requests of transfers between IBF/OBF and Message RAM

The time required to transfer the contents of a Message Buffer between IBF / OBF and Message RAM depends on the number of 4-byte words to be transferred, the number of concurrent tasks to be managed by the Message Handler, and in special cases the type and address range of the internal transfer. The number of 4-byte words varies from 4 (Header Section only) to 68 (Header + maximum Data Section) plus a short setup time to start the first transfer, while the number of concurrent task varies from one to three. The 4 Header words have to be included in calculation even if only the Data Section is requested for transfer.

The following concurrent tasks are executed under control of the Message Handler:

- Data transfer between IBF or OBF and MBF
- Data transfer between TBF1 and MBF, search next TX / RX Message Buffer CHA
- Data transfer between TBF2 and MBF, search next TX / RX Message Buffer CHB

Transfers between IBF and MBF respectively MBF and OBF can only be handled one after another. In case that e.g. a IBF ⇒ MBF has been started shortly before a

FlexRay™ Protocol Controller (E-Ray)

MBF ⇒ OBF is requested, the MBF ⇒ OBF has to wait until the IBF ⇒ MBF has completed.

In case that e.g. a second IBF⇒MBF is requested, a MBF⇒OBF is requested and a IBF⇒MBF is ongoing, the MBF⇒OBF has to wait until the first IBF⇒IBF has completed. The second IBF=MBF has to wait until the MBF⇒OBF has completed (see figure 25-20) independent whether MBF⇒OBF or second IBF⇒MBF is requested first.

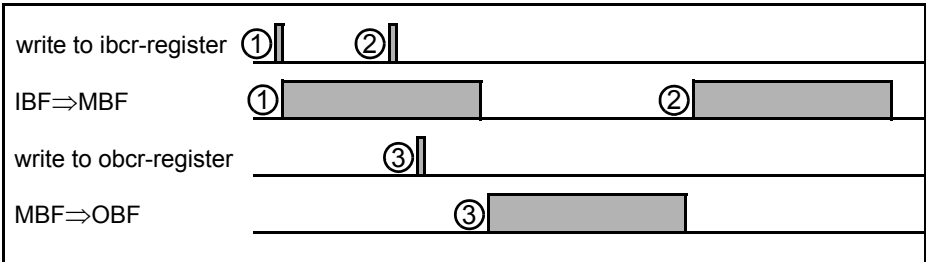


Figure 25-20 Multiple IBF/OBF Request

Worst case for single request

When a message with a large payload length is received the TBF⇒MBF is started at the begin of the next slot (n+1). If the next slot is a dynamic slot without transmission/reception (minislot), it may happen that the TBF⇒MBF has not finished until begin of the next but one slot (n+2). In this case the TBF⇒MBF will be service requested (break) to start a transmission in the next but one slot (MBF⇒TBF) and/or to update the slot status (SS⇒MBF) for the RX-buffer corresponding with next slot (n+1). After this interruption the TBF⇒MBF is continued.

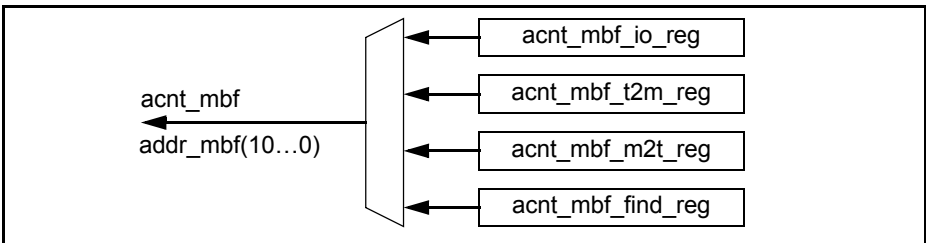
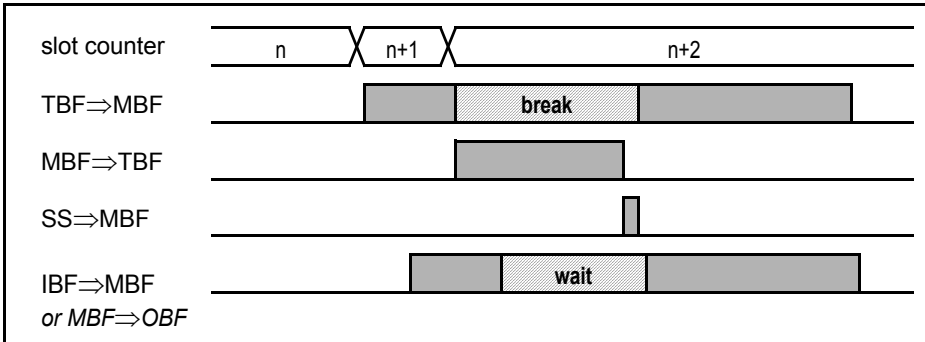


Figure 25-21 Address Counter Scheme of Message RAM (simplified)

For the transfers IBF⇒MBF / MBF⇒OBF, TBF⇒MBF and MBF⇒TBF separate address-counter are implemented (see Figure 25-21).


Figure 25-22 interruption of TBF=>MBF

If the address-counter for IBF=>MBF / MBF=>OBF (`acnt_mbf_io_reg`) reaches the address of the interrupted TBF=>MBF (`acnt_mbf_t2m_reg`) the IBF=>MBF / MBF=>OBF has to wait until the TBF=>MBF is continued (see Figure 25-22).

The relative time is measured in $f_{\text{CLC_ERAY}}$ cycles. Absolute time depends on the actual $f_{\text{CLC_ERAY}}$ cycle period.

$$\text{tbf_to_mbf_break time}_{\text{max}} = (\text{setup time} + \text{mbf_to_tbf time}_{\text{max}}) + (\text{setup time} + \text{ss_to_mbf})$$

$$\text{cycles}_{\text{req}} = (\text{number of concurrent tasks}) \times ((\text{setup time} + (\text{number of 4-byte words})_{\text{req}}) + \text{tbf_to_mbf_break time})$$

$$\text{setup time} = 2 \cdot f_{\text{CLC_ERAY}} \text{ cycles}$$

FlexRay™ Protocol Controller (E-Ray)

Worst case for one IBF⇒MBF or MBF⇒OBF:

$$\text{Max. break time: } t_{bf_to_mbf_break\ time}_{max} = (2+68) + (4+1) = 75$$

$$\text{Max. number of } f_{CLC_ERAY} \text{ cycles: } cycles_{req} = 3 \times (6 + 68 + 75) = 435$$

Worst case for multiple transfers

If a second IBF⇒MBF and a MBF⇒OBF (see Figure 25-20) is requested directly after the first IBF⇒MBF has started following worst case timing could appear:

$$cycles_{trans} = \begin{aligned} & \text{(remaining cycles of transfer running)} \\ & + \text{(cycles of second requested transfer)} \\ & + \text{(cycles of third requested transfer)} \end{aligned}$$

$$cycles_{trans} = cycles_{rem} + cycles_{req_2} + cycles_{req_3}$$

$$\text{Max. number of } f_{CLC_ERAY} \text{ cycles: } cycles_{trans} = 447 + 435 + 447 = 1329$$

25.6.11.3 Minimum f_{CLC_ERAY}

To calculate the minimum f_{CLC_ERAY} the worst case scenario has to be considered.

The worst case scenario depends on the following parameters

- maximum payload length
- minimum minislot length
- number of configured Message Buffers (excluding FIFO)
- used channels (single/dual channel)

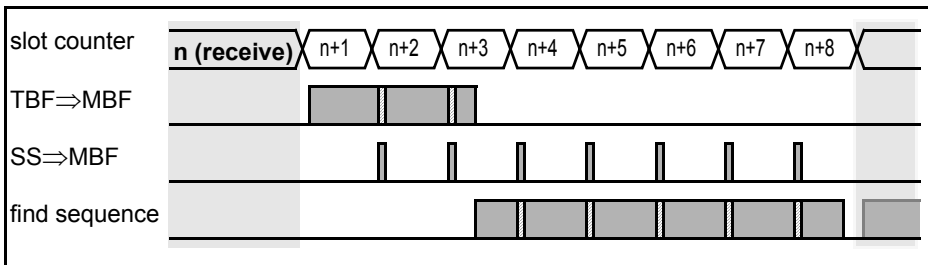


Figure 25-23 worst case scenario

FlexRay™ Protocol Controller (E-Ray)

Worst case scenario:

- reception of message with a maximum payload length in Slot n (n is 7,15,23,31,39,...)
- slot n+1 to n+7 are empty dynamic slots (minislot) and configured as receive buffer
- the find-sequence (usually started in slot 8,16,24,32,40,...) has to scan the maximum number of configured buffers
- the number of concurrent tasks has its maximum value of three

The find-sequence is executed each 8 Slots (slot 8,16,24,32,40,...). It has to be finished until the next find-sequence is requested.

The length of a TBF⇒MBF varies from 4 (Header Section only) to 68 (Header + maximum Data Section) time step plus a setup time of 6 time steps.

$$f_{\text{CLC_ERAY}} \text{ cycles}_{t2m} = \text{number of concurrent tasks} \times (\text{setup time}_{t2m} + (\text{number of 4-byte words})_{t2m})$$

A SS⇒MBF has a fixed length of 1 time steps plus a setup time of 4 time steps.

$$f_{\text{CLC_ERAY}} \text{ cycles}_{ss2m} = (\text{number of concurrent tasks}) \times 5$$

The find sequence has a maximum length of 128 (maximum number of buffers) time steps plus a setup time of 2 time steps.

$$f_{\text{CLC_ERAY}} \text{ cycles}_{\text{find}} = (\text{number of concurrent tasks}) \times (\text{setup time}_{\text{find}} + (\text{number of configured buffers}))$$

A minislot has a length of 2 to 63 Macrotick (gdMinislot). The minimum nominal Macrotick period (cdMinMTNom) is 1 μs. A sequence of 8 minislots has a length of

$$\text{time}_{8\text{minislots}} = 8 \times \text{gdMinislot} \times \text{cdMinMTNom}$$

FlexRay™ Protocol Controller (E-Ray)

The maximum period $T_{\text{CLC_ERAY}} = 1/f_{\text{CLC_ERAY}}$ can be calculated as followed:

$$\text{time}_{8\text{minislots}} \geq (f_{\text{CLC_ERAY}} \text{ period in } \mu\text{s}) \times (f_{\text{CLC_ERAY}} \text{ cycles}_{t2m}) + 7 \times (f_{\text{CLC_ERAY}} \text{ cycles}_{ss2m}) + (f_{\text{CLC_ERAY}} \text{ cycles}_{\text{find}})$$

$$f_{\text{CLC_ERAY}} \text{ period in ms} \leq \frac{\text{time}_{8\text{minislots}}}{(\text{cycles}_{t2m}) + 7 \times (\text{cycles}_{ss2m}) + (\text{cycles}_{\text{find}})}$$

$$\text{minimum time}_{8\text{minislots}} = 8 \times 2 \times 1 \mu\text{s} = 16 \mu\text{s}$$

$$\text{maximum } f_{\text{CLC_ERAY}} \text{ cycles}_{t2m} = 3 \times (6 + 68) = 222$$

$$\text{maximum } f_{\text{CLC_ERAY}} \text{ cycles}_{ss2m} = 3 * 5 = 15$$

$$\text{maximum } f_{\text{CLC_ERAY}} \text{ cycles}_{\text{find}} = 3 * (2 + 128) = 390$$

$$f_{\text{CLC_ERAY}} \text{ period in ms} \leq \frac{16\mu\text{s}}{222 + 7 \times 15 + 390} = 22.315\dots\text{ns}$$

The minimum $f_{\text{CLC_ERAY}}$ frequency for this worst case scenario is 44.8125 MHz.

A too low $f_{\text{CLC_ERAY}}$ frequency can cause a malfunction of the E-Ray.

The E-Ray can detect several malfunctions and reports this by setting the corresponding flag in the Message Handler Constraints Flags (**MHDF**) register.

Minimum $f_{\text{CLC_ERAY}}$ for various maximum payload length

Table 25-21 summarizes the minimum required $f_{\text{CLC_ERAY}}$ frequency for various maximum payload length assuming:

- a minimum minislot length of 2μs.
- a maximum of 128 configured Message Buffers.
- dual channels in use.

Table 25-21 Minimum $f_{\text{CLC_ERAY}}$ for different maximum payload length

Maximum payload length of 32 bit words	4	8	16	32	64
minimum $f_{\text{CLC_ERAY}}$	32,82 MHz	33,57 MHz	35,07 MHz	38,07 MHz	44,1 MHz

Minimum $f_{\text{CLC_ERAY}}$ for various minimum minislot length

Table 25-22 summarizes the minimum required $f_{\text{CLC_ERAY}}$ frequency for various minimum minislot length assuming:

FlexRay™ Protocol Controller (E-Ray)

- a maximum payload length of 254 bytes / 64 four-byte-words.
- a maximum 128 configured Message Buffers.
- dual channels in use.

Table 25-22 Minimum $f_{\text{CLC_ERAY}}$ for different minimum minislot length

gdMinislot at dMinMTNom = 1 μs	2 μs	3 μs	4 μs	7 μs	8 μs
minimum $f_{\text{CLC_ERAY}}$	44,82 MHz	29,88 MHz	22,412 MHz	12,8 MHz	9,96 MHz

Minimum $f_{\text{CLC_ERAY}}$ for various amount of configured Message Buffers

Table 25-23 summarizes the minimum required $f_{\text{CLC_ERAY}}$ frequency for various amount of configured Message Buffers assuming:

- a maximum payload length of 254 bytes / 64 four-byte-words.
- a minimum minislot length of 2 μs.
- dual channels in use.

Table 25-23 Minimum $f_{\text{CLC_ERAY}}$ for different amount of configured Message Buffers

Configured maximum amount of Message Buffers	128	64	32
minimum $f_{\text{CLC_ERAY}}$	44,82 MHz	32,82 MHz	26,82 MHz

Minimum $f_{\text{CLC_ERAY}}$ for a typical configuration

The minimum required $f_{\text{CLC_ERAY}}$ frequency for various assuming the following typical E-Ray configuration:

- a maximum payload length of 32 bytes / 8 four-byte-words.
- a minimum minislot length of 7 μs.
- a maximum 128 configured Message Buffers.
- dual channels in use

The minimum $f_{\text{CLC_ERAY}}$ frequency for this typical example would be 10 MHz.

FlexRay™ Protocol Controller (E-Ray)**25.6.11.4 FlexRay™ Protocol Controller access to Message RAM**

The two Transient Buffer RAMs (TBF 1, TBF 2) are used to buffer the data for transfer between the two FlexRay™ Protocol Controllers and the Message RAM.

Each Transient Buffer RAM is build up as a double buffer, able to store two complete FlexRay™ messages. There is always one buffer assigned to the corresponding Protocol Controller while the other one is accessible by the Message Handler.

If e.g. the Message Handler writes the next message to be send to Transient Buffer Tx, the FlexRay™ Channel Protocol Controller can access Transient Buffer Rx to store the message it is actually receiving. During transmission of the message stored in Transient Buffer Tx, the Message Handler transfers the last received message stored in Transient Buffer Rx to the Message RAM (if it passes acceptance filtering) and updates the respective Message Buffer.

Data transfers between the Transient Buffer RAMs and the shift registers of the FlexRay™ Channel Protocol Controllers are done in words of 32 bit. This enables the use of a 32 bit shift register independent of the length of the FlexRay™ messages.

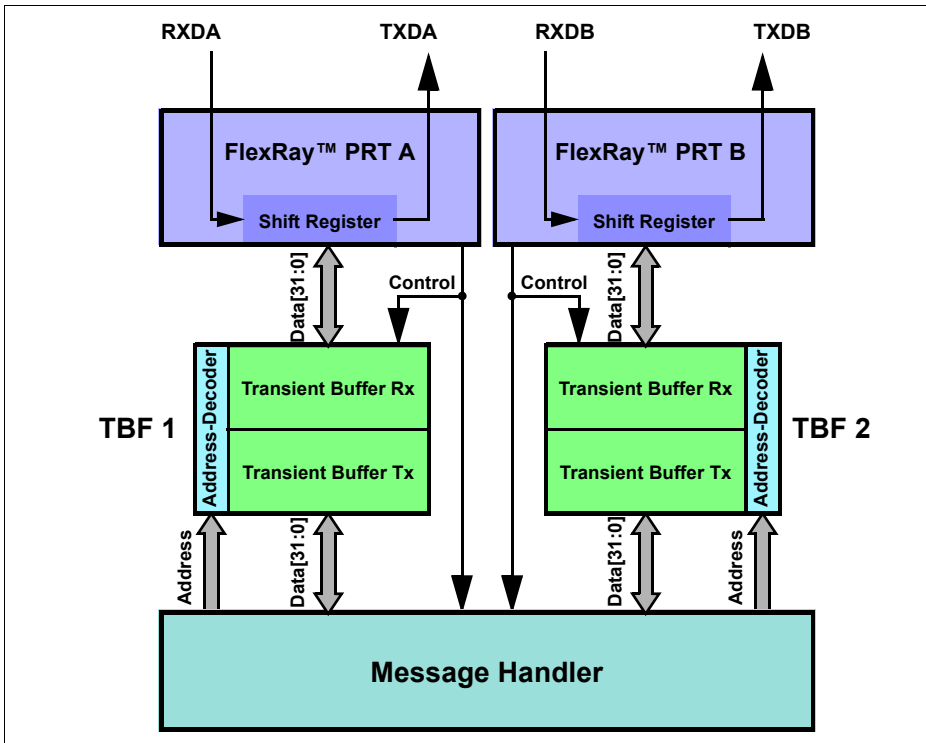


Figure 25-24 Access to Transient Buffer RAMs

25.6.12 Message RAM

To avoid conflicts between Host access to the Message RAM and FlexRay™ message reception / transmission, the Host cannot directly access the Message Buffers in the Message RAM. These accesses are handled via the Input and Output Buffers. The Message RAM is able to store up to 128 Message Buffers depending on the configured payload length.

The Message RAM is organized 2048 x 32. To achieve the required flexibility with respect to different numbers of data byte per FlexRay™ Frame (0 to 254), the Message RAM has a structure as shown in [Figure 25-25](#).

The Data Partition is allowed to start at Message RAM word number: $(MRC.LCB + 1) \cdot 4$

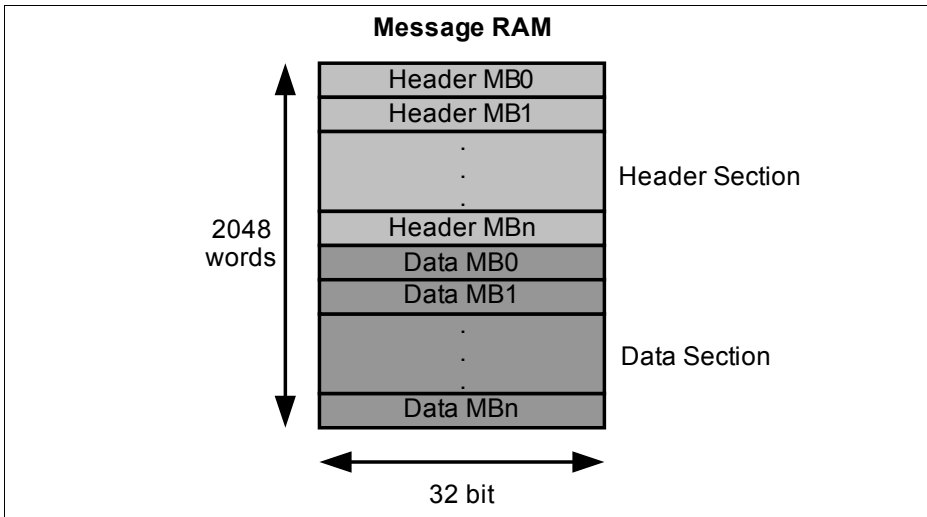


Figure 25-25 Structure of Message RAM

Header Partition

Stores Header Segments of FlexRay™ Frames:

- Supports a maximum of 128 Message Buffers
- Each Message Buffer has a Header of four 32 bit words
- Header 3 of each Message Buffer holds the 11 bit pointer to the respective Data Section in the Data Partition

Data Partition

Flexible storage of Data Sections with different length. Some maximum values are:

- 30 Message Buffers with 254 byte Data Section each
- Or 56 Message Buffers with 128 byte Data Section each
- Or 128 Message Buffers with 48 byte Data Section each

Restriction: Header Partition + Data Partition may not occupy more than 2048 32-bit words.

25.6.12.1 Header Partition

The Header of each Message Buffer occupies four 32-bit words in the Header Partition of the Message RAM. The Header of Message Buffer 0 starts with the first word in the Message RAM.

For transmit buffers the Header CRC has to be calculated by the Host.

Payload Length Received **PLR**, Receive Cycle Count **RCC**, Received on Channel Indication **RCI**, Startup Frame Indication bit **SFI**, Sync bit **SYN**, NULL Frame Indication bit **NFI**, Payload Preamble Indication bit **PPI**, and Reserved bit **RES** are only updated from received valid Frames (including valid NULL Frames).

Header word 4 of each configured Message Buffer holds the respective Message Buffer Status **MBS** information.

Table 25-24 Header Section of a Message Buffer in the Message RAM

Bit Word	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
1			M B I	T X I	N F I	C F E	C H G	C H B	C A	Cycle Code										Frame ID																	
2	Payload Length Received								Payload Length Configured								Tx Buffer: Header CRC Configured Rx Buffer: Header CRC Received																				
3			R E S	P E S	N F I	S Y N	S F N	R F I		Receive Cycle Count										Data Pointer																	
4			R E S	P E S	N F I	S Y N	S F N	R F I		Cycle Count Status										T F B	F T Y	M E S	E L S	T S C	T C C	S V V	S C V	C V V	S E E	C E E	S E E	S E E	V E E	V E E	F E E	F E E	V E E

- Frame Configuration
- Filter Configuration
- Message Buffer Control
- Message RAM Configuration
- Updated from received Frame
- Message Buffer Status
- unused

Header 1 (word 0)

Write access via **WRHS1**, read access via **RDHS1**:

- Frame ID: Slot counter filtering configuration
- Cycle Code: Cycle counter filtering configuration
- CHA, CHB: Channel filtering configuration
- CFG: Message Buffer configuration: receive / transmit
- PPIT: Payload Preamble Indicator Transmit
- XMI: Transmit mode configuration: single-shot / continuous
- MBI: Message Buffer receive / transmit service request enable

Header 2 (word 1)

Write access via **WRHS2**, read access via **RDHS2**:

- Header CRC
 - Transmit Buffer: Configured by the Host (calculated from Frame Header Segment)
 - Receive Buffer: Updated from received Frame
- Payload Length Configured
 - Length of Data Section (2-byte words) as configured by the Host
- Payload Length Received
 - Length of Payload Segment (2-byte words) stored from received Frame

Header 3

Write access via **WRHS3**, read access via **RDHS3**:

- Data Pointer
 - Pointer to the beginning of the corresponding Data Section in the Data Partition

Read access via **RDHS3**, valid for receive buffers only, updated from received Frames:

- Receive Cycle Count: Cycle count from received Frame
- RCI: Received on Channel Indicator
- SFI: Startup Frame Indicator
- SYN: SYNC Frame Indicator
- NFI: NULL Frame Indicator
- PPI: Payload Preamble Indicator
- RES: Reserved bit

Message Buffer Status MBS (word 3)

Read access via MBS, updated by the Communication Controller at the end of the configured slot.

- VFRA: Valid Frame Received on channel A
- VFRB: Valid Frame Received on channel B
- SEOA: Syntax Error Observed on channel A
- SEOB: Syntax Error Observed on channel B
- CEOA: Content Error Observed on channel A
- CEOB: Content Error Observed on channel B
- SVOA: Slot boundary Violation Observed on channel A
- SVOB: Slot boundary Violation Observed on channel B
- TCIA: Transmission Conflict Indication channel A
- TCIB: Transmission Conflict Indication channel B
- ESA: Empty Slot Channel A
- ESB: Empty Slot Channel B
- MLST: Message LoST
- FTA: Frame Transmitted on Channel A
- FTA: Frame Transmitted on Channel B
- Cycle Count Status: Actual cycle count when status was updated
- RCIS: Received on CHannel Indicator Status
- SFIS: Startup Frame Indicator Status
- SYNS: SYNC Frame Indicator Status
- NFIS: NULL Frame Indicator Status
- PPIS: Payload Preamble Indicator Status
- RESS: Reserved Bit Status

25.6.12.2 Data Partition

The Data Partition of the Message RAM stores the Data Sections of the Message Buffers configured for reception / transmission as defined in the Header Partition. The number of data bytes for each Message Buffer can vary from 0 to 254. To optimize the data transfer between the shift registers of the two FlexRay™ Protocol Controllers and the Message RAM as well as between the Host interface and the Message RAM, the physical width of the Message RAM is set to 4 bytes.

The Data Partition starts after the last word of the Header Partition. When configuring the Message Buffers in the Message RAM the programmer has to assure that the data pointers point to addresses within the Data Partition. [Table 25-25](#) below shows an example how the Data Sections of the configured Message Buffers can be stored in the Data Partition of the Message RAM.

The beginning and the end of a Message Buffer's Data Section is determined by the data pointer and the payload length configured in the Message Buffer's Header Section,

FlexRay™ Protocol Controller (E-Ray)

respectively. This enables a flexible usage of the available RAM space for storage of Message Buffers with different data length.

If the size of the Data Section is an odd number of 2-byte words, the remaining 16 bits in the last 32-bit word are unused (see [Table 25-25](#) below)

Table 25-25 Example for Structure of the Data Section in the Message RAM

Bit Word	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
...	unused								unused								unused								unused							
...	unused								unused								unused								unused							
...	MB1 Data3								MB1 Data2								MB1 Data1								MB1 Data0							
...							
...							
...	MB1 Data(n)								MB1 Data(n-1)								MB1 Data(n-2)								MB1 Data(n-3)							
...							
...							
...							
...	MB1 Data3								MB1 Data2								MB1 Data1								MB1 Data0							
...							
...	MB1 Data(k)0								MB1 Data(k-1)0								MB1 Data(k-2)0								MB1 Data(k-3)0							
2046	MB80 Data3								MB80 Data2								MB80 Data2								MB80 Data0							
2047	unused								unused								MB80 Data5								MB80 Data4							

25.6.12.3 ECC Check

There is an ECC checking mechanism implemented in the E-Ray module to assure the integrity of the data stored in the seven RAM blocks of the module. The RAM blocks have an ECC generator / checker attached as shown in [Figure 25-26](#). When data is written to a RAM block, the local ECC generator generates the ECC data. The ECC data is stored together with the respective data word. The ECC data is checked each time a data word is read from any of the RAM blocks.

If an ECC error is detected, the respective error flag is set. The ECC error flags [MHDS.EIBF](#), [MHDS.EOBF](#), [MHDS.EMR](#), [MHDS.ETBF1](#), [MHDS.ETBF2](#), and the faulty Message Buffer indicators [MHDS.FMBD](#), [MHDS.MFMB](#), [MHDS.FMB](#) are located in the Message Handler Status register. These error flags control the error interrupt flag [EIR.EERR](#).

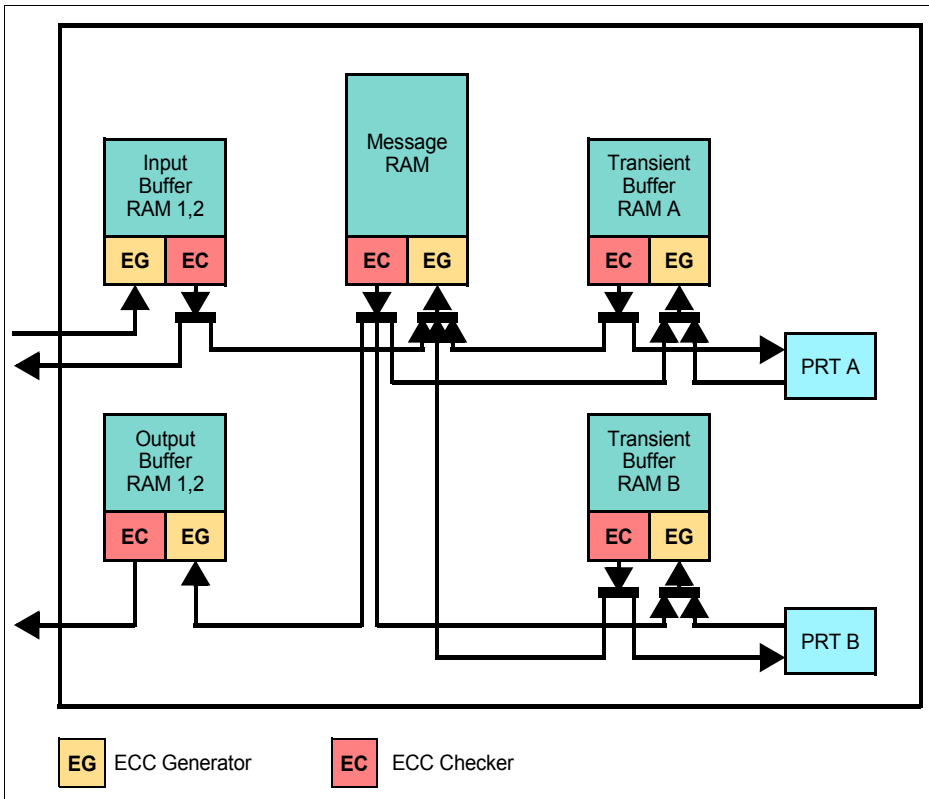


Figure 25-26 ECC Generation and Check

When an ECC error has been detected the following actions will be performed:

In all cases

- The respective ECC error flag in the Message Handler Status **MHDS** register is set
- The ECC error flag **EIR.EERR** in the Error Service Request Register is set, and if enabled, a module service request to the Host will be generated.

Additionally in specific cases

1. ECC error in data transfer from Input Buffer RAM 1,2 ⇒ Message RAM (Transfer of Header and Data Section)
 - a) **MHDS.EIBF** bit is set
 - b) **MHDS.FMBD** bit is set to indicate that **MHDS.FMB** has been updated
 - c) **MHDS.FMB** indicates the number of the faulty Message Buffer
 - d) Transmit buffer: Transmission request for the respective Message Buffer is not set
2. ECC error in data transfer from Input Buffer RAM 1,2 ⇒ Message RAM (Transfer of Data Section only)
 - a) **MHDS.EMR** bit is set
 - b) **MHDS.FMBD** bit is set to indicate that **MHDS.FMB** points to a faulty Message Buffer
 - c) **MHDS.FMB** indicates the number of the faulty Message Buffer
 - d) The Data Section of the respective Message Buffer is not updated
 - e) Transmit buffer: Transmission request for the respective Message Buffer is not set
3. ECC error during host reading Input Buffer RAM
 - a) • **MHDS.EIBF** bit is set
4. ECC error during scan of Header Sections in Message RAM
 - a) **MHDS.EMR** bit is set
 - b) **MHDS.FMBD** bit is set to indicate that **MHDS.FMB** points to a faulty Message Buffer
 - c) **MHDS.FMB** indicates the number of the faulty Message Buffer
 - d) Ignore Message Buffer (Message Buffer is skipped)
5. ECC error during data transfer from Message RAM ⇒ Transient Buffer RAM A, B
 - a) **MHDS.EMR** bit is set
 - b) **MHDS.FMBD** bit is set to indicate that **MHDS.FMB** points to a faulty Message Buffer
 - c) **MHDS.FMB** indicates the number of the faulty Message Buffer
 - d) Frame not transmitted, Frames already in transmission are invalidated by setting the Frame CRC to zero
6. ECC error during data transfer from Transient Buffer RAM A, B ⇒ Protocol Controller 1, 2
 - a) **MHDS.ETBF1**, **MHDS.ETBF2** bit is set
7. ECC error in data transfer from Transient Buffer RAM A, B ⇒ Message RAM (ECC error when reading Header Section of respective Message Buffer from Message RAM)
 - a) **MHDS.EMR** bit is set
 - b) **MHDS.FMBD** bit is set to indicate that **MHDS.FMB** points to a faulty Message Buffer
 - c) **MHDS.FMB** indicates the number of the faulty Message Buffer
 - d) The Data Section of the respective Message Buffer is not updated

FlexRay™ Protocol Controller (E-Ray)

8. ECC error in data transfer from Transient Buffer RAM A, B ⇒ Message RAM (ECC error when reading Transient Buffer RAM A, B)
 - a) **MHDS.ETBF1, MHDS.ETBF2** bit is set
 - b) **MHDS.FMBD** bit is set to indicate that **MHDS.FMB** points to a faulty Message Buffer
 - c) **MHDS.FMB** indicates the number of the faulty Message Buffer
9. ECC error during data transfer from Message RAM ⇒ Output Buffer RAM
 - a) **MHDS.EMR** bit is set
 - b) **MHDS.FMBD** bit is set to indicate that **MHDS.FMB** points to a faulty Message Buffer
 - c) **MHDS.FMB** indicates the number of the faulty Message Buffer
10. ECC error during Host reading Output Buffer RAM
 - a) • **MHDS.EOBF** bit is set
11. ECC error during data read of Transient Buffer RAM A, B

If an ECC error occurs while the Message Handler reads a Frame with Network Management information (PPI = 1) from the Transient Buffer RAM A, B the corresponding Network Management vector registers NMV1 to NMV3 are not updated from that Frame.

25.6.13 Host Handling of Errors

An ECC error caused by transient bit flips can be fixed by:

25.6.13.1 Self-Healing

ECC errors located in

- Input Buffer RAM 1,2
- Output Buffer RAM 1,2
- Data Section of Message RAM
- Transient Buffer RAM A
- Transient Buffer RAM B

are overwritten with the next write access to the disturbed bit(s) caused by Host access or by FlexRay communication.

25.6.13.2 CLEAR_RAMs Command

When called in DEFAULT_CONFIG or CONFIG state POC command CLEAR_RAMs initializes all module-internal RAMs to zero.

25.6.13.3 Temporary Unlocking of Header Section

An ECC error in the header section of a locked message buffer can be fixed by a transfer from the Input Buffer to the locked buffer Header Section. For this transfer, the write-

FlexRay™ Protocol Controller (E-Ray)

access to the IBCR (specifying the message buffer number) must be immediately preceded by the unlock sequence normally used to leave CONFIG state (see **“Lock Register (LCK)” on Page 25-30**).

For that single transfer the respective message buffer header is unlocked, regardless whether it belongs to the FIFO or whether its locking is controlled by MRC.SEC[1:0], and will be updated with new data.

25.7 Module Service Request

In general, service requests provide a close link to the protocol timing as they are triggered almost immediately when an error or status change is detected by the controller, a Frame is received or transmitted, a configured timer service request is activated, or a stop watch event occurred. This enables the Host to react very quickly on specific error conditions, status changes, or timer events. On the other hand too many service requests can cause the Host to miss deadlines required for the application. Therefore the Communication Controller supports disable / enable controls for each individual service request source separately.

An service request may be triggered when

- An error was detected
- A status flag is set
- A timer reaches a preconfigured value
- A message transfer from Input Buffer to Message RAM or from Message RAM to Output Buffer has completed
- A stop watch event occurred

Tracking status and generating service requests when a status change or an error occurs are two independent tasks. Regardless of whether an service request is enabled or not, the corresponding status is tracked and indicated by the Communication Controller. The Host has access to the actual status and error information by reading the Error Service Request Register **EIR** and the Status Service Request **SIR** Register.

Table 25-26 Module Service Request Flags and Service Request Line Enable

Register	Bit	Function
SIR	WST	Wakeup Status
	CAS	Collision Avoidance Symbol
	CYCS	Cycle Start Service Request
	TXI	Transmit Service Request
	RXI	Receive Service Request
	RFNE	Receive FIFO not Empty
	RFF	Receive FIFO Full
	NMVC	Network Management Vector Changed
	TI0	Timer Service Request 0
	TI1	Timer Service Request 1
	TIBC	Transfer Input Buffer Completed
	TOBC	Transfer Output Buffer Completed
	SWE	Stop Watch Event
	SUCS	Startup Completed Successfully
	MBSI	Message Buffer Status Interrupt
	SDS	Start of Dynamic Segment
	WUPA	Wakeup Pattern Channel A
	MTSA	MTS Received on Channel A
	WUPB	Wakeup Pattern Channel B
MTSB	MTS Received on Channel B	
ILE	EINT0	Enable Service Request Line 0
	EINT1	Enable Service Request Line 1
EIR	PEMC	Protocol Error Mode Changed
	CNA	Command Not Valid
	SFBM	SYNC Frames Below Minimum
	SFO	SYNC Frame Overflow
	CCF	Clock Correction Failure
	CCL	CHI Command Locked
	EERR	ECC Error
	RFO	Receive FIFO Overrun

FlexRay™ Protocol Controller (E-Ray)
Table 25-26 Module Service Request Flags and Service Request Line Enable

Register	Bit	Function
EIR	EFA	Empty FIFO Access
	IIBA	Illegal Input Buffer Access
	IOBA	Illegal Output Buffer Access
	MHF	Message Handler Constraints Flag
	EDA	Error Detected on Channel A
	LTV A	Latest Transmit Violation Channel A
	TABA	Transmission Across Boundary Channel A
	EDB	Error Detected on Channel B
	LTV B	Latest Transmit Violation Channel B
	TABB	Transmission Across Boundary Channel B

The interrupt lines to the Host TINT0SR and TINT1SR are controlled by the enabled interrupts. In addition each of the two interrupt lines can be enabled / disabled separately by programming bit ILE.EINT0/INT0SRC.SRE and ILE.EINT1/INT1SRC.SRE.

The interrupt lines to the Host NDAT0SR and NDAT1SR are controlled by the enabled new data interrupts (**NDIC1** to **NDIC4**). In addition each of the two interrupt lines can be enabled / disabled separately by programming bit NDAT0SRC.SRE and NDAT1SRC.SRE.

The interrupt lines to the Host MBSC0SR and MBSC1SR are controlled by the enabled new data interrupts (**MSIC1** to **MSIC4**). In addition each of the two interrupt lines can be enabled / disabled separately by programming bit MBSC0SRC.SRE and MBSC1SRC.SRE.

The two timer service requests generated by service request timer 0 and 1 are available on pins TINT0SR and TINT1SR. They can be configured via the Timer 0 and Timer 1 Configuration register. In addition each of the two interrupt lines can be enabled / disabled separately by programming bit TINT0SRC.SRE and TINT1SRC.SRE.

A stop watch event may be triggered via input pin STPWn.

The status of the data transfer between IBF / OBF and the Message RAM is signalled on signals IBUSY and OBUSY. When a transfer has completed bit **SIR.TIBC** or **SIR.TOBC** is set.

25.8 Restrictions

The following restrictions have to be considered when programming the E-Ray IP-module. A violation of these restrictions may lead to an erroneous behavior of the E-Ray IP-module.

25.8.1 Message Buffers with the same Frame ID

If two or more Message Buffers are configured with the same Frame ID, and if they have a matching cycle counter filter value for the same slot, then the Message Buffer with the lowest Message Buffer number is used.

Sharing of a static time slot via cycle counter filtering between different nodes of a FlexRay™ network is **not** allowed.

25.8.2 Data Transfers between IBF / OBF and Message RAM

The time required to transfer the contents of a Message Buffer between IBF / OBF and Message RAM depends on the setup time to start the first transfer, the number of 4-byte words to be transferred, and the number of concurrent tasks to be managed by the Message Handler. The number of 4-byte words varies from 4 (Header Section only) to 68 (Header + maximum Data Section) while the number of concurrent task varies from one to three.

The following concurrent tasks are executed under control of the Message Handler:

- Data transfer between IBF or OBF and Message RAM
- Data transfer between TBF1 and Message RAM, search next TX / RX Message Buffer CHA
- Data transfer between TBF2 and Message RAM, search next TX / RX Message Buffer CHB

Transfers between IBF and Message RAM respectively Message RAM and OBF can only be handled one after another. In case that e.g. a transfer between IBF and Message RAM has been started shortly before a transfer between Message RAM and OBF is requested, the OBF transfer has to wait until the IBF transfer has completed.

The relative time is measured in f_{CLC_ERAY} cycles. Absolute time depends on the actual f_{CLC_ERAY} cycle period.

$cyclestrans = (\text{remaining cycles of transfer running}) + (\text{cycles of requested transfer})$

$cyclesrem = cyclesrem + cyclesreq$

$cyclesrem = (\text{number of concurrent tasks}) * (\text{setup time} + (\text{number of 4-byte words})rem)$

$cyclesreq = (\text{number of concurrent tasks}) * (\text{setup time} + (\text{number of 4-byte words})req)$

$setup\ time = 2 \cdot f_{CLC_ERAY}\ \text{cycles}$

Under worst case conditions a transfer is requested directly after the previous transfer started:

FlexRay™ Protocol Controller (E-Ray)

Max. number of f_{CLC_ERAY} cycles: $cyclestrans = (3 * (2 + 68)) + (3 * (2 + 68)) = 420$
 Worst case timing: $timetrans(40MHz) = 420 * 25ns = 10.5 ms$

25.9 E-Ray Module Implementation

This section describes the E-Ray interfaces as implemented in TC26x B-Step with the clock control, port and DMA connections, interrupt control, and address decoding.

Figure 25-27 shows a detailed view of the E-Ray interface.

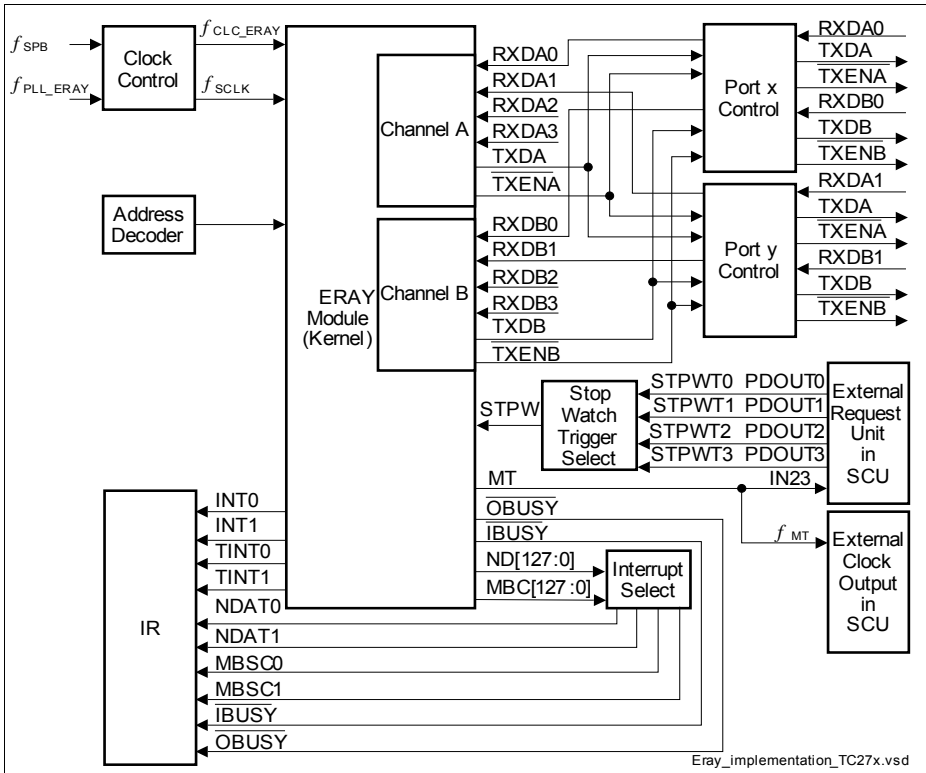


Figure 25-27 Detailed Block Diagram of the E-Ray Interface

25.9.1 Interconnections of the E-Ray Module

The E-Ray module has 2 FlexRay™ communication channels, channel A and channel B. Each channel provides a set of signals to drive a bus driver. The E-Ray module requires two different clocks, a sampling clock of the FlexRay™ bus f_{SCLK} . f_{SCLK} has to be 8 times the baud rate of the FlexRay™ communication. A second clock f_{CLC_ERAY} is

FlexRay™ Protocol Controller (E-Ray)

used for the main protocol controller state machine and the customer interface logic. To enable deactivation of the E-Ray Module, f_{CLC_ERAY} and f_{SCLK} may be disabled (clock gated) by the **CLC.DISR** Enable E-Ray (Clock Gating) bit. The following items are described in this section:

- E-Ray module (kernel) external registers
- Port control and connections
 - I/O port line assignment
 - I/O function selection
 - Pad driver characteristics selection
- On-chip connections
 - SCU Connections
 - DMA connections
- Module clock generation
- Interrupt registers
- E-Ray address map

25.9.2 Port Control and Connections

This section describes the I/O connections of the E-Ray module.

25.9.2.1 Input/Output Function Selection

Table 25-27 shows how bits and bit fields must be programmed for the required I/O functionality of the E-Ray I/O lines. This table also shows the values of the peripheral input select registers.

Table 25-27 E-Ray I/O Control Selection and Setup

Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
FlexRay™ Channel A			
RXDA0/ P14.8	ERAY_CUST1.RISA = 00 _B	P14_IOC8.PC8 = 0XXXX _B	In
RXDA1/ P11.9	ERAY_CUST1.RISA = 01 _B	P11_IOC8.PC9 = 0XXXX _B	In
RXDA2/ P02.1	ERAY_CUST1.RISA = 10 _B	P02_IOC0.PC1 = 0XXXX _B	In
RXDA3/ P14.1	ERAY_CUST1.RISA = 11 _B	P14_IOC0.PC1 = 0XXXX _B	In
TXDA/ P02.0	not applicable	P02_IOC0.PC0 = 1X110 _B	Out

FlexRay™ Protocol Controller (E-Ray)
Table 25-27 E-Ray I/O Control Selection and Setup (cont'd)

Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
TXDA/ P11.3	not applicable	P11_IOC0.PC3 = 1X100 _B	Out
TXDA/ P14.10	not applicable	P14_IOC8.PC10 = 1X110 _B	Out
TXDA/ P14.0	not applicable	P14_IOC0.PC0 = 1X011 _B	Out
TXENA/ P02.4	not applicable	P2_IOC4.PC4 = 1X110 _B	Out
TXENA/ P11.6	not applicable	P11_IOC4.PC6 = 1X100 _B	Out
TXENA/ P14.9	not applicable	P14_IOC8.PC9 = 1X110 _B	Out
FlexRay™ Channel B			
RXDB0/ P14.7	ERAY_CUST1.RISB = 00 _B	P14_IOC4.PC7 = 0XXXX _B	In
RXDB1/ P11.10	ERAY_CUST1.RISB = 01 _B	P11_IOC8.PC10 = 0XXXX _B	In
RXDB2/ P02.3	ERAY_CUST1.RISB = 10 _B	P02_IOC0.PC3 = 0XXXX _B	In
RXDB3/ P14.1	ERAY_CUST1.RISB = 11 _B	P14_IOC0.PC1 = 0XXXX _B	In
TXDB/ P02.2	not applicable	P02_IOC0.PC2 = 1X110 _B	Out
TXDB/ P14.0	not applicable	P14_IOC0.PC0 = 1X100 _B	Out
TXDB/ P14.5	not applicable	P14_IOC4.PC5 = 1X110 _B	Out
TXDB/ P11.12	not applicable	P11_IOC12.PC12 = 1X100 _B	Out
TXENB/ P02.5	not applicable	P2_IOC4.PC5 = 1X110 _B	Out
TXENB/ P14.6	not applicable	P14_IOC4.PC6 = 1X110 _B	Out

FlexRay™ Protocol Controller (E-Ray)
Table 25-27 E-Ray I/O Control Selection and Setup (cont'd)

Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
$\overline{\text{TXENB}}/\text{P14.9}$	not applicable	P14_IOCR8.PC9 = 1X101 _B	Out
$\overline{\text{TXENB}}/\text{P11.11}$	not applicable	P11_IOCR8.PC11 = 1X110 _B	Out
$\overline{\text{TXENB}}/\text{P11.6}$	not applicable	P11_IOCR4.PC6 = 1X010 _B	Out

25.9.3 On-Chip Connections

This section describes all on-chip interconnections of the E-Ray modules except the connections to I/O ports (see).

25.9.3.1 E-Ray Connections with IR

The E-Ray module of the TC26x B-Step has several on-chip interconnections to the IR. [Table 25-28](#) shows these interconnections. These enable the IR to handle different service request of E-Ray module via the DMA or Interrupt Service Routine.

Table 25-28 Request Assignment for IR

Line #	ERAY Output Signal	IR Request Input Line
00	INT0	SRC_ERAYINT0
01	INT1	SRC_ERAYINT1
02	TINT0	SRC_ERAYTINT0
03	TINT1	SRC_ERAYTINT1
04	NDAT0	SRC_ERAYNDAT0
05	NDAT1	SRC_ERAYNDAT1
06	MBSC0	SRC_ERAYMBSC0
07	MBSC1	SRC_ERAYMBSC1
08	OBUSY	SRC_ERAYOBUSY
09	IBUSY	SRC_ERAYIBUSY

25.9.3.2 E-Ray Connections with SMU

The E-Ray module of the TC26x B-Step provides to the SMU the following alarms (ALMx): SRAM ECC single bit correction, SRAM ECC uncorrectable error, SRAM address error, SRAM buffer address error..

25.9.3.3 E-Ray Connections with the External Request Unit of SCU

The E-Ray module of the TC26x B-Step has several on-chip interconnections to the External Request Unit (ERU) in the SCU to externally trigger stop watch events and to provide a global time e.g. to the on chip timers. [Table 25-29](#) and [Table 25-30](#) show these interconnections.

Table 25-29 External Stop Watch Request Assignment

ERAY Input Signal	ERU Request Output Line	Selected by
STPWT0	ERU_PDOUT0	CUST1.STPWTS = 00 _B
STPWT1	ERU_PDOUT1	CUST1.STPWTS = 01 _B
STPWT2	ERU_PDOUT2	CUST1.STPWTS = 10 _B
STPWT3	ERU_PDOUT3	CUST1.STPWTS = 11 _B

Table 25-30 Global Macrotick Connection to ERU

ERAY Output Signal	ERU Request Input Line	Selected by
MT	ERU_IN23	ERU_EICR1.EXIS0 = 11 _B

25.9.3.4 E-Ray Connections to GTM

The E-Ray module of the TC26x B-Step has several on-chip interconnections to the Generic Timer Module (GTM). [Table 25-31](#) show these interconnections.

Table 25-31 Global Macrotick Connection to GTM

ERAY Output Signal	TIM Input Line
MT	TIM0_7
MT	TIM1_7
MT	TIM2_7
MT	TIM3_7

25.9.3.5 E-Ray Connections with the External Clock Output of SCU

The E-Ray module of the TC26x B-Step has one on-chip interconnections to the External Clock Output Unit in the SCU to distribute externally as also internally the Macro Tick as time base for distributed system control. [Table 25-32](#) shows this interconnection.

Table 25-32 Global Macrotick Connection to External Clock Output

ERAY Output Signal	External Clock Output	Selected by
MT0	f_{MT0}	SCU_EXTCON.SEL0 = 1111 _B

25.9.4 OCDS Trigger Bus (OTGB) Interface

The E-Ray module has two 16 bit and one 32 bit Trigger Sets ([Table 25-33](#)) which are selected with the **OTSS** register.

Table 25-33 E-Ray Trigger Sets

Trigger Set	Details
TS16_SEP Service Requests, Errors and POC State	Table 25-35
TS16_MC Macrotick Counter	Table 25-36
TS32_SCSC State, Cycle and Slot Counter	Table 25-37

Table 25-34 shows all possible Trigger Set mapping options. If OTGB0 and/or OTGB1 is not used for E-Ray, Trigger Sets of other sources can be added in the OTGM module.

Table 25-34 Trigger Set Mapping Options

Width	OTGB0	OTGB1	OTGB2
32 Bit	TS16_SEP/MC		
		TS16_SEP/MC	
	TS16_SEP/MC	TS16_SEP/MC	
64 Bit			TS32_SCSC
	TS16_SEP/MC		TS32_SCSC
		TS16_SEP/MC	TS32_SCSC
	TS16_SEP/MC	TS16_SEP/MC	TS32_SCSC

Table 25-35 TS16_SEP Service Requests, Errors and POC State

Bits	Description
0	Interrupt 0 Service Request (INT0SRC)
1	Interrupt 1 Service Request (INT1SRC)
2	Timer Interrupt 0 Service Request (TINT0SRC)
3	Timer Interrupt 1 Service Request (TINT1SRC)
4	New Data 0 Service Request (NDAT0SRC)
5	New Data 1 Service Request (NDAT1SRC)
6	Message Buffer Status Changed 0 Service Request (MBSC0SRC)
7	Message Buffer Status Changed 1 Service Request (MBSC1SRC)

FlexRay™ Protocol Controller (E-Ray)
Table 25-35 TS16_SEP Service Requests, Errors and POC State (cont'd)

Bits	Description
8	Output Buffer Busy Service Request (OBUSYSRC)
9	Input Buffer Busy Service Request (IBUSYSRC)
10	Reserved
11	Error on Channel A (EIR.EDA)
12	Error on Channel B (EIR.EDB)
[15:13]	POC State bits[2:0] (CCSV.POCS)

Table 25-36 TS16_MC Macrotick Counter

Bits	Description
[13:0]	Macrotick Value (MTCCV.MTV)
[15:14]	Reserved

Table 25-37 TS32_SCSC State, Cycle and Slot Counter

Bits	Description
[10:0]	Slot Counter Channel A (SCV.SCCA)
[22:12]	Slot Counter Channel B (SCV.SCCB)
[29:24]	Cycle Counter Value (MTCCV.CCV)
30	Transfer Input Buffer Completed (SIR.TIBC)
31	Transfer Output Buffer Completed (SIR.TOBC)
11,23	Reserved

TS32_SCSC becomes valid (posedge on otgb2_valid_o) on a change of any Slot Counter or Transfer Buffer state. This covers also the Cycle Counter which will always change with a slower rate.

25.9.5 OTGB E-Ray Registers

25.9.5.1 OCDS Trigger Bus (OTGB)

The OTGB control register is cleared by Debug Reset. Write access is 32 bit wide only and requires Supervisor Mode.

OTSS

OCDS Trigger Set Select (0870_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															OTGB2
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						OTGB1		0						OTGB0	
r						rw		r						rw	

Field	Bits	Type	Description
OTGB0	[1:0]	rw	Trigger Set for OTGB0 0 _D No Trigger Set selected 1 _D Trigger Set TS16_SEP (Table 25-35) 2 _D Trigger Set TS16_MC (Table 25-36) 3 _D reserved
OTGB1	[9:8]	rw	Trigger Set for OTGB1 0 _D No Trigger Set selected 1 _D Trigger Set TS16_SEP (Table 25-35) 2 _D Trigger Set TS16_MC (Table 25-36) 3 _D reserved
OTGB2	16	rw	Trigger Set for OTGB2 0 _D No Trigger Set selected 1 _D Trigger Set TS32_SCSC (Table 25-37)
0	[7:2], [15:10], [31:17]	r	Reserved Read as 0; must be written with 0.

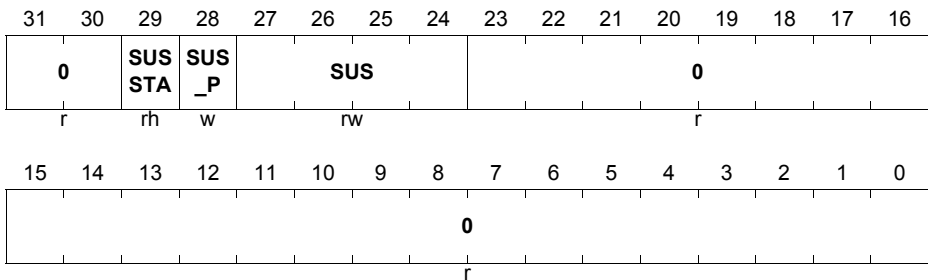
FlexRay™ Protocol Controller (E-Ray)

Field	Bits	Type	Description
RMC	[10:8]	rw	<p>Clock Divider in Run Mode</p> <p>000_B No clock signal f_{CLC_ERAY} generated (default after reset)</p> <p>001_B Clock $f_{CLC_ERAY} = f_{SPB}$ selected</p> <p>010_B Clock $f_{CLC_ERAY} = f_{SPB} / 2$ selected</p> <p>011_B Clock $f_{CLC_ERAY} = f_{SPB} / 3$ selected</p> <p>100_B Clock $f_{CLC_ERAY} = f_{SPB} / 4$ selected</p> <p>101_B Clock $f_{CLC_ERAY} = f_{SPB} / 5$ selected</p> <p>110_B Clock $f_{CLC_ERAY} = f_{SPB} / 6$ selected</p> <p>111_B Clock $f_{CLC_ERAY} = f_{SPB} / 7$ selected</p> <p><i>Note: This bit field is not affected by an application reset.</i></p> <p><i>Note: This bit field only controls the kernel clock f_{CLC_ERAY} and not the sampling clock f_{SCLK}.</i></p>
0	[31:16], [15:11], [7:4], 2	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Note: The number of module clock cycles (wait states) which are required by the kernel to execute a read or write access depends on the selected CLC clock frequency, which is selected via bit field RMC in the CLC register. Therefore, increasing CLC.RMC may result in a longer FPI Bus read cycle access time for kernel registers and can also slow down the write throughput to the kernel registers.

FlexRay™ Protocol Controller (E-Ray)
OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset. The OCS register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS register value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32 bit wide only and requires Supervisor Mode.

OCS
OCDS Control and Status
(08E8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clocks f_{CLC_ERAY} and the sampling clock f_{SCLK} are switched off immediately. No read or write access to any registers. 2 _H Soft suspend. This bit forces the module into freeze state. others , reserved
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:0], [31:30]	r	Reserved Read as 0; must be written with 0.

FlexRay™ Protocol Controller (E-Ray)
Access Enable Register (ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ... , EN31 -> TAG ID 011111_B.

ACCEN0
Access Enable Register 0
(08FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Field	Bits	Type	Description
ENn (n = 0-31)	n	r/w	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

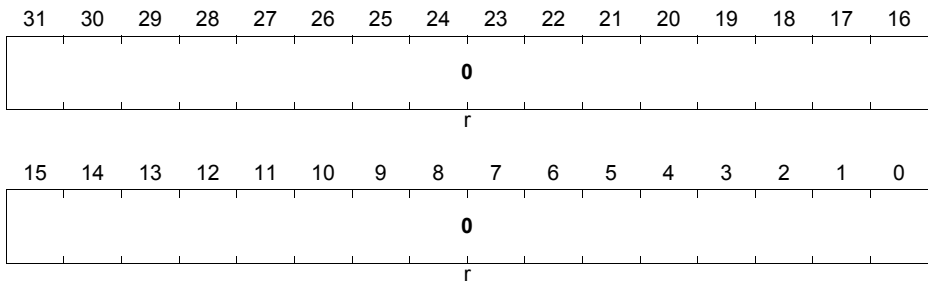
Access Enable Register (ACCEN1)

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

FlexRay™ Protocol Controller (E-Ray)

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 10000_B, EN1 -> TAG ID 100001_B, ... , EN31 -> TAG ID 111111_B.

ACCEN1
Access Enable Register 1
(08F8_H)
Reset Value: 0000 0000_H


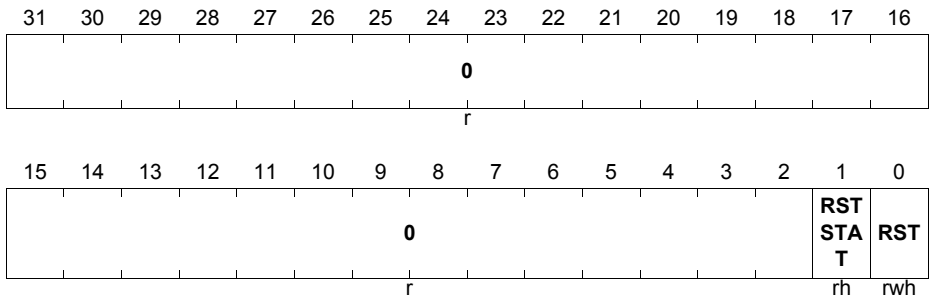
Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLE.CLR register bit.

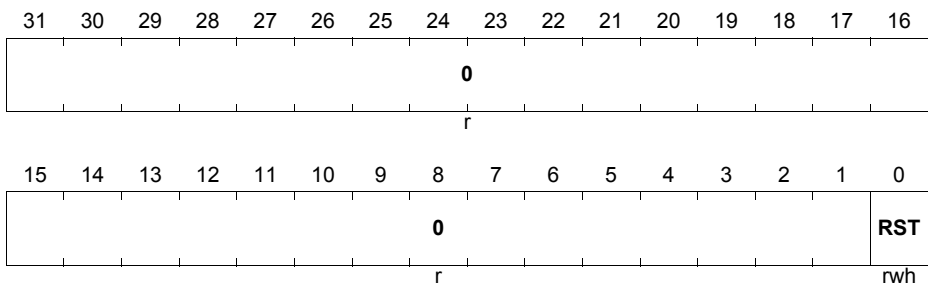
Note: During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

FlexRay™ Protocol Controller (E-Ray)
KRST0
Kernel Reset Register 0
(08F4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
RSTSTAT	1	rw	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits. 0 _B No kernel reset was executed 1 _B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)
Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers (KRSTx1.RST and KRSTx0.RST) related to the module kernel that should be reset (kernel 0 or kernel 1). The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

KRST1
Kernel Reset Register 1
(08F0_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

FlexRay™ Protocol Controller (E-Ray)

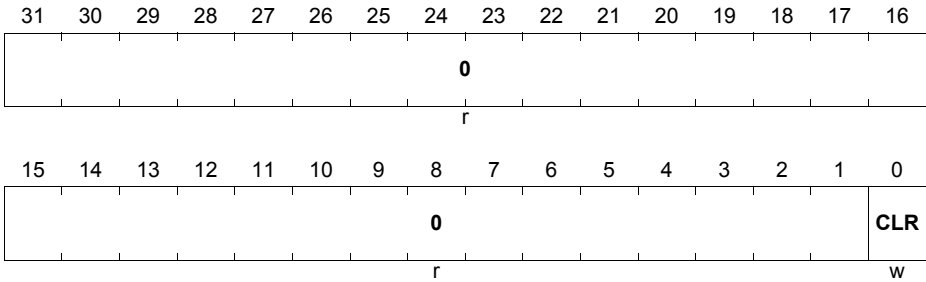
Kernel Reset Status Clear Register (KRSTCLR)

The Kernel Reset Register Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

KRSTCLR

Kernel Reset Status Clear Register (08EC_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

25.9.7 Interrupt Registers

Two different type of Interrupt Registers are described within this chapter.

The Interrupt Control register enable the selection of the Service Request used to signal an event. The Interrupt Control registers **NDIC1** to **NDIC4** select the service request node used for New Data Events. The Interrupt Control registers **MSIC1** to **MSIC4** select the service request node used for Message Buffer Status Changed Events.

The Interrupt Service Request Control Registers control the eight service request nodes.

FlexRay™ Protocol Controller (E-Ray)
New Data Interrupt Control 1 (NDIC1)

This New Data Interrupt Control register controls the interrupt that becomes active (NDAT0SRC or NDAT1SRC) on a ND flag turning active of all configured Message Buffers 0 to Message Buffers 31.

NDIC1
New Data Interrupt Control 1
(03A8_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NDIP 31	NDIP 30	NDIP 29	NDIP 28	NDIP 27	NDIP 26	NDIP 25	NDIP 24	NDIP 23	NDIP 22	NDIP 21	NDIP 20	NDIP 19	NDIP 18	NDIP 17	NDIP 16
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDIP 15	NDIP 14	NDIP 13	NDIP 12	NDIP 11	NDIP 10	NDIP 9	NDIP 8	NDIP 7	NDIP 6	NDIP 5	NDIP 4	NDIP 3	NDIP 2	NDIP 1	NDIP 0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
NDIPn (n = 0-31)	n	rW	New Data Interrupt Pointer n (n = 0-31) NDIPn determines the interrupt (NDAT0SRC or NDAT1SRC) of the service request output that becomes active on a New Data Flag becoming active. 0 _B NDAT0SRC selected for New Data Service Request 1 _B NDAT1SRC selected for New Data Service Request

FlexRay™ Protocol Controller (E-Ray)
New Data Interrupt Control 2 (NDIC2)

This New Data Interrupt Control register controls the interrupt that becomes active (NDAT0SRC or NDAT1SRC) on a ND flag turning active of all configured Message Buffers 32 to Message Buffers 63.

NDIC2
New Data Interrupt Control 2
(03AC_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NDIP 63	NDIP 62	NDIP 61	NDIP 60	NDIP 59	NDIP 58	NDIP 57	NDIP 56	NDIP 55	NDIP 54	NDIP 53	NDIP 52	NDIP 51	NDIP 50	NDIP 49	NDIP 48
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDIP 47	NDIP 46	NDIP 45	NDIP 44	NDIP 43	NDIP 42	NDIP 41	NDIP 40	NDIP 39	NDIP 38	NDIP 37	NDIP 36	NDIP 35	NDIP 34	NDIP 33	NDIP 32
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
NDIPn (n = 32-63)	n - 32	rW	New Data Interrupt Pointer n (n = 32-63) NDIPn determines the interrupt (NDAT0SRC or NDAT1SRC) of the service request output that becomes active on a New Data Flag becoming active. 0 _B NDAT0SRC selected for New Data Service Request 1 _B NDAT1SRC selected for New Data Service Request

FlexRay™ Protocol Controller (E-Ray)
New Data Interrupt Control 3 (NDIC3)

This New Data Interrupt Control register controls the interrupt that becomes active (NDAT0SRC or NDAT1SRC) on a ND flag turning active of all configured Message Buffers 64 to Message Buffers 95.

NDIC3
New Data Interrupt Control 3
(03B0_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NDIP 95	NDIP 94	NDIP 93	NDIP 92	NDIP 91	NDIP 90	NDIP 89	NDIP 88	NDIP 87	NDIP 86	NDIP 85	NDIP 84	NDIP 83	NDIP 82	NDIP 81	NDIP 80
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDIP 79	NDIP 78	NDIP 77	NDIP 76	NDIP 75	NDIP 74	NDIP 73	NDIP 72	NDIP 71	NDIP 70	NDIP 69	NDIP 68	NDIP 67	NDIP 66	NDIP 65	NDIP 64
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
NDIPn (n = 64-95)	n - 64	rW	New Data Interrupt Pointer n (n = 64-95) NDIPn determines the interrupt (NDAT0SRC or NDAT1SRC) of the service request output that becomes active on a New Data Flag becoming active. 0 _B NDAT0SRC selected for New Data Service Request 1 _B NDAT1SRC selected for New Data Service Request

FlexRay™ Protocol Controller (E-Ray)
New Data Interrupt Control 4 (NDIC4)

This New Data Interrupt Control register controls the interrupt that becomes active (NDAT0SRC or NDAT1SRC) on a ND flag turning active of all configured Message Buffers 96 to Message Buffers 127.

NDIC4

New Data Interrupt Control 4 (03B4_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NDIP 127	NDIP 126	NDIP 125	NDIP 124	NDIP 123	NDIP 122	NDIP 121	NDIP 120	NDIP 119	NDIP 118	NDIP 117	NDIP 116	NDIP 115	NDIP 114	NDIP 113	NDIP 112
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDIP 111	NDIP 110	NDIP 109	NDIP 108	NDIP 107	NDIP 106	NDIP 105	NDIP 104	NDIP 103	NDIP 102	NDIP 101	NDIP 100	NDIP 99	NDIP 98	NDIP 97	NDIP 96
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
NDIPn (n = 96-127)	n - 96	rw	New Data Interrupt Pointer n (n = 96-127) NDIPn determines the interrupt (NDAT0SRC or NDAT1SRC) of the service request output that becomes active on a New Data Flag becoming active. 0 _B NDAT0SRC selected for New Data Service Request 1 _B NDAT1SRC selected for New Data Service Request

FlexRay™ Protocol Controller (E-Ray)

Message Buffer Status Changed Interrupt Control 1 (MSIC1)

This Message Buffer Status Change Interrupt Control register controls the interrupt that becomes active (MBSC0SRC or MBSC1SRC) on a MBC flag of all configured Message Buffer 0 to Message Buffer 31 turning active.

MSIC1

Message Buffer Status Changed Interrupt Control 1

(03B8_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIP 31	MSIP 30	MSIP 29	MSIP 28	MSIP 27	MSIP 26	MSIP 25	MSIP 24	MSIP 23	MSIP 22	MSIP 21	MSIP 20	MSIP 19	MSIP 18	MSIP 17	MSIP 16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIP 15	MSIP 14	MSIP 13	MSIP 12	MSIP 11	MSIP 10	MSIP 9	MSIP 8	MSIP 7	MSIP 6	MSIP 5	MSIP 4	MSIP 3	MSIP 2	MSIP 1	MSIP 0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MSIPn (n = 0-31)	n	rw	<p>Message Buffer Status Changed Interrupt Pointer n (n = 0-31)</p> <p>MSIPn determines the interrupt (MBSC0SRC or MBSC1SRC) of the service request output that becomes active on a Message Buffer Status Changed Flag becoming active.</p> <p>0_B MBSC0SRC selected for Message Buffer Status Changed Service Request</p> <p>1_B MBSC1SRC selected for Message Buffer Status Changed Service Request</p>

FlexRay™ Protocol Controller (E-Ray)

Message Buffer Status Changed Interrupt Control 2 (MSIC2)

This Message Buffer Status Change Interrupt Control register controls the interrupt that becomes active (MBSC0SRC or MBSC1SRC) on a MBC flag of all configured Message Buffer 32 to Message Buffer 63 turning active.

MSIC2

Message Buffer Status Changed Interrupt Control 2

(03BC_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIP 63	MSIP 62	MSIP 61	MSIP 60	MSIP 59	MSIP 58	MSIP 57	MSIP 56	MSIP 55	MSIP 54	MSIP 53	MSIP 52	MSIP 51	MSIP 50	MSIP 49	MSIP 48
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIP 47	MSIP 46	MSIP 45	MSIP 44	MSIP 43	MSIP 42	MSIP 41	MSIP 40	MSIP 39	MSIP 38	MSIP 37	MSIP 36	MSIP 35	MSIP 34	MSIP 33	MSIP 32
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
MSIPn (n = 32-63)	n - 32	rh	<p>Message Buffer Status Changed Interrupt Pointer n (n = 32-63)</p> <p>MSIPn determines the interrupt (MBSC0SRC or MBSC1SRC) of the service request output that becomes active on a Message Buffer Status Changed Flag becoming active.</p> <p>0_B MBSC0SRC selected for Message Buffer Status Changed Service Request</p> <p>1_B MBSC1SRC selected for Message Buffer Status Changed Service Request</p>

FlexRay™ Protocol Controller (E-Ray)

Message Buffer Status Changed Interrupt Control 3 (MSIC3)

This Message Buffer Status Change Interrupt Control register controls the interrupt that becomes active (MBSC0SRC or MBSC1SRC) on a MBC flag of all configured Message Buffer 64 to Message Buffer 95 turning active.

MSIC3

Message Buffer Status Changed Interrupt Control 3

(03C0_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIP 95	MSIP 94	MSIP 93	MSIP 92	MSIP 91	MSIP 90	MSIP 89	MSIP 88	MSIP 87	MSIP 86	MSIP 85	MSIP 84	MSIP 83	MSIP 82	MSIP 81	MSIP 80
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIP 79	MSIP 78	MSIP 77	MSIP 76	MSIP 75	MSIP 74	MSIP 73	MSIP 72	MSIP 71	MSIP 70	MSIP 69	MSIP 68	MSIP 67	MSIP 66	MSIP 65	MSIP 64
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MSIPn (n = 64-95)	n - 64	rw	<p>Message Buffer Status Changed Interrupt Pointer n (n = 64-95)</p> <p>MSIPn determines the interrupt (MBSC0SRC or MBSC1SRC) of the service request output that becomes active on a Message Buffer Status Changed Flag becoming active.</p> <p>0_B MBSC0SRC selected for Message Buffer Status Changed Service Request</p> <p>1_B MBSC1SRC selected for Message Buffer Status Changed Service Request</p>

FlexRay™ Protocol Controller (E-Ray)

Message Buffer Status Changed Interrupt Control 4 (MSIC4)

This Message Buffer Status Change Interrupt Control register controls the interrupt that becomes active (MBSC0SRC or MBSC1SRC) on a MBC flag of all configured Message Buffer 96 to Message Buffer 127 turning active.

MSIC4

Message Buffer Status Changed Interrupt Control 4

(03C4_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSIP 127	MSIP 126	MSIP 125	MSIP 124	MSIP 123	MSIP 122	MSIP 121	MSIP 120	MSIP 119	MSIP 118	MSIP 117	MSIP 116	MSIP 115	MSIP 114	MSIP 113	MSIP 112
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSIP 111	MSIP 110	MSIP 109	MSIP 108	MSIP 107	MSIP 106	MSIP 105	MSIP 104	MSIP 103	MSIP 102	MSIP 101	MSIP 100	MSIP 99	MSIP 98	MSIP 97	MSIP 96
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MSIPn (n = 96-127)	n - 96	rw	<p>Message Buffer Status Changed Interrupt Pointer n (n = 96-127)</p> <p>MSIPn determines the interrupt (MBSC0SRC or MBSC1SRC) of the service request output that becomes active on a Message Buffer Status Changed Flag becoming active.</p> <p>0_B MBSC0SRC selected for Message Buffer Status Changed Service Request</p> <p>1_B MBSC1SRC selected for Message Buffer Status Changed Service Request</p>

25.10 Revision History

This User's Manual is based on the IP Specification: **Revision 1.2.6.**

Table 25-38 Revision History

Version Number	Changes to Previous Version
Rev_0.1	First Draft
Rev_0.2	Adapted to PWD 0.8
Rev_0.3	Chapters 3, 5, 6 completed
Rev_0.4	Adapted to actual state of protocol development
Rev_0.5	Adapted to actual state of protocol development
Rev_0.51	Adapted to actual state of protocol development
Rev_0.52	Adapted to actual state of protocol development
Rev_0.53	Adapted to actual state of protocol development
Rev_0.6	Adapted to actual state of protocol development
Rev_0.61	Adapted to actual state of protocol development
Rev_0.62	Adapted to actual state of protocol development
Rev_1.0	First complete revision
Rev_1.01	Message Buffer Status bits PLE, MLST, ES replaced by bits ESA, ESB, MLST
Rev_1.02	IBCR, IBCM, OBCR, OBCM: addresses changed HDC2: register removed MHDC1: renamed to MHDC Message Buffer 0 dedicated to hold key slot ID SFS: description updated ESIDn, OSIDn: description updated EIR: bit SCE removed EILS: bit SCEL removed EIES, EIER: bit SCEE removed

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.2	<p>IDEFAULT_CONFIG added to POC working</p> <p>CCSV: assignment of states to ^{POCS changed:}</p> <p>POCS = 00 0000_B = "DEFAULT_CONFIG"</p> <p>POCS = 00 1111_B = "CONFIG"</p> <p>CCSV: bit DCREQ removed</p> <p>SIR: bit MBSI added</p> <p>SILS: bit MBSIL added</p> <p>SIES, SIER: bit MBSIE added</p> <p>Register BGSC removed</p> <p>EIR: bits SMEB, SMEA removed</p> <p>EILS: bits SMEBL, SMEAL removed</p> <p>EIES, EIER: bits SMEBE, SMEAE removed</p> <p>Registers TXRQ3, TXRQ4, NDAT3, NDAT4, MBSC3, MBSC4 added</p> <p>Bus guardian related pins eray_arm, eray_bgt, eray_mt, eray_bge1, and eray_bge2 have no function</p> <p>PRTC1: Configuration parameter CASM added</p> <p>WRHS1: Bit NME changed to PPIT</p> <p>RDHS1: Bit NME changed to PPIT</p> <p>Pin eray_scanmode for scan mode control added</p>
Rev_1.3	<p>Changed "r" bits into "rh" bits.</p> <p>Made ACS Register writable.</p> <p>Included Reserved Bits into EIES and EIER</p> <p>Changed access write from "rw" to "rwh" for Register CUST1.IEN. Changed access type of unused bits (0) to "rw".</p> <p>Changed access type of CUST1.INT0 to rwh.</p> <p>Changed Reset value of CUST0 to C104 0105_H</p> <p>Updated the "Known Limitations of E-Ray IP-Module", revision 4.07.2005.</p>

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.4	<p>Updated the “Errata of E-Ray IP-Module”, revision 1.0 30.06.2006. Renamed from “Interrupt Flag Interface” to “Internal Signal and Flag Interface”</p> <p>With this revision it is possible to use Message Buffer 1 for SYNC Frame transmission in addition to Message Buffer 0 if SYNC Frames should have different payloads on channel A and B</p> <p>TEST1: Bit ELBE for control of internal / external loop back mode added, description of internal loop back added</p> <p>EIR: Handling of bits PERR and RFO same as for other bits, bit MHF added</p> <p>SIR: Bit RFF renamed to RFCL, handling of bits RFNE, RFCL same as for other bits</p> <p>EILS: Bit MHFL added, SILS: Bit RFFL renamed to RFCLL</p> <p>EIES, EIER: Bit MHFE added</p> <p>SIES, SIER: Bit RFFE renamed to RFCLE</p> <p>Register STPW renamed to STPW1</p> <p>STPW2: Register added</p> <p>CCSV: Bits PSL added</p> <p>SWNIT: Bits MTSA, MTSB added</p> <p>MRC: Bit SPLM added</p> <p>FCL: Register added</p> <p>FSR: Register added</p> <p>MHDF: Register added</p> <p>MBSC1/2/3/4: Naming of bits changed from MBS to MBC to distinguish between Message Buffer status flag (MBC) and Message Buffer status register (MBS)</p> <p>CREL: Register added</p> <p>ENDN: Register added</p> <p>Message Buffers in Message RAM: Header 2 and 3 updated from received Data Frames only</p> <p>MBS: Bits FTA, FTB, CCS, RCIS, SFIS, SYNS, NFIS, PPIS, RESS added</p> <p>Description Asynchronous Transmit Mode, added: ... This write operation has to be directly preceded by two consecutive write accesses to the Configuration LockKey (unlock sequence)...</p> <p>Description Loop Back Mode, added: ... This write operation has to be directly preceded by two consecutive write accesses to the Configuration LockKey (unlock sequence).</p>

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.4 (cont'd)	<p>Description LCK.CLK, corrected: To leave CONFIG state by writing SUCC1.CMD (commands READY, MONITOR_MODE, ATM_LOOP_BACK), ...Third write: SUCC1.CMD</p> <p>Description EIR.MHF, added: The flag signals a Message Handler constraints violation condition. It is set whenever one of the flags MHDF.SNUA, MHDF.SNUB, MHDF.FNFA, MHDF.FNFB, MHDF.TBFA, MHDF.TBFB, MHDF.WAHP changes from 0 to 1.</p> <p>Description SIR.CAS, added: This flag is set by the Communication Controller during STARTUP state when a CAS or a potential CAS was received. 1 = Bit pattern matching the CAS symbol received 0 = No bit pattern matching the CAS symbol received.</p> <p>Description SIR.MBSI, corrected: This flag is set by the Communication Controller when the Message Buffer status MBS has changed if bit MBI of that Message Buffer is set.</p> <p>Description T0C, added: Note: The configuration of timer 0 is compared against the Macrotick counter value, there is no separate counter for timer 0...</p> <p>Description SUCC1.CMD, modified: Note included into description of CHI command CLEAR_RAMs. ...Access to the configuration and status registers is possible during execution of CHI command CLEAR_RAMs...</p> <p>Description SUCC1.TSM, changed: ...The key slot ID is configured in the Header Section of Message Buffer 0 respectively Message Buffers 0 and 1 depending on bit MRC.SPLM. In case TSM = 1, Message Buffer 0 respectively Message Buffers 0,1 can be (re)configured in "DEFAULT_CONFIG" or "CONFIG" state only. In ALL slot mode the Communication Controller may transmit in all slots. TSM is a configuration bit which can only be set / reset by the Host. The bit can be written in "DEFAULT_CONFIG" or "CONFIG" state only. The Communication Controller changes to ALL slot mode when the Host successfully applied the ALL_SLOTS command by writing CMD = "0101" in POC states "NORMAL_ACTIVE" or "NORMAL_PASSIVE"...</p> <p>Description PRTC1.SPP, added: Note: The current revision 2.1 of the FlexRay™ protocol requires that SPP = "00". The alternate strobe point positions could be used to compensate for asymmetries in the physical layer.</p> <p>Description CCSV.CSNI, corrected: ...Reset by CHI command RESET_STATUS_INDICATORS or by transition from "HALT" to "DEFAULT_CONFIG" state or from "READY" to "STARTUP" state.</p>

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.4 (cont'd)	<p>Description CCSV.CSAI, corrected: ...Reset by CHI command RESET_STATUS_INDICATORS or by transition from "HALT" to "DEFAULT_CONFIG" state or from "READY" to "STARTUP" state.</p> <p>Description CCSV.WSV, corrected: ...Reset by CHI command RESET_STATUS_INDICATORS or by transition from "HALT" to "DEFAULT_CONFIG" state or from "READY" to "STARTUP" state.</p> <p>Description SFS.VSAE, modified: Holds the number of valid SYNC Frames received on channel A in the even communication cycle. If transmission of SYNC Frames is enabled by SUCC1.TXSY the value is incremented by one...</p> <p>Description SFS.VSAO, modified: Holds the number of valid SYNC Frames received on channel A in the odd communication cycle. If transmission of SYNC Frames is enabled by SUCC1.TXSY the value is incremented by one...</p> <p>Description SFS.VSBE, modified: Holds the number of valid SYNC Frames received on channel B in the even communication cycle. If transmission of SYNC Frames is enabled by SUCC1.TXSY the value is incremented by one...</p> <p>Description SFS.VSBO, modified: Holds the number of valid SYNC Frames received on channel B in the odd communication cycle. If transmission of SYNC Frames is enabled by SUCC1.TXSY the value is incremented by one...</p> <p>Description ACS, added: Note: The set condition of flags CIA and CIB is also fulfilled if there is only one single Frame in the slot and the slot boundary at the end of the slot is reached during the Frames channel idle recognition phase...</p> <p>Description MRC.SEC1:0, changed: ...Exception: In nodes configured for SYNC Frame transmission or for single slot mode operation Message Buffer 0 (and if SPLM = 1, also Message Buffer 1) is always locked</p> <p>Description MRC.SPLM, changed: This bit is only evaluated if the node is configured as sync node (SUCC1.TXSY = 1) or for single slot mode operation (SUCC1.TSM = 1)...</p> <p>Description MRC.SPLM, changed: Note: In case the node is configured as sync node (SUCC1.TXSY = 1) or for single slot mode operation (SUCC1.TSM = 1), Message Buffer 0 resp. 1 is reserved for SYNC Frames or single slot Frames and have to be configured with the node-specific key slot ID. In case the node is neither configured as sync node nor for single slot operation Message Buffer 0 resp. 1 is treated like all other Message Buffers.</p>

FlexRay™ Protocol Controller (E-Ray)
Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.4 (cont'd)	<p>Description Parity Check, changed: ...When a parity error occurs when the Message Handler reads a Frame with Network Management information (PPI = 1) from the Transient Buffer RAM A, B the corresponding Network Management vector register NMV1...3 is not updated from that Frame.</p> <p>Included TC1797 O Ports the Physical Layer Interface is connected to.</p> <p>Description MBS.MLST, added: The flag is set in case the Host did not read the message before the Message Buffer was updated from a received Data Frame... ..The flag is reset by a Host write to the Message Buffer via IBF or when a new message is stored into the Message Buffer after the Message Buffers ND flag was reset by reading out the Message Buffer via OBF.</p> <p>Description MBS, corrected: Note: The FlexRay™ protocol specification requires that FTA, and FTB can only be reset by the Host...</p> <p>Description NULL Frame Transmission, changed: ...In this case, no Message Buffer status MBS is updated.</p>
Rev_1.5	<p>Included BPI Register (Service Request Nodes: INT0SRC, INT1SRC, TINT0SRC, TINT1SRC, NDAT0SRC, NDAT0SRC, MBSC0SRC, MBSC1SRC, CLC).</p> <p>Included for New Data and Message Buffer Status Changed Flags eight Interrupt Select Register (NDIC0, NDIC1, NDIC2, NDIC3, MSIC0, MSIC1, MSIC2, MSIC3)</p> <p>Included new address range for TC1797.</p> <p>Included into CUST1 Register Control (IBF1PAG, IBF1PAG) and Status Bits (IBS) for multiple buffer control.</p> <p>Included for TC1797 the IO Ports the Physical Layer Interface is connected to.</p> <p>Included into CUST1 1 out of 4 multiplexer Select Input Control for RXA (CUST1.RISA), RXB (CUST1.RISB) and STPWT (CUST1.STPWTS).</p> <p>Included DMA Trigger connections for TC1797.</p>
Rev_1.6	<p>Included a description of the Delayed Write Scheme to OBF and IBF.</p> <p>Corrected an erroneous description of IBF1PAG and IBF2PAG concerning the write access for specific values of IBFS. Included a verification scheme to verify correct polarity of IBFS and logic for IBF1PAG and IBF2PAG.</p>
Rev_1.7	<p>Changed P6.7 to P6.10 to use only p6.PDR1 only.</p> <p>Included table summarizing access wait states.</p>
Rev_1.8	<p>Included a description of registering the reset when E-Ray module is disabled and executing the reset when it is enabled hereafter.</p>

FlexRay™ Protocol Controller (E-Ray)

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.9	<p>Included IP Specification changes of Revision 1.2.3.</p> <p>Description TEST1, added: When the E-Ray IP is operated in one of its test modes that requires WRTEEN to be set (RAM Test Mode, I/O Test Mode, Asynchronous Transmit Mode, and Loop Back Mode) only the selected test mode functionality is available. The test functions are not available in addition to the normal operational mode functions, they change the functions of parts of the E-Ray module. Therefore normal operation as specified outside this chapter and as required by the FlexRay™ protocol specification and the FlexRay™ conformance test is not possible. Test mode functions may not be combined with each other or with FlexRay™ protocol functions. The test mode features are intended for hardware testing or for FlexRay™ bus analyzer tools. They are not intended to be used in FlexRay™ applications.</p> <p>Description TEST1.ELBE, modified: ...Bit ELBE is evaluated only when POC is in loop back mode and test multiplexer control is in non-multiplexing mode TMC = 00.</p> <p>Description TEST1.TMC, modified: TMC Test Multiplexer Control 00, 11= Normal signal path (default); 01 = RAM Test Mode - Internal busses are multiplexed to make all RAM blocks of the E-Ray module directly accessible by the Host...</p> <p>Description Loop Back Mode, corrected: ...Reading CCSV.POCS will return "00 1101" while the E-Ray module is in loop back mode...</p> <p>Description Loop Back mode, completed: ...When the CC is in loop back mode, a loop back test is started by the Host writing a message to the Input Buffer and requesting the transmission by writing to register IBCR...</p> <p>Description LCK.CLK, corrected: ...If the write sequence below is interrupted by other write accesses between the second write to the Configuration Lock Key and the write access to the SUCC1 register, the Communication Controller remains in CONFIG state and the sequence has to be repeated.</p> <p>Description LCK.TMK, corrected: ...If the write sequence below is interrupted by other write accesses between the second write to the Test Mode Key and the write access to the TEST1 register, TEST1.WRTEEN is not set to '1' and the sequence has to be repeated.</p> <p>Description EIR.CCL, corrected: The flag signals that the write access to the CHI command vector SUCC1.CMD was not successful because the execution of the previous CHI command has not yet completed...</p> <p>Description SIR.MTSA,B, modified: Media Access Test symbol received on channel A,B during the preceding symbol window...</p>

FlexRay™ Protocol Controller (E-Ray)

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.9 (cont'd)	<p>Description T1C, removed: ... In case the configured Macro tick count is not within the valid range, timer 1 will not start...</p> <p>Description SUCC1.CMD, corrected: ...In case the previous CHI command has not yet completed, EIR.CCL is set to 1 together with EIR.CNA; the CHI command needs to be repeated. Except for HALT state, a POC state change command applied while the CC is already in the requested POC state will be ignored.</p> <p>Description SUCC1.CMD, added: ALLOW_COLDSTART; ...When called in states DEFAULT_CONFIG, CONFIG, HALT, or MONITOR_MODE, CMD will be reset to 0000 = command_not_accepted...</p> <p>Description SUCC1.CMD, completed: RESET_STATUS_INDICATORS; ...Flags internally evaluated in the actual POC state are not reset...</p> <p>Description SUCC1.CMD, corrected: MONITOR_MODE; ...In this mode the CC is able to receive FlexRay™ Frames and wakeup pattern...</p> <p>Description SUCC1.CMD, added: Table added which references the CHI commands from the FlexRay™ Protocol Specification v2.1 (section 2.2.1.1, Table 2-2) to the E-Ray CHI command vector CMD.</p> <p>Description CCSV.RCA, completed: ...The READY command resets this counter to the maximum number of coldstart attempts as configured by SUCC1.CSA.</p> <p>Description SWNIT.MTSA,B, modified: Media Access Test symbol received on channel A,B during the preceding symbol window...</p> <p>Description ESID[1...15].RXEA,B completed: ...sorted in ascending order,If the node transmits a SYNC Frame in an even communication cycle by itself, register ESID1 holds the respective SYNC Frame ID as configured in Message Buffer 0 and flags RXEA, RXEB are set... RXEA,B Received / Configured Even Sync ID on Channel A,B Signals that a SYNC Frame corresponding to the stored even sync ID was received on channel A or that the node is configured to be a sync node with key slot = EID (ESID1 only). 1 = SYNC Frame received on channel A / node configured to transmit SYNC Frames; 0 = No SYNC Frame received on channel A / node not configured to transmit SYNC Frames.</p>

FlexRay™ Protocol Controller (E-Ray)

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.9 (cont'd)	<p>Description OSID[1...15].RXOA,B completed: ... sorted in ascending order,If the node transmits a SYNC Frame in an odd communication cycle by itself, register OSID1 holds the respective SYNC Frame ID as configured in Message Buffer 0 and flags RXOA, RXOB are set...RXOA,B Received / Configured Odd Sync ID on Channel A,B Signals that a SYNC Frame corresponding to the stored odd sync ID was received on channel A or that the node is configured to be a sync node with key slot = OID(OSID1 only). 1 = SYNC Frame received on channel A / node configured to transmit SYNC Frames;0 = No SYNC Frame received on channel A / node not configured to transmit SYNC Frames</p> <p>Description FRF.FID, modified: Determines the Frame ID to be rejected by the FIFO. With the additional configuration of register FRFM, the corresponding Frame ID filter bits are ignored, which results in further rejected Frame IDs. When FRFM.MFID is zero, a Frame ID filter value of zero means that no Frame ID is rejected.</p> <p>Description MHDF.WAHP, added: Outside DEFAULT_CONFIG and CONFIG state this flag is set by the CC when the message handler tries to write message data into the Header Partition of the Message RAM due to faulty configuration of a Message Buffer...</p> <p>Description MONITOR_MODE, completed: ...In this mode the CC is able to receive FlexRay™ Frames and to detect wakeup pattern...</p> <p>Modified the description of CUST1.POBFEN to: Parity Error Reporting of Output Buffer (OBF1/OBF1) RAMs Enable/Test Disable; 0_B: Parity error in Output Buffer (OBF1,OBF2) RAMs is not reported to SCU nor to OBF1, OBF2 RAM wrapper.1_B: The parity error reporting and thus the activation of error signals to SCU as also to OBF1 and OBF2 RAM wrapper is enabled.</p> <p>Modified the description of CUST1.PMBFEN to: Parity Error Reporting of Message Buffer (MBF) RAMs Enable/Test Disable; 0_B: Parity error in Message Buffer (MBF) RAMs is not reported to SCU nor to MBF RAM wrapper.1_B: The parity error reporting and thus the activation of error signals to SCU as also to MBF RAM wrapper is enabled.</p> <p>Modified the description of CUST1.PITBFEN to: Parity Error Reporting of Message Buffer (IBF1, IBF2, TBF1, TBF2) RAMs Enable/Test Disable; 0_B: Parity error in Input Buffer and Transient Buffer (IBF1, IBF2, TBF1, TBF2) RAMs is not reported to SCU nor to IBF1, IBF2, TBF1, and TBF2 RAM wrapper.1_B: The parity error reporting and thus the activation of error signals to SCU as also to IBF1, IBF2, TBF1, and TBF2 RAM wrapper is enabled</p>

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_2.0	<p>Modified the address of INT0SRC, INT1STC, TINT0SRC, TINT1SRC, NDIC1...NDIC4, MSIC1...MSIC4, NDAT_NDAT0SRC, NDAT1SRC, MBSC0SRC, MBSC1SRC.</p> <p>Cleared up the Index entries.</p> <p>Changed Address Range from to 00000000_H ...00007FFF_H to F0010000_H ...F0017FFF_H</p>
Rev_2.1	<p>Inserted additional information on the minimal eray_bclk required to work properly.</p> <p>Corrected some Typos in the port implementation.</p> <p>Renamed the Interrupt Control Registers due to some problems extracting register names by automatic tools.</p> <p>Changed in Figure 2-1 the names of the clock signals.</p> <p>Changed the Long Name of Register ACS to Aggregated Channel Status Register.</p> <p>Included Message Handler Timing Details.</p> <p>Modified the Clock Signal Names in Figure 2-37.</p>
Rev_2.2	<p>Modified the Pinning of TC1797: TXDA: P0.14, RXDA:P0.9; TXENA: P0.10; TXDB: P0.12; RXDB: P0.12; TXENB: P0.11. Changed the name of the Bit RFCLL in SILS Register.</p>
Rev_2.3	<p>Included IP Specification changes of Revision 1.2.3.</p> <p>Description write access to Data Section of IBF, changed: ...If not all bytes of a 32-bit word have been written by the Host (8/16-bit access only), WRDSn holds partly undefined data.</p> <p>Description SIR.WST, completed: This flag is set when the wakeup status vector CCSV.WSV is changed by a protocol event.</p> <p>Description SIR.SWE, modified: This flag is set after a stop watch activation when the actual cycle counter and Macroctick value are stored in the Stop Watch register (see section Stop Watch Register 1 (STPW1)).</p> <p>Description STPW1.ESWT, corrected: ...In single-shot mode this bit is reset to 0 after the actual cycle counter and Macroctick value are stored in the Stop Watch register.</p> <p>Description SUCC1.CMD3:0, changed: RESET_STATUS_INDICATORS Resets status flags CCSV.CSNI, CCSV.CSAI, and CCSV.WSV to their default values. May be called in POC state "READY". When called in any other state, CMD will be reset to 0000_B = command_not_accepted.</p> <p>Description CCSV.POC, added: ...101011_B = STARTUP_SUCCESS state 101100_B ... 111111_B = reserved</p>

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_2.3	<p>Description SUCC1.MTSA,B, added: Note: MTSA,B may also be changed outside “DEFAULT_CONFIG” or “CONFIG” state when the write to SUCC1 register is directly preceded by the unlock sequence as described in chapter Lock Register (LCK). This may be combined with CHI command SEND_MTS...</p> <p>Description Communication Controller Status Registers, added: ...The status vector may change faster than the Host can poll the status vector, depending on eray_bclk frequency.</p> <p>Description Communication Controller Status Registers, removed: ...All internal counters and the Communication Controller status flags are reset when the Communication Controller transits from “CONFIG” to “READY” state.</p> <p>Description CCSV.FSI, changed: ...Reset by transition from HALT to “DEFAULT_CONFIG” state.</p> <p>Description CCSV.HRQ, changed: ...Reset by transition from HALT to DEFAULT_CONFIG state or when entering READY state.</p> <p>Description CCSV.SLM, changed: ...Set to the value defined by SUCC1.TSM when entering “READY”, “CONFIG”, or “DEFAULT_CONFIG” state.</p> <p>Description CCSV.CSI, completed: ...The flag is set whenever the POC enters READY state due to CHI command READY.</p> <p>Description CCSV.WSV, modified: ...000_B = UNDEFINED. Wakeup not yet executed by the Communication Controller...</p> <p>Description CCSV.RCA, corrected: ...The “RUN” command resets this counter to the maximum number of coldstart attempts as configured by SUCC1.CSA.</p> <p>Description Communication Controller Status Registers, removed: ...Note: CHI command “RESET_STATUS_INDICATORS” (SUCC1.CMD = 1010_B) resets flags FSI, HRQ, CSNI, CSAI, the slot mode SLM, and the wakeup status WSV.</p> <p>Description CC Error Vector, changed: Reset by transition from “HALT” to “DEFAULT_CONFIG” state or when entering “READY” state...</p> <p>Description MONITOR_MODE, added: ...In “MONITOR_MODE” the Communication Controller is not able to distinguish between CAS and MTS symbols. In case one of these symbols is received on one or both of the two channels, the flags SIR.MTSA resp. SIR.MTSB are set. SIR.CAS has no function in “MONITOR_MODE”.</p> <p>Inserted remark to negate eray_ibusy and eray_obusy signal to simplify the usage for DMA requesting and interrupt triggering.</p> <p>Corrected Typo in CMD field of SUCC1.</p>

FlexRay™ Protocol Controller (E-Ray)

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_2.4	Updated ERAY Kernel Errata List (Bosch Errata List from November 24th, 2006: "Errata_E-Ray_R1.0_20061124")
Rev_2.5	<p>Modified RISA: Receive Input Select Channel A 00_B Channel A receiver input RXDA0 selected 01_B Channel A receiver input RXDA1 selected 10_B Channel A receiver input RXDA2 selected 11_B Channel A receiver input RXDA3 selected</p> <p>Modified RISB: Receive Input Select Channel B 00_B Channel B receiver input RXDB0 selected 01_B Channel B receiver input RXDB1 selected 10_B Channel B receiver input RXDB2 selected 11_B Channel B receiver input RXDB3 selected</p> <p>Description MONITOR_MODE, removed: ...In "MONITOR_MODE" the receive FIFO is not available. Description Message RAM Configuration Register, added: ... Note: ... In case two or more Message Buffers are assigned to slot 1 by use of cycle filtering, all of them must be located either in the "Static Buffers" or at the beginning of the "Static + Dynamic Buffers" section... Description CCSV.POC[5:0], added: ...101011_B = "STARTUP_SUCCESS" state 101100_B...111111_B = reserved. Inserted text condition to commonly cover IP revision 1.0.0 and IP revision 1.0.1. Updated ERAY Kernel Errata List (Bosch Errata List from December 20th, 2006: "Errata_E-Ray_R1.0.0_20061220"). Included IP Specification changes of Revision 1.2.5. Description EIR.SFO, changed: Set when either the number of SYNC Frames received during the last communication cycle or the total number of SYNC Frames received during the last double cycle exceeds the maximum number of SYNC Frames as defined by GTUC2.SNM. Description SUCC1.CMD[3:0], changed: RESET_STATUS_INDICATORS ... May be called in POC states "READY" and "STARTUP"...</p>

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_2.6	<p>Description CCSV.SLM, changed: Indicates the actual slot mode of the POC in states “READY”, “STARTUP”, “NORMAL_ACTIVE”, and “NORMAL_PASSIVE”. Default is SINGLE. Changes to ALL, depending on SUCC1.TSM. In “NORMAL_ACTIVE” or “NORMAL_PASSIVE” state the CHI command ALL_SLOTS will change the slot mode from SINGLE over ALL_PENDING to ALL. Set to SINGLE in all other states.</p> <p>Description SFS.VSAE, removed: ... (vSyncFramesEvenA)</p> <p>Description SFS.VSAO, removed: ... (vSyncFramesOddA)</p> <p>Description SFS.VSBE[3:0], removed:... (vSyncFramesEvenB)</p> <p>Description SFS.VSBO[3:0], removed: ... (vSyncFramesOddB)</p> <p>Description MBS.RCIS, MBS.SFIS, MBS.SYNS, MBS.NFIS, MBS.PPIS, MBS.RESS, completed: For receive buffers (CFG = 0) the following status bits are updated from both valid data and NULL Frames. If no valid Frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.</p> <p>Description Network Management, added: Note: ...When the Communication Controller transits to “HALT” state, the cycle count is not incremented and therefore the NM Vector is not updated. In this case NMV1...3 holds the value from the cycle before.</p> <p>Description Configuration of the FIFO, added: Note: It is recommended to program the MBI bits of the Message Buffers belonging to the FIFO to 0 via WRHS1.MBI to avoid generation of RX interrupts...</p> <p>Updated the CLC reset value to 0000 0100_H.</p> <p>Included the ERAY trigger signals to the DMA channels 10 -17.</p> <p>Modified the stop watch clock input. Shifted them from GOUT to PDOUT.</p> <p>Removed a description of the non implemented fractional divider.</p> <p>Updated ERAY Kernel Errata List (Bosch Errata List from February 19th, 2007: “E-Ray_Errata_Sheet_20070219”).</p> <p>Connected Macrotick signal to ERU input IN23.</p> <p>Modified the DMA connection. Included IBUSY and OBUSY and TINT0.</p> <p>Included Service Request node for IBUSY and OBUSY.</p> <p>Modified the addresses of NDIC1, NDIC2, NDIC3, NDIC4, MSIC1, MSIC2, MSIC3, and MSIC4 to have a continues address range for the new service request registers OBUSYSRC and IBUSYSRC.</p>
Rev_2.7	<p>Removed Parity Bit generation and checking</p> <p>Added ECC generation and checking</p> <p>Wording and typos correction</p> <p>Signal name clean up (e.g. clock signals, ERAY internal signals etc.)</p> <p>CLC is endinit protected</p>

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_3.0	<p>Included IP Specification changes of Revision 1.2.6 Inserted text condition to commonly cover IP revision 1.0.0, IP revision 1.0.1 and IP revision 1.0.2. It is not possible to change the Stop Watch Mode (STPW1.SWMS) during enabled stop watch trigger (STPW1.ESWT) Implemented AI00042258. It is not possible to distinguish between static and dynamic frames, because limited functions in Monitor Mode (FRF.RSS will be ignored, filtering not functional). Added note in TEST1L/H register about the change of TEST1H.CERA/B if accessing TEST1H or L Reset value for KSCFG changed to 0x0001_h. Was the initial value 0x0000_h. ECC control and status registers added (SEC_CON, SED_CON, DED_CON, ECCR, ECCW) Removed ECC control form CUST1 Removed ECC test description from Register TEST2 Address space enlarged from 2k to 4k Figure E-Ray Register Map contains the new registers</p>
Rev_3.1	<p>ECC control and status registers renamed (SEC_CON -> SECCON, SED_CON -> SEDCON, DED_CON -> DEDCON) Corrected RDHS3H.0 length field Renamed STPW1, 2, 3, 4 to STPWT0, 1, 2, 3 16-Bit: Detailed KSCFG functionality MT connected to ERU_3B2 (was ERU_3A2 in V3.0) 16-Bit: KSCFG moved to 0x0000h 16-Bit: MT connected to ERU_3B2 (was ERU_3A2 in V3.0) Completed Register Table with page numbers Corrected address spaces that are reserved for customer specific purposes</p>
Rev_3.2	<p>16-bit: "Implement DEDCON registers" "Description of register SECCONL, SEDCON, DEDCON": changed TBFA/B to TBF1/2 implemented individual ECC control over each RAM + implement KSCFG register</p>
Rev_3.3	<p>ECC: "Define Signal WRECC in E-Ray Register TEST1" with modification: TEST2 used like in original implementation for parity 32-Bit: ECC: Reset Values of SECCON, SEDCON, DEDCON = 0x007Fh = enabled</p>

Table 25-38 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_3.4	16-bit: “Specify and implement Module ID Register for FlexRay like TC1797” 16-bit: SECCON, SEDCON, DEDCON = 0x0000h = disabled 16-bit: Layout and pagination changes for User Manual (no Change Bars) wording corrected for User Manual (with Change Bars) add High Registers to Register Table correct all markers for Index / Appendix corrected listing of all reserved addresses
Rev_3.5	16-bit: Hint, to reset IBUSY and OBUSY before enabling these interrupts Added detailed figure of implementation Added overview figure of module
Rev_3.6	TXDA, TXDB and TXEN no longer enumerated (docu fix) Typos fixed Remove Examples
Rev_3.7	Markers for key-word -index reduced to 3 levels
Rev_3.8	port mapping added 32-bit: CLC description corrected: “requesting the HW to clear set bit CLC.DISS” 32-bit: added hint about initial value of CLC 16-bit: ERU_3B2 used for FR_MT and not ERU3A2 corrected IOCR settings in Table of Port connections.
Rev_3.9	32-bit: added BPI Protection and Module Reset 32-bit: removed SRCs
Rev_3.10	32-bit: added trace interface 32-bit: added tagging
Rev_3.11	32-bit: updated BPI Registers Chapter
Rev_3.12	32-bit: changed MHDS.CRAM reset value to ‘0’ added: Reset Value CUST1.IBFS changes after 2 clock cycles from ‘0’ to ‘1’. added: Reset Value SUCC1.PBUSY changed from ‘1’ to ‘0’ changed base address to F001C000
Rev_3.13	32-bit: added port mapping added IR wiring, added SMU interconnection
Rev_3.14	32-bit: added 2nd Kernel (ERAY0 and ERAY1).
Rev_3.15	32-bit: added: MT output connection to GTM

FlexRay™ Protocol Controller (E-Ray)**Table 25-38 Revision History (cont'd)**

Version Number	Changes to Previous Version
Rev_3.16	32-bit: removed OCS.TGS, TGB, TB_P. Changed OCS description to current implementation. Removed OTSS and OTGB description.
Rev_3.17	32-bit: updated pin out description (typo fix)

26 Generic Timer Module (GTM)

26.1 Introduction

This document is based on the GTM_IP Specification version 1.5.5.1 (Released on 20.06.2013) of the Robert Bosch GmbH.

26.1.1 Overview

This document is the specification for the Generic Timer Module (GTM). It contains a module framework with submodules of different functionality. These submodules can be combined in a configurable manner to form a complex timer module that serves different application domains and different classes within one application domain. Because of this scalability and configurability the timer is called generic.

The scalability and configurability is reached with an architecture philosophy where dedicated hardware submodules are located around a central routing unit (called Advanced Routing Unit (ARU)). The ARU can connect the submodules in a flexible manner. The connectivity is software programmable and can be configured during runtime.

Nevertheless, the GTM is designed to unload the CPU or a peripheral core from a high interrupt load. Most of the tasks inside the GTM can run -once setup by an external CPU-independent and in parallel to the software. There may be special situations, where the CPU has to take action but the goal of the GTM design was to reduce these situations to a minimum.

The hardware submodules have dedicated functionalities, e.g. there are timer input modules where incoming signals can be captured and characterized together with a notion of time. By combination of several submodules through the ARU complex functions can be established. E.g. the signals characterized at an input module can be routed to a signal processing unit where an intermediate value about the incoming signal frequency can be calculated.

The modules that help to implement such complex functions are called infrastructural components further on. These components are present in all GTM variants.

Other submodules have a more general architecture and can fulfil typical timer functions, e.g. there are PWM generation units. The third class of submodules are those fulfilling a dedicated functionality for a certain application domain, e.g. the DPLL serves engine management applications. A fourth group of submodules is responsible for supporting the implementation of safety functions to fulfil a defined safety level. The module ICM is responsible for interrupt services and defines the fifth group.

Generic Timer Module (GTM)

Each GTM is build up therefore with submodules coming from those four groups. The application class is defined by the amount of components of those submodules integrated into the implemented GTM.

26.1.2 Document Structure

The structure of this document is motivated out of the aforementioned submodule classes. [Section 26.2](#) describes the dedicated GTM implementation this specification is written for. It gives an overview about the implemented submodules.

[Section 26.3](#) up to [Section 26.9](#) deal with the so called infrastructural components for routing, clock management and common time base functions. [Section 26.10](#) to [Section 26.12](#) describe the signal input and output modules while the following [Section 26.13](#) explains the signal processing and generation submodule. [Section 26.14](#) outlines a memory configuration module for the described signal processing and generation submodule. The next sections provide a detailed description of application specific and safety related modules like the MAP, DPLL, SPE, CMP and MON submodules. [Section 26.18](#) describes a module that bundles several interrupts coming from the other submodules and connect them to the outside world.

These submodule groups are shown in the following table:

Table 26-1 Submodule groups

Section	Submodule	Group
Section 26.3	Advanced Routing Unit (ARU)	Infrastructural components
Section 26.4	Broadcast Module (BRC)	Infrastructural components
Section 26.5	First In First Out Module (FIFO)	Infrastructural components
Section 26.6	AEI-to-FIFO Data Interface (AFD)	Infrastructural components
Section 26.7	FIFO-to-ARU Interface (F2A)	Infrastructural components
Section 26.8	Clock Management Unit (CMU)	Infrastructural components
Section 26.9	Time Base Unit (TBU)	Infrastructural components
Section 26.10	Timer Input Module (TIM)	IO Modules
Section 26.11	Timer Output Module (TOM)	IO Modules
Section 26.12	ARU-connected Timer Output Module (ATOM)	IO Modules
Section 26.13	Multi Channel Sequencer (MCS)	Signal generation and processing

Generic Timer Module (GTM)
Table 26-1 Submodule groups (cont'd)

Section	Submodule	Group
Section 26.1 4	Memory Configuration Module (MCFG)	Infrastructural component for MCS
Section 26.1 5	TIM0 Input Mapping Module (MAP)	Dedicated
Section 26.1 6	Digital PLL (DPLL)	Dedicated
Section 26.1 7	Sensor Pattern Evaluation Module (SPE)	BLDC support
Section 26.1 8	Interrupt Concentrator Module (ICM)	Interrupt services
Section 26.1 9	Output Compare Unit (CMP)	Safety features
Section 26.2 0	Monitoring Unit (MON)	Safety features

26.2 GTM Architecture

26.2.1 Overview

As already mentioned in [Section 26.1](#) the GTM forms a generic timer platform that serves different application domains and different classes within these application domains. Depending on these multiple requirements of application domains multiple device configurations with different count of submodules (i.e. ATOM, BRC, MCS, PSM, SPE, TIM, TOM) are possible. In this section the GTM realization is outlined. The architecture of the GTM is depicted in.

26.2.1.1 GTM Architecture Block Diagram

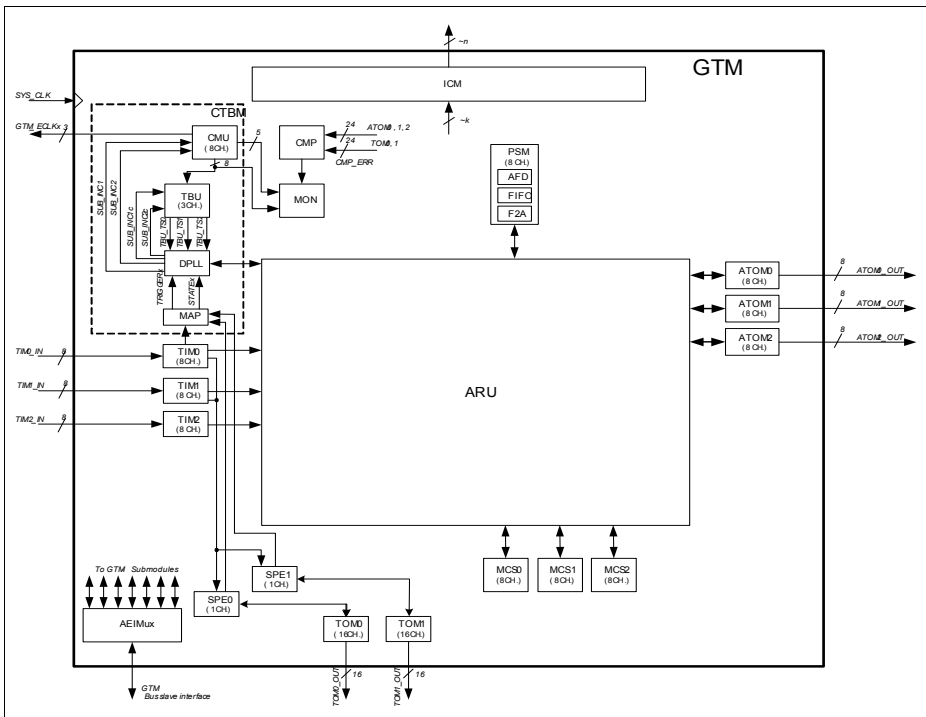


Figure 26-1 GTM Architecture Block Diagram

Generic Timer Module (GTM)

The central component of the GTM is the Advanced Routing Unit (ARU) where most of the submodules are located around and connected to. This ARU forms together with the Broadcast (BRC) and the Parameter Storage Module (PSM) the infrastructural part of the GTM. The ARU is able to route data from a connected source submodule to a connected destination submodule. The routing is done in a deterministic manner with a round-robin scheduling scheme of connected channels which receive data from ARU and with a worst case round-trip time.

The routed data word size of the ARU is 53 bit. The data word can logically be split into three parts. These parts are shown in **Figure 26-2**. Bits 0 to 23 and bits 24 to 47 typically hold data for the operation registers of the GTM. This can be for example the duty cycle and period duration of a measured PWM input signal or the output characteristic of an output PWM to be generated. Another possible content of Data0 and Data1 can be two 24 bit values of the GTM time bases TBU_TS0, TBU_TS1 and TBU_TS2. Bits 48 to 52 can contain control bits to send control information from one submodule to another. These ARU Control Bits (ACB) can have a different meaning for different submodules.

It is also possible to route data from a source to a destination and the destination can act later on as source for another destination. These routes through the GTM are further on called data streams. For a detailed description of the ARU submodule please refer to **Section 26.3**.

26.2.1.2 ARU Data Word Description

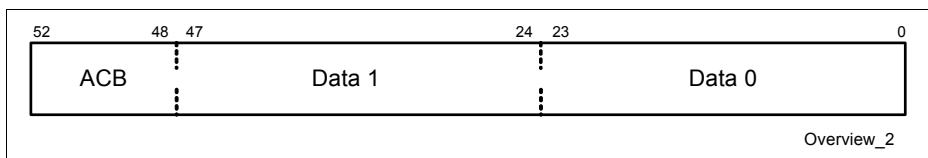


Figure 26-2 ARU Data Word Description

The BRC is able to distribute data from one source module to more than one destination modules connected to the ARU. The PSM submodule consists of three subunits, the AEI-to-FIFO Data Interface (AFD), FIFO-to-ARU Interface (F2A) and the FIFO itself. The PSM can serve as a data storage for incoming data characteristics or as parameter storage for outgoing data. This data is stored in a RAM that is logically located inside the FIFO subunit. The AFD subunit is the interface between the FIFO and the GTM system bus interface AEI (please see **Section 26.2.2.1** for detailed discussion). The F2A subunit is the interface between the FIFO subunit and the ARU.

Signals are transferred into the GTM at the Timer Input Modules (TIM). These modules are able to filter the input signals and annotate additional information. Each channel is for example able to measure pulse high or low times and the period of a PWM signal in

Generic Timer Module (GTM)

parallel and route the values to ARU for further processing. The internal operation registers of the TIM submodule are 24 bits wide.

The Clock Management Unit (CMU) serves up to 13 different clocks for the GTM and up to three external clock pins `GTM_ECLK0...2`. It acts as a clock divider for the system clock. The counters implemented inside other submodules are typically driven from this submodule. Please note, that the CMU clocks are implemented as enable signals for the counters while the whole system runs with the GTM global clock `SYS_CLK`. This global clock typically corresponds to the microcontroller bus clock the GTM is connected to.

The TBU provides up to three independent common time bases for the GTM. Two of these time bases can also be clocked with the digital PLL (DPLL) `sub_inc1c` and `sub_inc2c` outputs. The DPLL generates the higher frequent clock signals `sub_inc1`, `sub_inc2`, `sub_inc1c` and `sub_inc2c` on behalf of the frequencies of up to two input signals. These two input signals can be selected out of six incoming signals from the TIM0 submodule. In this submodule the incoming signals are filtered and transferred to the MAP submodule where two of these six signals are selected for further processing inside the DPLL.

Signal outputs are generated with the Timer Output Modules (TOM) and the ARU-connected TOMs (ATOM). Each TOM channel is able to generate a PWM signal at its output. Because of the integrated shadow register even the generation of complex PWM outputs is possible with the TOM channels by serving the parameters with the CPU. It is possible to trigger TOM channels for a successor TOM submodule through a trigger line between `TOM(x)_CH(15)` and `TOM(x+1)_CH(0)`. But to avoid long trigger paths the trigger signal is registered at the TOM submodule output, resulting in one `SYS_CLK` cycle delay of the trigger signal.

In addition each TOM submodule can integrate functions to drive one BLDC engine. This BLDC support is established together with the TIM and Sensor Pattern Evaluation (SPE) submodule.

The ATOMs offer the additional functionality to generate complex output signals without CPU interaction by serving these complex waveform characteristics by other submodules that are connected to the ARU like the PSM or Multi Channel Sequencer (MCS). While the internal operation and shadow registers of the TOM channels are 16 bit wide, the operation and shadow registers of the ATOM channels are 24 bit wide to have a higher resolution and to have the opportunity to compare against time base values coming from the TBU.

It is possible to trigger ATOM channels for a successor ATOM submodule through a trigger line between `ATOM(x)_CH(7)` and `ATOM(x+1)_CH(0)`. But to avoid long trigger paths the trigger signal is registered at each second ATOM submodule in the trigger line, resulting in one `SYS_CLK` cycle delay of the trigger signal between each second ATOM submodule.

Together with the MCS the ATOM is able to generate an arbitrary predefined output sequence at the GTM output pins. The output sequence is defined by instructions

Generic Timer Module (GTM)

located in RAM connected to the MCS submodule. The instructions define the points where an output signal should change or to react on other signal inputs. The output points can be one or two time stamps (or even angle stamp in case of an engine management system) provided by the TBU. Since the MCS is able to read data from the ARU it is also able to operate on incoming data routed from the TIM. Additionally, the MCS can process data that is located in its connected RAMs. The MCS RAM is located logically inside the MCS while the silicon vendor has to implement its own RAM technology there.

The two modules Compare Module (CMP) and Monitor Module (MON) implement safety related features. The CMP compares two output channels of an ATOM or TOM and sends the result to the MON submodule where the error is signalled to the CPU. The MON module is also able to monitor the ARU and CMU activities.

In the described implementation the submodules of the GTM have a huge amount of different interrupt sources. These interrupt sources are grouped and concentrated by the Interrupt Concentrator Module (ICM) to form a much easier management bunch of interrupts that are visible outside of the GTM.

On the GTM toplevel there are some configurable signal connections from the signal output of the TOM, ATOM modules to the input signals of the TIM modules.

26.2.1.3 GTM signal multiplex

Generic Timer Module (GTM)

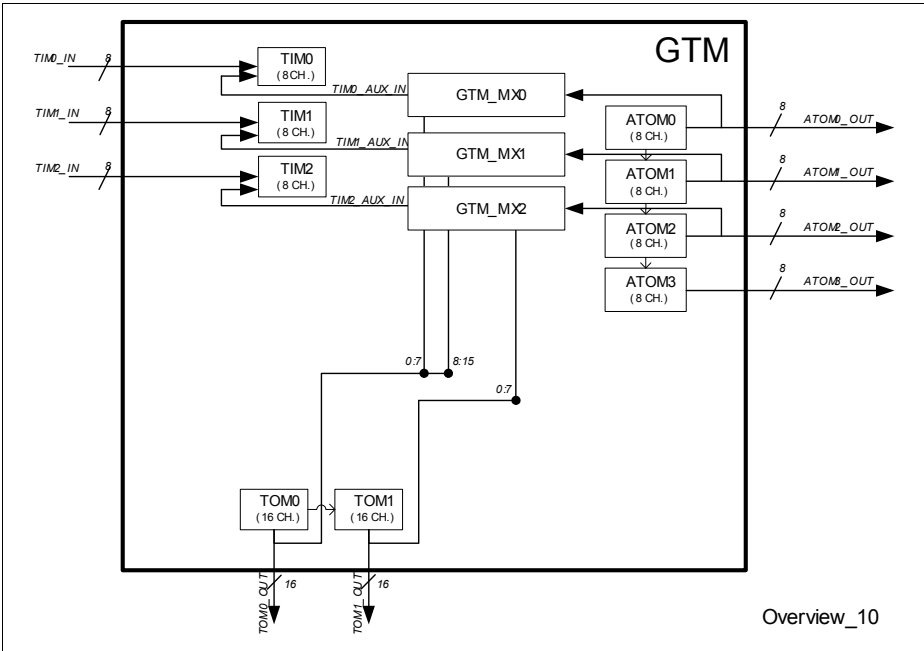


Figure 26-3 GTM Signal Multiplexer

The next diagram gives an overview of the connectivity. The source selection is defined with the bit SRXx in the register GTM_TIM[y]_AUX_IN_SRC.

Generic Timer Module (GTM)

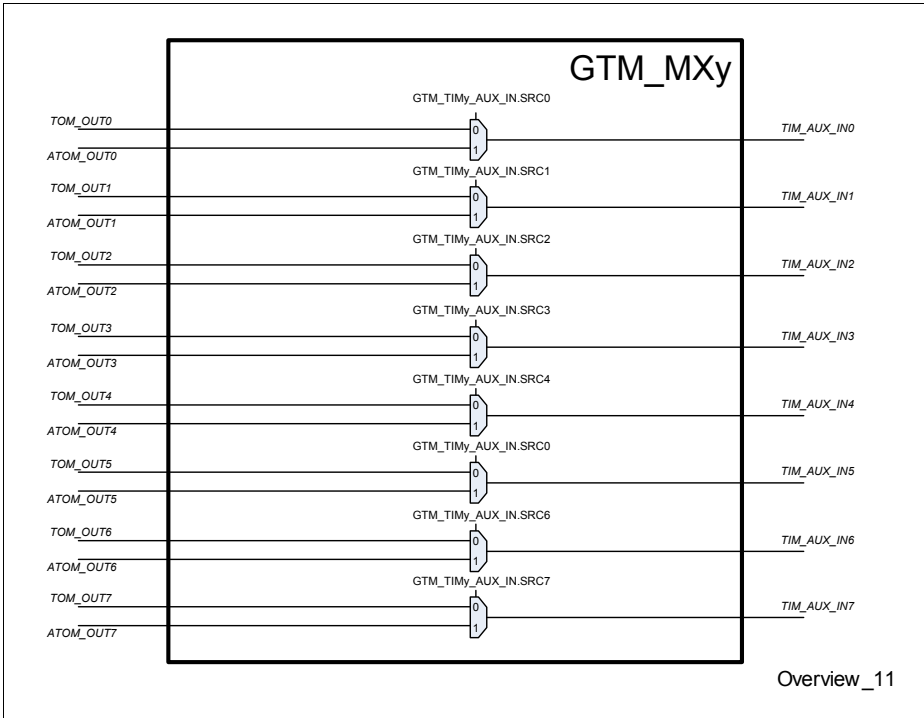


Figure 26-4 GTM TOM / ATOM to TIM Multiplexer

26.2.2 GTM Interfaces

In general the GTM can be divided into four interface groups. Two interface groups represent the ports of the GTM where incoming signals are assembled and outgoing signals are created. These interfaces are therefore connected to the GTM input submodule TIM and to the GTM output submodules TOM and ATOM.

Another interface is the bus interface where the GTM can be connected to the SoC system bus. This generic bus interface is described in more detail in [Section 26.2.2.1](#). The last interface is the interrupt controller interface. The GTM provides several interrupt lines coming from the various submodules. These interrupt lines are concentrated inside the ICM and have to be adapted to the dedicated microcontroller environment where each interrupt handling can look different. The interrupt concept is described in more detail in [Section 26.2.5](#).

Generic Timer Module (GTM)
26.2.2.1 GTM Generic Bus Interface (AEI)

The GTM is equipped with a generic bus interface that can be widely adapted to different SoC bus systems. This generic bus interface is called AE-Interface (AEI). The adaptation of the AEI to SoC buses is done with a bridge module translating the AEI signals to the SPB bus signals. The AEI bus signals are depicted in the following table:

Table 26-2 AEI Bus Signals

Signal name	I/O	Description	Bit width
AEI_SEL	I	GTM select line	1
AEI_ADDR	I	GTM address	32
AEI_PIPE	I	AEI Address phase signal	1
AEI_W1R0	I	Read/Write access	1
AEI_WDATA	I	Write data bus	32
AEI_RDATA	O	Read data bus	32
AEI_READY	O	Data ready signal	1
AEI_STATUS	O	AEI Access status	2

The AEI Status Signal may drive on of the following values:

Table 26-3 AEI Status Options

AEI_STATUS	Description
00	No Error
01	Illegal Byte Addressing
10	Illegal Address Access
11	Unsupported Address

The signal value "00" is driven if no error occurred during AEI access.

The signal value "01" is driven if the bus address is not an integer multiple of 4 (byte addressing).

The signal value "11" is driven if the address is not handled in the GTM.

The signal value "10" is driven if an illegal write access to one of the following register is performed

a) register is protected (e.g. protected by bit RF_PROT).

In case of an illegal write access signaled by status "10" the register will not be modified.

Reading registers will never return status "10".

Generic Timer Module (GTM)

Write access to following addresses returns status "10" under special conditions:

ARU_IRQ_FORCINT
ATOMi_CHx_CM0
ATOMi_CHx_CM1
ATOMi_CHx_SR0
ATOMi_CHx_SR1
ATOMi_CHx_RDADDR
ATOMi_CHx_IRQ_FORCINT
BRC_IRQ_FORCINT
BRC_SRCx_ADDR (x=0...11)
CMP_IRQ_FORCINT
CMU_GCLK_NUM
CMU_GCLK_DEN
CMU_CLKx_CTRL (x = 0...7)
CMU_ECLK_NUM
CMU_ECLK_DEN
CMU_FXCLK_CTRL
DPLL_ACT_STA
DPLL_OSW
DPLL_IRQ_FORCINT
DPLL_ID_PMTRx (x = 0...23)
DPLL_INC_CNT1
DPLL_INC_CNT2
DPLL_TSACx (x = 0...23)
DPLL_PSACx (x = 0...23)
DPLL_ACBx (x = 0...5)
DPLL_TSAx
F2A_CHx_ARU_RD_FIFO
F2A_CHx_STR_CFG
FIFOi_CHx_IRQ_FORCINT
GTM_IRQ_FORCINT
GTM_RST
SPEi_IRQ_FORCINT
TIMi_CHx_CNTS

TIMi_CHx-GPR1
 TIMi_CHx_IRQ_FORCINT
 TOMi_CHx_IRQ_FORCINT
 MCSi_CHx_CTRL
 MCSi_CHx_PC
 MCSi_CHx_IRQ_FORCINT
 MCSi_CTRL
 TBU_CHx_BASE
 TBU_CHx_CTRL
 MCS RAM during initialization
 DPLL RAM during initialization

26.2.2.2 GTM Multi-master and multi-tasking support

To support multi-master and multi-task access to the registers of the GTM a dedicated write-access scheme is used for critical control bits inside the GTM that need such a mechanism. This can be for example a shared register where more than one channel can be controlled globally by one register write access. Such register bits are implemented inside the GTM with a double bit mechanism, where the writing of '00' and '11' has no effect on the register bit and where '01' sets the bit and '10' resets the bit. If the CPU wants to read the status of the bit it always gets a '00' if the bit is reset and it gets a '11' if the bit is set.

Note: CPU accesses have to consider the configured SPB to SYS_CLK ratio if $SPB > SYSCLK$.

26.2.3 ARU Routing Concept

One central concept of the GTM is the routing mechanism of the ARU submodule for data streams. Each data word transferred between the ARU and its connected submodule is 53 bit wide. It is important to understand this concept in order to use the resources of the GTM effectively. Each module that is connected to the ARU may provide an arbitrary number of ARU write channels and an arbitrary number of ARU read channels. In the following, the ARU write channels are named data sources and the ARU read channels are named data destinations.

The concept of the ARU intends to provide a flexible and resource efficient way for connecting any data source to an arbitrary data destination. In order to save resource costs, the ARU does not implement a switch matrix, but it implements a data router with serialized connectivity providing the same interconnection flexibility. **Figure 26-5** shows the ARU data routing principle. Data sources are marked with a green rectangle and the data destinations are marked with yellow rectangles. The dashed lines in the ARU depict

Generic Timer Module (GTM)

the configurable connections between data sources and data destinations. A connection between a data source and a data destination is also called a data stream.

26.2.3.1 Principle of data routing using ARU

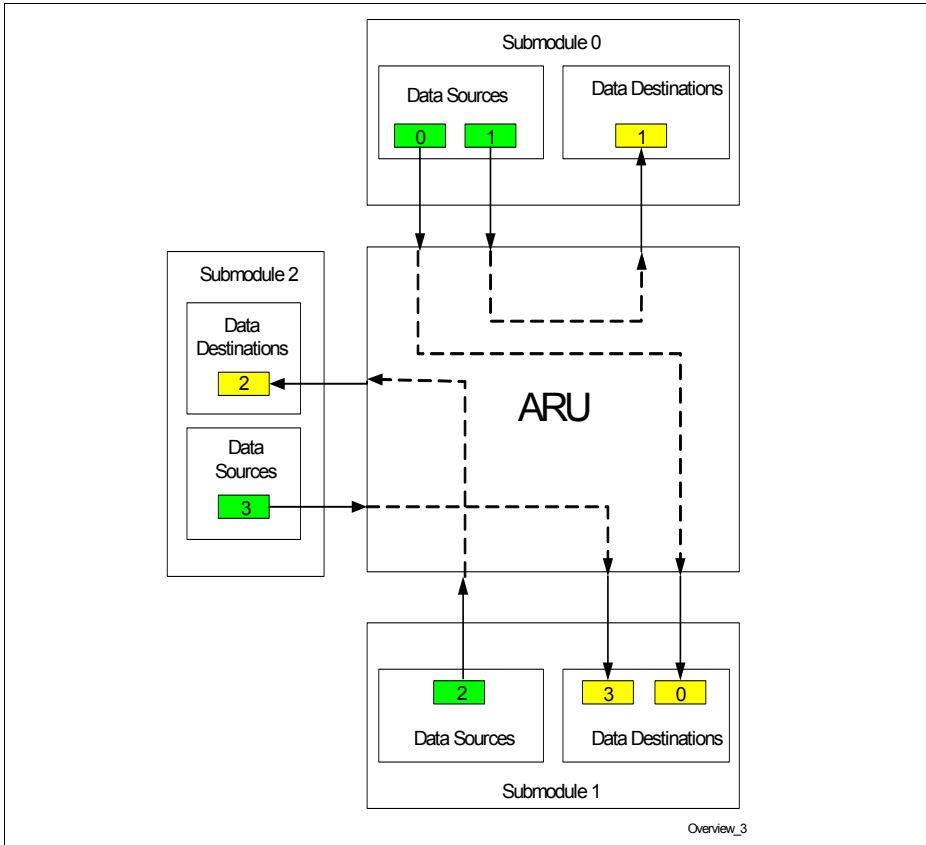


Figure 26-5 ARU Data Routing Principle

The configuration of the data streams is realized according to the following manner: Each data source has its fixed and unique source address: The fixed address of each data source is pointed out by the numbers in the green boxes of **Figure 26-5**. The address definitions of all available data sources in the GTM can be obtained from **Table 26.21.1**. The connection from a specific data source to a specific data destination is defined by configuring the corresponding address of a data source in the desired data destination. The configured address of each data destination is pointed out by the numbers in the yellow boxes of **Figure 26-5**.

Generic Timer Module (GTM)

Normally, the destination is idle and waits for data from the source. If the source offers new data, the destination does a destructive read, processes the data and goes idle again. The same data is never read twice.

There is one submodule for which this destructive read access does not hold. This is the BRC submodule configured in Maximal Throughput Mode. For a detailed description of this module please refer to [Section 26.4](#).

The functionality of the ARU is as follows: The ARU sequentially polls the data destinations of the connected modules in a round-robin order. If a data destination requests new data from its configured data source and the data source has data available, the ARU delivers the data to the destination and it informs both, the data source and destination that the data is transferred. The data source marks the delivered ARU data as invalid which means that the destination consumed the data.

It should be noted that each data source should only be connected to a single data destination. This is because the destinations consume the data. If two destinations would reference the same source one destination would consume the data before the other destination could consume it. Since the data transfers are blocking, the second destination would block until it receives new data from the source. If a data source should be connected to more than one data destination the submodule Broadcast (BRC) has to be used. On the other hand, the transfer from a data source to the ARU is also blocking, which means that the source channel can only provide new data to the ARU when an old data word is consumed by a destination. In order to speed up the process of data transfers, the ARU handles two different data destinations in parallel. Therefore, a transfer between source and destination takes two cycles, but since the transfers are pipelined these two cycles have only effect for one round trip of the ARU.

[Table 26-4](#) gives an overview about the number of channels for the GTM.

Table 26-4 ARU Channels

Submodule	Number of data sources	Number of data destinations
ARU	1	0
DPLL	24	24
TIM 0-2	24	0
MCS 0-2	72	24
BRC	22	12
TOM	0	0
ATOM 0-2	24	24
PSM 0	8	8
ICM	0	0
CMP	0	0

Table 26-4 ARU Channels (cont'd)

Submodule	Number of data sources	Number of data destinations
MON	0	0
Total	165	92

26.2.3.2 ARU Round Trip Time

The ARU uses a round-robin arbitration scheme with a fixed round trip time for all connected data destinations. This means that the time between two adjacent read requests resulting from a data destination channel always takes the round trip time, independently if the read request succeeds or fails.

The worst case round-trip time is defined as $65 * \text{SYS_CLC}$ of the GTM input system clock. Since the round-trip time depends on the number of destinations the ARU has to ensure that the round-trip time never exceeds the defined time.

26.2.3.3 ARU Blocking Mechanism

Another important concept of the ARU is its blocking mechanism that is implemented for transferring data from a data source to a data destination. This mechanism is used by ARU connected submodules to synchronize the submodules to the routed data streams. [Figure 26-6](#) explains the blocking mechanism.

Graphical representation of ARU blocking mechanism

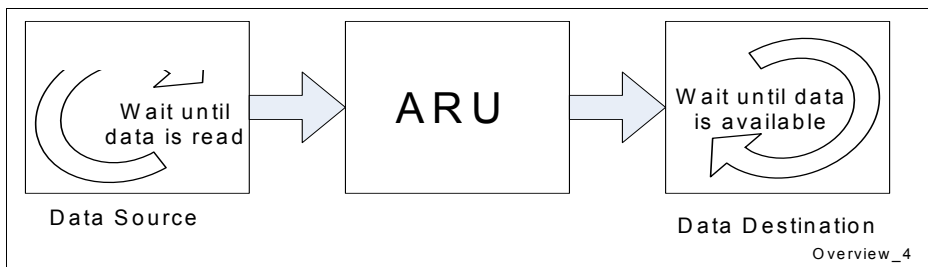


Figure 26-6 Graphical representation of ARU blocking mechanism

If a data destination requests data from a data source over the ARU but the data source does not have any data yet, it has to wait until the data source provides new data. In this case the submodule that owns the data destination may perform other tasks. When a data source produces new data faster than a data destination can consume the data the source raises an error interrupt and signals that the data could not be delivered in time. The new data is marked as valid for further transfers and the old data is overwritten.

Generic Timer Module (GTM)

In any case the round trip time for the ARU is always fixed.

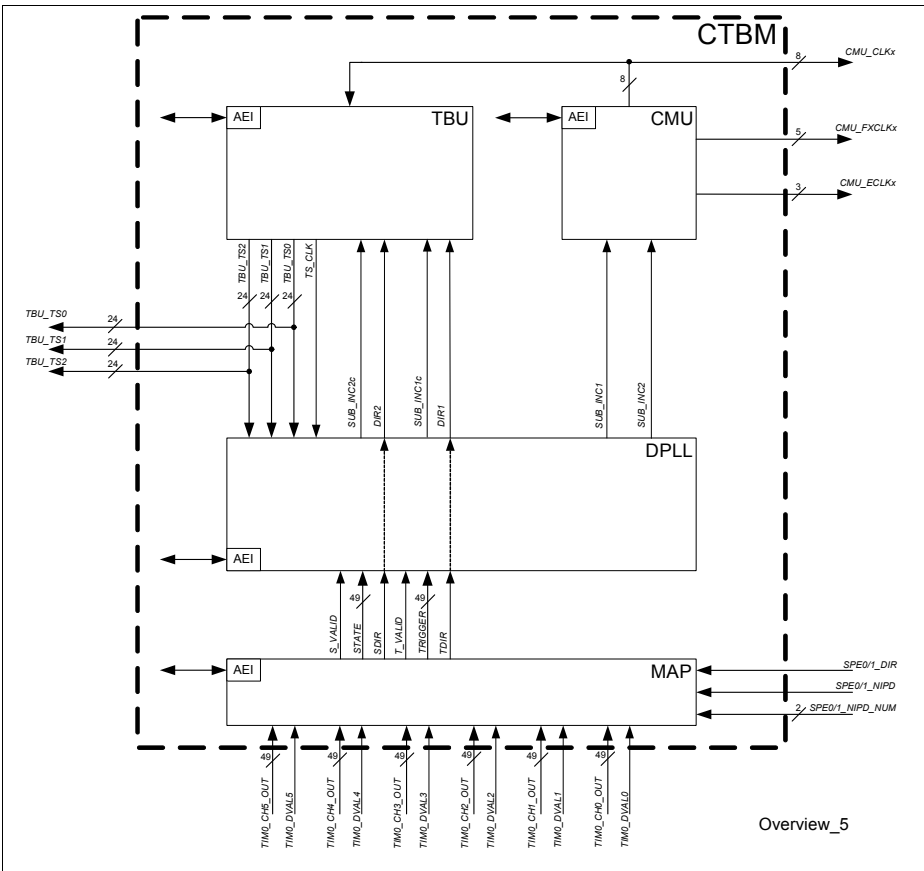
One exception is the BRC submodule when configured in Maximal Throughput Mode. Please refer to [Section 26.4](#) for a detailed description.

26.2.4 GTM Clock and Time Base Management (CTBM)

Inside the GTM several subunits are involved in the clock and time base management of the whole GTM. [Figure 26-7](#) shows the connections and subblocks involved in these tasks. The subblocks involved are called Clock and Time Base Management (CTBM) modules further on.

26.2.4.1 GTM Clock and time base management architecture

Generic Timer Module (GTM)



Overview_5

Figure 26-7 GTM Clock and time base management architecture

One important module of the CTBM is the Clock Management Unit (CMU) which generates up to 13 clocks for the submodules of the GTM and up to three GTM external clocks CMU_ECLK[z] (z: 0...2). For a detailed description of the CMU functionality and clocks please refer to [Section 26.8](#).

The five (5) CMU_FXCLK[y] (y: 0...4) clocks are used by the TOM submodule for PWM generation. A maximum of eight (8) CMU_CLK[x] (x: 0...7) clocks are used by other submodules of the GTM for signal generation.

Inside the Time Base Unit (TBU) one of these maximum eight clocks is used per channel to generate a common time base for the GTM. Besides the CMU_CLK[x] signals, the TBU can use the compensated SUB_INC[i]c (i: 1,2) signals coming from the DPLL

Generic Timer Module (GTM)

submodule for time base generation. This time base then typically represents an angle clock for an engine management system. For the meaning of compensated (SUB_INC[i]c) and uncompensated (SUB_INC[i]) DPLL signals please refer to the DPLL [Section 26.16](#). The SUB_INC[i]c signals in combination with the two direction signal lines DIR[i] the TBU time base can be controlled to run forwards or backwards. The TBU functionality is described in [Section 26.9](#).

The TBU submodule generates the three time base signals TBU_TS0, TBU_TS1 and TBU_TS2 which are widely used inside the GTM as common time bases for signal characterization and generation.

As stated before, the DPLL submodule provides the four clock signals SUB_INC[i] and SUB_INC[i]c which can be seen as a clock multiplier generated out of the two input signal vectors TRIGGER and STATE coming from the MAP submodule. For a detailed description of the DPLL functionality please refer to [Section 26.16](#).

The MAP submodule is used to select the TRIGGER and STATE signals for the DPLL out of six input signals coming from TIM0 submodule. Besides this, the MAP submodule is able to generate a TDIR (TRIGGER Direction) and SDIR (STATE Direction) signal for the DPLL and TBU coming from the SPE0 and SPE1 signal lines. The direction signals are generated out of a defined input pattern. For a detailed description of the MAP submodule please refer to [Section 26.15](#).

26.2.5 GTM Interrupt Concept

The submodules of the GTM can generate thousands of interrupts on behalf of internal events. This high amount of interrupts is combined inside the Interrupt Concentrator Module (ICM) into interrupt groups. In this interrupt groups the GTM submodule interrupt signals are bundled to a smaller set of interrupts. Out of these interrupt sets a smaller amount of interrupt signals is created and signalled outside of the GTM as a signal GTM_<MOD>_IRQ, where as <MOD> identifies the name of the corresponding GTM submodule.

The controlling of the individual interrupts is done inside the submodules. If a submodule consists of several submodule channels that are most likely to work independent from each other (like TIM, PSM, MCS, TOM, and ATOM), each submodule channel has its own interrupt control and status register set, named as interrupt set in the following. Other submodules (SPE, ARU, DPLL, BRC, CMP and global GTM functionality) have a common interrupt set for the whole submodule.

The interrupt set consists of four registers: The IRQ_EN register, the IRQ_NOTIFY register, the IRQ_FORCINT register, and the IRQ_MODE register. While the registers IRQ_EN, IRQ_NOTIFY, and IRQ_FORCINT signalize the status and allow controlling of each individual interrupt source within an interrupt set, the register IRQ_MODE configures the interrupt mode that is applied to all interrupts that belong to the same interrupt set.

Generic Timer Module (GTM)

In order to support a wide variety of microcontroller architectures and interrupt systems with different interrupt signal output characteristics and internal interrupt handling the following four modes can be configured:

Level mode

Pulse mode

Pulse-Notify mode

Single-Pulse mode

It is recommended to use the Pulse-Notify mode as interrupt mode.

These interrupt modes are described in more details in the following subsections.

The register `IRQ_EN` allows the enabling and disabling of an individual interrupt within an interrupt set. Independent of the configured mode, only enabled interrupts can signalize an interrupts on its signal `GTM_<MOD>_IRQ`.

The register `IRQ_NOTIFY` collects the occurrence of interrupt events. The behaviour for setting a bit in this register depends on the configured mode and thus it is described later on in the mode descriptions.

Independent of the configured mode any write access with value '1' to a bit in the register `IRQ_NOTIFY` always clears the corresponding `IRQ_NOTIFY` bit.

Moreover, the enabling of a disabled interrupt sources with a write access to the register `IRQ_EN` also clears the corresponding bit in the `IRQ_NOTIFY` register but only if the error interrupt source `EIRQ_EN` is disabled. However, if the enabling of a disabled interrupt is simultaneous to an incoming interrupt event, the interrupt event is dominant and the register `IRQ_NOTIFY` is not cleared.

Additionally, each write access to the register `IRQ_MODE`, clears all bits in the `IRQ_NOTIFY` register. It should be notified that the clearing of `IRQ_NOTIFY` is applied independently of the written data (e.g. no mode change).

Thus, a secure way for reconfiguring the interrupt mode of an interrupt set, is to disable all interrupts of the interrupt set with the register `IRQ_EN`, define the new interrupt mode by writing register `IRQ_MODE`, followed by enabling the desired interrupts with the register `IRQ_EN`.

Thus, a secure way for reconfiguring the interrupt mode of an error interrupt set, is to disable all error interrupts of the error interrupt set with the register `EIRQ_EN`, define the new interrupt mode by writing register `IRQ_MODE`, followed by enabling the desired error interrupts with the register `EIRQ_EN`.

The register `IRQ_FORCINT` is used by software for triggering individual interrupts with a write access with value '1'. Since a write access to `IRQ_FORCINT` only generates a single pulse, `IRQ_FORCINT` is not implemented as a true register and thus any read access to `IRQ_FORCINT` always results with a value of '0'.

Generic Timer Module (GTM)

It should be noted, that the mechanism for triggering interrupts with `IRQ_FORCINT` is globally disabled after reset. It has to be explicitly enabled by clearing the bit `RF_PROT` in the register `GTM_CTRL` (see [Section 26.2.9.3](#))

For the modules AEI-bridge, BRC, FIFO, TIM, MCS, DPLL, SPE and CMP each interrupt may be configured to raise instead of the normal interrupt an error interrupt if enabled by the corresponding error interrupt enable bit in register `EIRQ_EN`.

Note, it is possible for one source to enable the normal interrupt and the error interrupt in parallel. Because of both interrupt clear signals could reset the notify bit this is expected to cause problems in a system and therefore it is strongly recommended to not enable both interrupt types at the same point in time.

Similar to enabling an interrupt, the enabling of a disabled error interrupt source with a write access to the register `EIRQ_EN` also clears the corresponding bit in the `IRQ_NOTIFY` register only if the interrupt source is disabled. However, if the enabling of a disabled error interrupt is simultaneous to an incoming interrupt event, the interrupt event is dominant and the register `IRQ_NOTIFY` is not cleared.

All enabled error interrupts are or-combined inside the ICM and assigned to the dedicated GTM port `gtm_err_irq`.

To be able to detect the module source of the error interrupt the ICM provides the register `ICM_IRQG_MEI`.

The error interrupt causing channel can be determined for the modules FIFO, TIM and MCS by evaluating the ICM register `ICM_IRQG_CEI0` to `ICM_IRQG_CEI4`.

26.2.5.1 Level interrupt mode

The default interrupt mode is the Level Interrupt Mode. In this mode each occurred interrupt event is collected in the register `IRQ_NOTIFY`, independent of the corresponding enable bit of register `IRQ_EN` and `EIRQ_EN`.

An interrupt event, which is defined as a pulse on the signal `Int_out` of [Figure 26-8](#), may be triggered by the interrupt source of the submodule or by software performing a write access to the corresponding register `IRQ_FORCINT`, with a disabled bit `RF_PROT` in register `GTM_CTRL`.

Level interrupt mode scheme

Generic Timer Module (GTM)

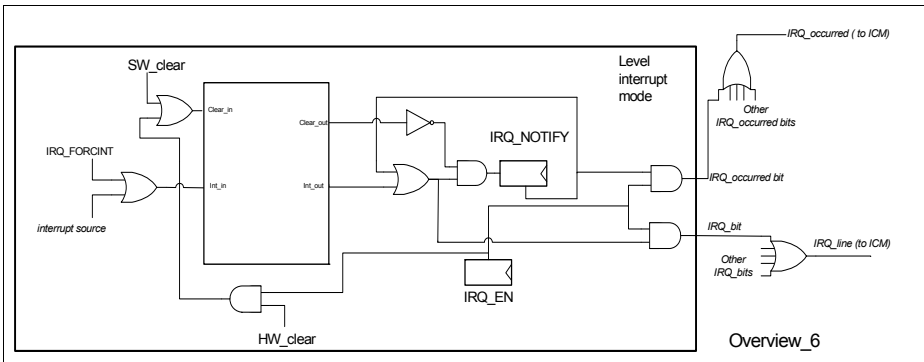


Figure 26-8 Level interrupt mode scheme

A collected interrupt bit in register `IRQ_NOTIFY` may be cleared by a clear event, which is defined as a pulse on signal `Clear_out` of **Figure 26-8**. A clear event can be performed with a write access with value '1' to the corresponding bit in the register `IRQ_NOTIFY` leading to a pulse on signals `SW_clear`. A clear event may also result from an externally connected signal `GTM_<MOD>_IRQ_CLR`, which is routed to the signal `HW_clear` of **Figure 26-8**. However, the hardware clear mechanism is only possible, if the corresponding interrupt is enabled by register `IRQ_EN`.

As the **Table 26-5** shows, interrupt events are dominant in the case of a simultaneous interrupt event and clear event.

Table 26-5 IRQ_NOTIFY behaviour

Int_in	Clear_in	Int_out	Clear_out
0	0	0	0
0	1	0	1
1	0	1	0
1	1	1	0

Priority of Interrupt Events and Clear Events

As it can be seen from the **Figure 26-8** an occurred interrupt event is signaled as a constant signal level with value 1 to the signal `IRQ_bit`, if the corresponding interrupt is enabled in register `IRQ_EN`.

With exception of the submodules `ARU` and `DPLL`, the signal `IRQ_bit` is OR-combined with the neighboring `IRQ_bit` signals of the same interrupt set and they are routed as a single signal `IRQ_line` to the interrupt concentrator module (ICM). The interrupt signals `IRQ_bit` of the submodules `DPLL` and `ARU` are routed directly as a signal `IRQ_line` to the submodule `ICM`. In some cases (submodules `TOM` and `ATOM`) the `ICM` may further OR-

Generic Timer Module (GTM)

combine several IRQ_line signals to an outgoing interrupt signal GTM_<MOD>_IRQ. In the other cases the IRQ_line signals are directly connected to the outgoing signals GTM_<MOD>_IRQ. within the submodule ICM.

The signal IRQ_occurred is connected in a similar way as the signal IRQ_line, however this signal is used for monitoring the interrupt state of the register IRQ_NOTIFY in the registers of the ICM.

The additional error interrupt enable mechanism for level interrupt is shown below.

Level interrupt scheme for modules AEI-bridge, BRC, FIFO, TIM, MCS, DPLL, SPE, CMP

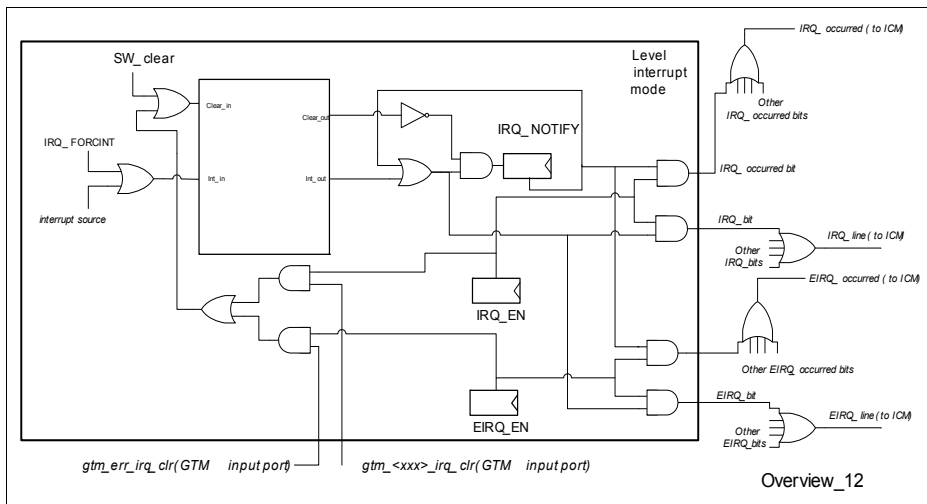


Figure 26-9 Level interrupt scheme

A collected interrupt bit in register IRQ_NOTIFY may be cleared by a clear event, which is defined as a pulse on signal Clear_out of [Figure 26-8](#). A clear event can be performed with a write access with value '1' to the corresponding bit in the register IRQ_NOTIFY leading to a pulse on signals SW_clear. A clear event may also result from externally connected signal gtm_<xxx>_irq_clr or gtm_err_irq_clr, which is routed as a HW_clear to Clear_in of [Figure 26-8](#). However, the hardware clear mechanism is only possible, if the corresponding interrupt or error interrupt is enabled by register IRQ_EN or EIRQ_EN.

As it can be seen from the [Figure 26-9](#) an occurred interrupt event is signalled as a constant signal level with value 1 to the signal IRQ_bit, if the corresponding interrupt is enabled in register IRQ_EN.

26.2.5.2 Pulse interrupt mode

The Pulse interrupt mode behaviour can be observed from [Figure 26-10](#).

Pulse interrupt mode scheme

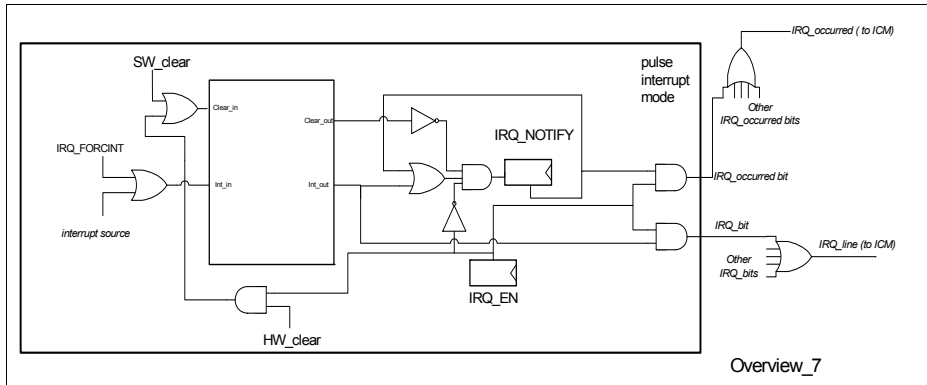


Figure 26-10 Pulse interrupt mode scheme

In Pulse Interrupt Mode each Interrupt Event will generate a pulse on the IRQ_bit signal if IRQ_EN is enabled.

As it can be seen from the figure, the interrupt bit in IRQ_NOTIFY register is always cleared if IRQ_EN or IRQ_EN are enabled.

However, if an interrupt is disabled in the register IRQ_EN, an occurred interrupt event is captured in the register IRQ_NOTIFY, in order to allow polling for disabled interrupts by software.

Disabled interrupts may be cleared by an interrupt clear event.

In Pulse interrupt mode, the signal IRQ_occurred is always 0.

The additional error interrupt enable mechanism for pulse interrupt is shown below.

Pulse interrupt scheme for modules AEI-bridge, BRC, FIFO, TIM, MCS, DPLL, SPE, CMP

Generic Timer Module (GTM)

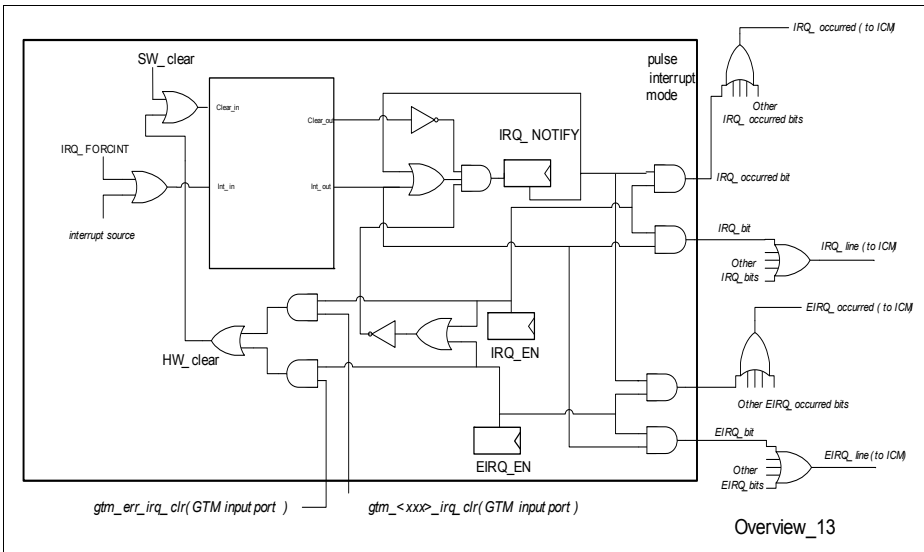


Figure 26-11 Pulse interrupt scheme

In Pulse Interrupt Mode each Interrupt Event will generate a pulse on the EIRQ_bit signal if EIRQ_EN is enabled.

As it can be seen from the figure, the interrupt bit in IRQ_NOTIFY register is always cleared if EIRQ_EN is enabled.

However, if an error interrupt is disabled in the register EIRQ_EN, an occurred error interrupt event is captured in the register IRQ_NOTIFY, in order to allow polling for disabled error interrupts by software.

Disabled error interrupts may be cleared by an error interrupt clear event.

In Pulse interrupt mode, the signal EIRQ_occurred is always 0.

26.2.5.3 Pulse-notify interrupt mode

In Pulse-notify Interrupt mode, all interrupt events are captured in the register IRQ_NOTIFY. If an interrupt is enabled by the register IRQ_EN, each interrupt event will also generate a pulse on the IRQ_bit signal. The signal IRQ_occurred will be high if interrupt is enabled in register IRQ_EN and the corresponding bit of register IRQ_NOTIFY is set. The Pulse-notify interrupt mode is shown in [Figure 26-12](#).

Pulse-notify interrupt mode scheme

Generic Timer Module (GTM)

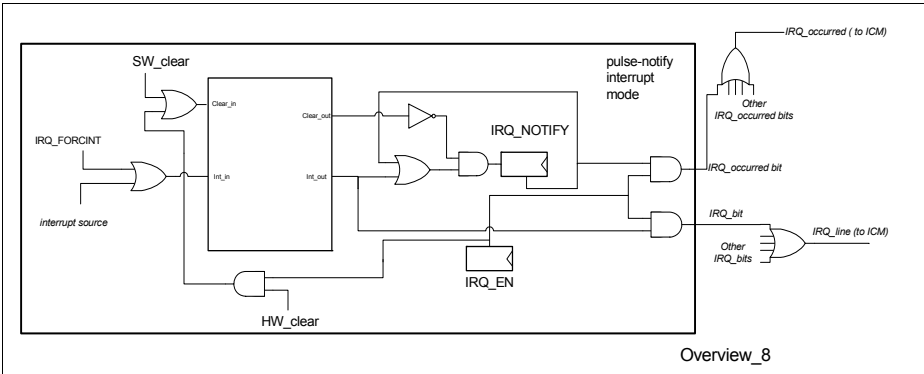


Figure 26-12 Pulse-notify interrupt mode scheme

The additional error interrupt enable mechanism for pulse-notify interrupt is shown below

Pulse-notify interrupt scheme for modules AEI-bridge, BRC, FIFO, TIM, MCS, DPLL, SPE, CMP

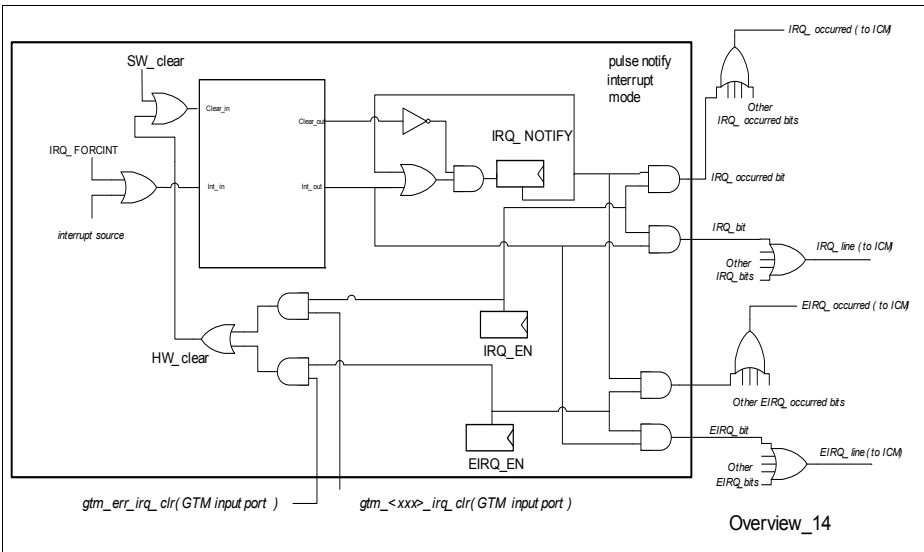


Figure 26-13 Pulse-notify interrupt scheme

Generic Timer Module (GTM)

In Pulse-notify Interrupt mode, all error interrupt events are captured in the register IRQ_NOTIFY. If an error interrupt is enabled by the register EIRQ_EN, each error interrupt event will also generate a pulse on the EIRQ_bit signal. The signal EIRQ_occurred will be high if error interrupt is enabled in register EIRQ_EN and the corresponding bit of register IRQ_NOTIFY is set. The Pulse-notify interrupt mode for error interrupts is shown in figure 2.5.3.2.

26.2.5.4 Single-pulse interrupt mode

In Single-pulse Interrupt Mode, an interrupt event is always captured in the register IRQ_NOTIFY, independent of the state of IRQ_EN. However, only the first interrupt event of an enabled interrupt within a common interrupt set is forwarded to signal IRQ_line. Additional interrupt events of the same interrupt set cannot generate pulses on the signal IRQ_line, until the corresponding bits in register IRQ_NOTIFY of enabled interrupts are cleared by a clear event. The IRQ_occurred signal line will be high, if the IRQ_EN and the IRQ_NOTIFY register bits are set. The Single-pulse interrupt mode is shown in **Figure 26-14**.

The only exceptions are the modules ARU and DPLL. In these modules the IRQ_occurred bit of each interrupt is directly connected (without OR-conjunction of neighboring IRQ_occurred bits) to the inverter for suppressing further interrupt pulses.

Single-pulse interrupt mode scheme

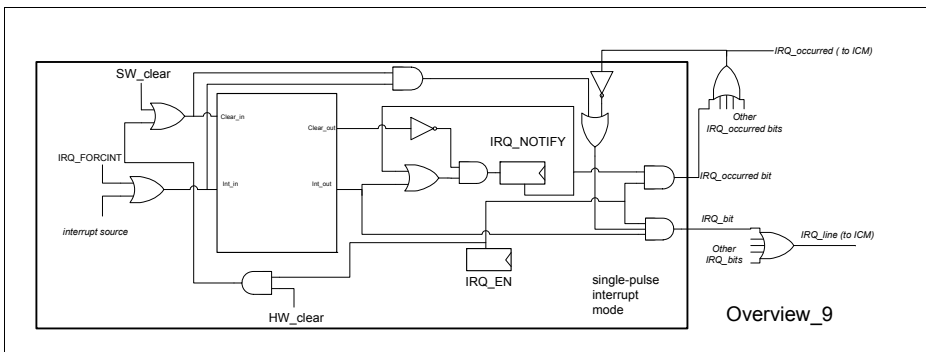


Figure 26-14 Single-pulse interrupt mode scheme

To avoid unexpected IRQ behaviour in the single pulse mode, all desired interrupt sources should be enabled by a single write access to IRQ_EN and the notification bits should be cleared by a single write access to the register IRQ_NOTIFY.

The additional error interrupt enable mechanism for single-pulse interrupt is shown below

Generic Timer Module (GTM)

Single-pulse interrupt scheme for modules AEI-bridge, BRC, FIFO, TIM, MCS, DPLL, SPE, CMP

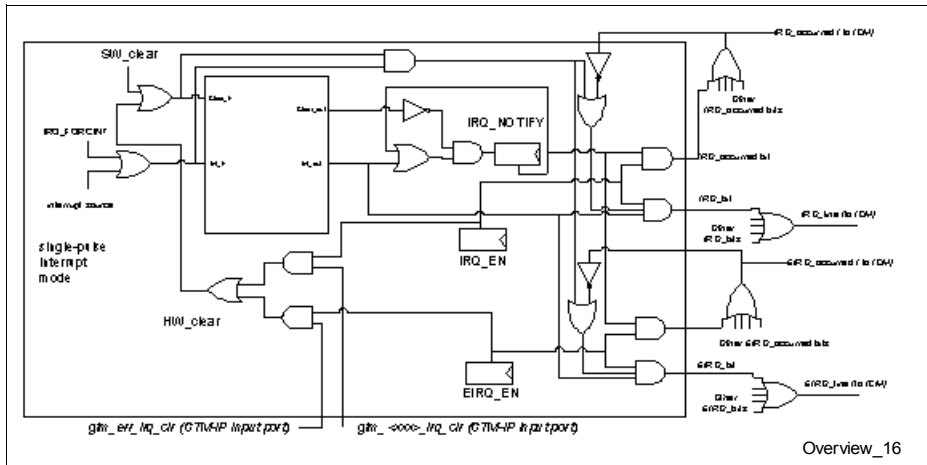


Figure 26-15 Single-pulse interrupt mode scheme

In Single-pulse Interrupt Mode, an error interrupt event is always captured in the register IRQ_NOTIFY, independent of the state of EIRQ_EN. However, only the first error interrupt event of an enabled error interrupt within a common error interrupt set is forwarded to signal EIRQ_line. Additional error interrupt events of the same error interrupt set cannot generate pulses on the signal EIRQ_line, until the corresponding bits in register IRQ_NOTIFY of enabled error interrupts are cleared by a clear event. The EIRQ_occurred signal line will be high, if the EIRQ_EN and the IRQ_NOTIFY register bits are set. The Single-pulse interrupt mode for error interrupts is shown in figure 2.5.4.2.

To avoid unexpected EIRQ behaviour in the single pulse mode, all desired error interrupt sources should be enabled by a single write access to EIRQ_EN and the notification bits should be cleared by a single write access to the register IRQ_NOTIFY.

The only exceptions are the modules ARU and DPLL. In these modules the EIRQ_occurred bit of each error interrupt is directly connected (without OR-conjunction of neighboring EIRQ_occurred bits) to the inverter for suppressing further error interrupt pulses.

Generic Timer Module (GTM)

26.2.5.5 GTM Interrupt concentration method

Because of the grouping of interrupts inside the ICM, it can be necessary for the software to access the ICM submodule first to determine the interrupt set that is responsible for an interrupt. A second access to the responsible register IRQ_NOTIFY is then necessary to identify the interrupt source, serve it and to reset the interrupt flag in register IRQ_NOTIFY afterwards. The interrupt flags are never reset by an access to the ICM. For a detailed description of the ICM submodule please refer to [Chapter 26.18](#).

26.2.6 GTM Software Debugger Support

For software debugger support the GTM comes with several features. E.g. status register bits must not be altered by a read access from a software debugger. To avoid this behaviour to reset a status register bit by software, the CPU has to write a '1' explicitly to the register bit to reset its content.

The [Table 26-6](#) describes the behaviour of some GTM registers with special functionality on behalf of read accesses from the AEI bus interface.

26.2.6.1 Register behaviour in case of Software Debugger accesses

Table 26-6 Register behaviour in case of Software Debugger accesses

Module	Register	Description
AFD	AFDi_CHx_BUFFACC	The FIFO read access pointers are not altered on behalf of a debugger read access to this register.
TIM	TIMi_CHx_GPR0 / 1	The overflow bit is not altered in case of a debugger read access to this register.
ATOM	ATOMi_CHx_SR0 / 1	In SOMC mode a read access to this register by the debugger does not release the channel for a new compare / match event.

Further on, some important states inside the GTM submodule have to be signalled to the outside world, when reached and should for example trigger the software debugger to stop program execution. For this internal state signalling please refer to the GTM module integration guide.

The GTM provides an external signal gtm_halt, which disables clock signal SYS_CLK for debugging purposes. If SYS_CLK is disabled, a connected debugger can read any GTM related register and the GTM internal RAMs using AEI. Moreover, the debugger can also perform write accesses to the internal RAMs and to several GTM related registers in order to enable advanced debugging features (e.g. modifications of register contents in single step mode).

26.2.7 GTM Programming conventions

To serve different application domains the GTM is a highly configurable module with many configuration modes. In principle the submodules of the GTM are intended to be configured at system startup to fulfil certain functionality for the application domain the microcontroller runs in.

For example, a TIM input channel can be used to monitor an application specific external signal, and this signal has to be filtered. Therefore, the configuration of the TIM channel filter mode will be specific to the external signal characteristic. While it can be necessary to adapt the filter thresholds during runtime an adaptation of the filter mode during runtime is not reasonable. Thus, the change of the filter mode during runtime can lead to an unexpected behaviour.

In general, the programmer has to be careful when reprogramming configuration registers of the GTM submodules during runtime. It is recommended to disable the channels before reconfiguration takes place to avoid unexpected behaviour of the GTM.

26.2.8 GTM TOP-Level Configuration Registers Overview

GTM TOP-level contains following configuration registers:

Table 26-7 GTM TOP-Level Configuration Registers Overview

Register name	Description	Details in Section
GTM_REV	GTM Version control register	Section 26.2.9.1
GTM_RST	GTM Global reset register	Section 26.2.9.2
GTM_CTRL	GTM Global control register	Section 26.2.9.3
GTM_AEI_ADDR_XPT	GTM AEI Timeout exception address register	Section 26.2.9.4
GTM_IRQ_NOTIFY	GTM Interrupt notification register	Section 26.2.9.5
GTM_IRQ_EN	GTM Interrupt enable register	Section 26.2.9.6
GTM_IRQ_EIRQ_EN	GTM Error interrupt generation register	Section 26.2.9.12
GTM_IRQ_FORCINT	GTM Software interrupt generation register	Section 26.2.9.7
GTM_IRQ_MODE	GTM top level interrupts mode selection. Please note that this mode selection is only valid for the three interrupts described in Section 26.2.9.5	Section 26.2.9.8
GTM_TIMi_AUX_IN_S RC (i= 0...n)	GTM-IP TIMi module AUX_IN source selection register	Section 26.2.9.13

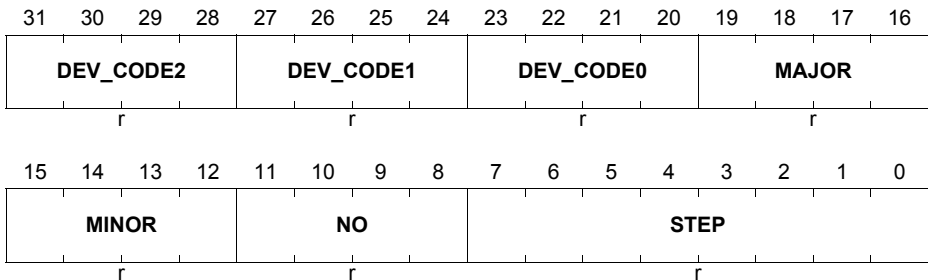
26.2.9 GTM TOP-Level Configuration Registers Description

All of the following registers are 32-bit only accessible.

26.2.9.1 Register GTM_REV

GTM_REV

GTM Version Control Register (00000_H) **Reset Value: 122155A1_H**



Field	Bits	Type	Description
STEP	[7:0]	r	Release Step
NO	[11:8]	r	Define delivery number of GTM specification
MINOR	[15:12]	r	Define minor version number of GTM specification
MAJOR	[19:16]	r	Define major version number of GTM specification
DEV_COD E0	[23:20]	r	Device encoding digit 0
DEV_COD E1	[27:24]	r	Device encoding digit 1
DEV_COD E2	[31:28]	r	Device encoding digit 2 Note: The numbers are encoded in BCD. Values "A" - "F" are chapters.

Generic Timer Module (GTM)

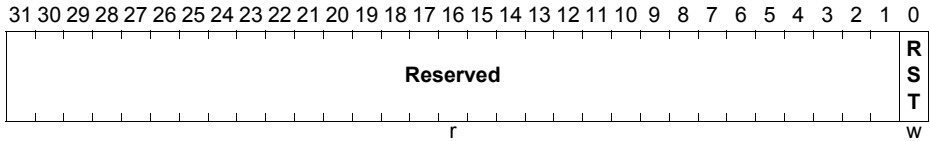
26.2.9.2 Register GTM_RST

GTM_RST

GTM Global Reset Register

(00004_H)

Reset Value: 00000000_H



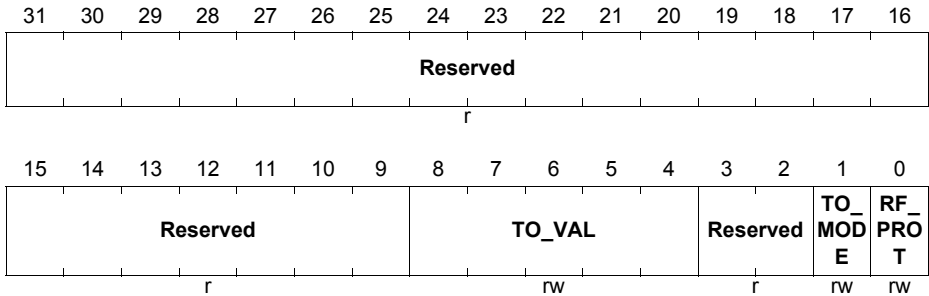
Field	Bits	Type	Description
RST	0	w	<p>GTM Reset</p> <p>0_B No reset action</p> <p>1_B Initiate reset action for all submodules</p> <p>Note: This bit is automatically cleared by hardware after it was written. Therefore, the register is always read as zero (0) by the software.</p> <p>Note: This bit is write protected by bit GTM_CTRL.RF_PROT.</p>
Reserved	[31:1]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>

Generic Timer Module (GTM)

26.2.9.3 Register GTM_CTRL

GTM_CTRL

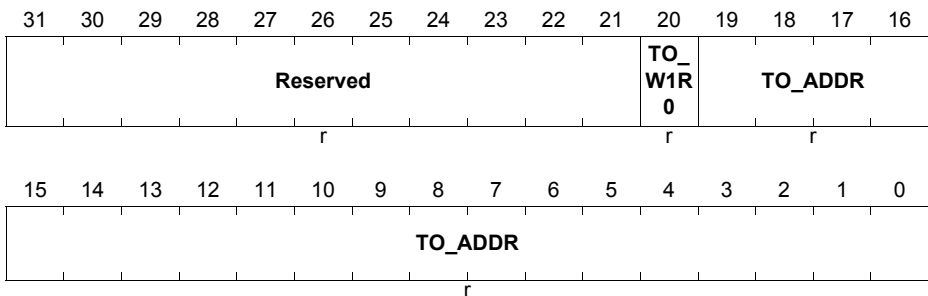
GTM Global Control Register (00008_H) Reset Value: 00000001_H



Field	Bits	Type	Description
RF_PROT	0	rw	<p>RST and FORCINT protection</p> <p>0_B SW RST (global), SW interrupt FORCINT, and SW RAM reset functionality is enabled</p> <p>1_B SW RST (global), SW interrupt FORCINT, and SW RAM reset functionality is disabled</p>
TO_MODE	1	rw	<p>AEI Timeout mode</p> <p>0_B Observe: If timeout_counter=0 the address and rw signal in addition with timeout flag will be stored to the GTM_AEI_ADDR_XPT register. Following timeout_counter=0 accesses will not overwrite the first entry in the aei_addr_timeout register. Clearing the timeout flag/aei_status error_code will reenale the storing of a next faulty access.</p> <p>1_B Abort: In addition to observe mode the pending access will be aborted by signalling an illegal module access on aei_status and sending ready. In case of a read deliver as data 0 by serving of next AEI accesses.</p>
Reserved	[3:2]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
TO_VAL	[8:4]	rw	AEI Timeout value Note: These bits define the number of cycles after which a timeout event occurs. When TO_VAL equals zero (0) the AEI timeout functionality is disabled.
Reserved	[31:9]	r	Reserved Read as zero, should be written as zero

26.2.9.4 Register GTM_AEI_ADDR_XPT
GTM_AEI_ADDR_XPT
**GTM AEI Timeout Exception Address Register
(0000C_H)**
Reset Value: 00000000_H


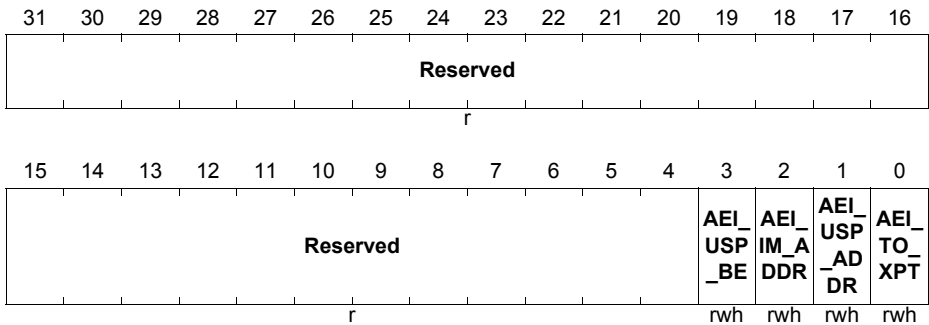
Field	Bits	Type	Description
TO_ADDR	[19:0]	r	AEI Timeout address Note: This bit field defines the AEI address for which the AEI timeout event occurred.
TO_W1R0	20	r	AEI Timeout Read/Write flag Note: This bit defines the AEI Read/Write flag for which the AEI timeout event occurred.
Reserved	[31:21]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.2.9.5 Register GTM_IRQ_NOTIFY

GTM_IRQ_NOTIFY

 GTM Interrupt Notification Register (00010_H)

 Reset Value: 00000000_H


Field	Bits	Type	Description
AEI_TO_XPT	0	rwh	<p>AEI Timeout exception occurred</p> <p>0_B No interrupt occurred</p> <p>1_B AEI_TO_XPT interrupt was raised by the AEI Timeout detection unit</p> <p>Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.</p>
AEI_USP_ADDR	1	rwh	<p>AEI Unsupported address interrupt</p> <p>0_B No interrupt occurred</p> <p>1_B AEI_USP_ADDR interrupt was raised by the AEI interface</p> <p>Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.</p>
AEI_IM_ADDR	2	rwh	<p>AEI Illegal Module address interrupt</p> <p>0_B No interrupt occurred</p> <p>1_B AEI_IM_ADDR interrupt was raised by the AEI interface</p> <p>Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
AEI_USP_BE	3	rwh	AEI Unsupported byte enable interrupt 0 _B No interrupt occurred 1 _B AEI_USP_BE interrupt was raised by the AEI interface Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
Reserved	[31:4]	r	Reserved Read as zero, should be written as zero

26.2.9.6 Register GTM_IRQ_EN

GTM_IRQ_EN

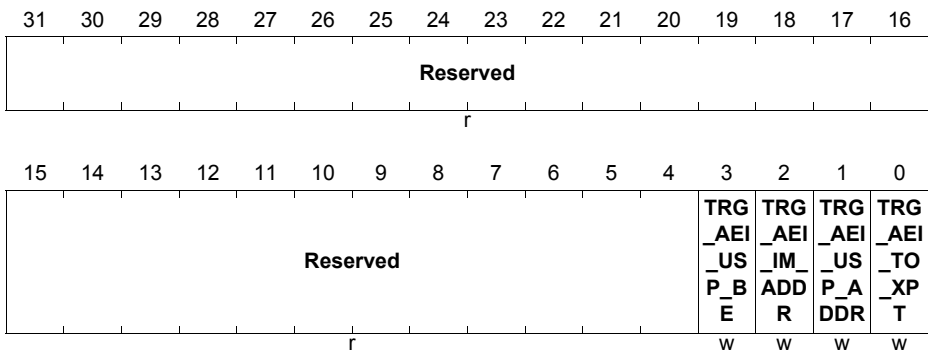
 GTM Interrupt Enable Register (00014_H) Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												AEI_USP_IRQ_EN	AEI_IM_DDR_IRQ_EN	AEI_USP_ADDR_DRQ_EN	AEI_TO_XPT_IRQ_EN
r												rw	rw	rw	rw

Field	Bits	Type	Description
AEI_TO_XPT_IRQ_EN	0	rw	AEI_TO_XPT_IRQ interrupt enable. 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
AEI_USP_ADDR_IRQ_EN	1	rw	AEI_USP_ADDR_IRQ interrupt enable. 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM

Generic Timer Module (GTM)

Field	Bits	Type	Description
AEI_IM_A DDR_IRQ_ EN	2	rw	AEI_IM_ADDR_IRQ interrupt enable. 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
AEI_USP_ BE_IRQ_E N	3	rw	AEI_USP_BE_IRQ interrupt enable. 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
Reserved	[31:4]	r	Reserved Read as zero, should be written as zero

26.2.9.7 Register GTM_IRQ_FORCINT
GTM_IRQ_FORCINT
**GTM Software Interrupt Generation Register
(00018_H)**
Reset Value: 00000000_H


Field	Bits	Type	Description
TRG_AEI_ TO_XPT	0	w	Trigger AEI_TO_XPT_IRQ interrupt by software. 0 _B No interrupt triggering 1 _B Assert AEI_TO_XPT_IRQ interrupt for one clock cycle Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG_AEI_USP_ADDR	1	w	Trigger AEI_USP_ADDR_IRQ interrupt by software. 0 _B No interrupt triggering 1 _B Assert AEI_USP_ADDR_IRQ interrupt for one clock cycle Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
TRG_AEI_IM_ADDR	2	w	Trigger AEI_IM_ADDR_IRQ interrupt by software. 0 _B No interrupt triggering 1 _B Assert AEI_IM_ADDR_IRQ interrupt for one clock cycle Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
TRG_AEI_USP_BE	3	w	Trigger AEI_USP_BE_IRQ interrupt by software. 0 _B No interrupt triggering 1 _B Assert AEI_USP_BE_IRQ interrupt for one clock cycle Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
Reserved	[31:4]	r	Reserved Read as zero, should be written as zero

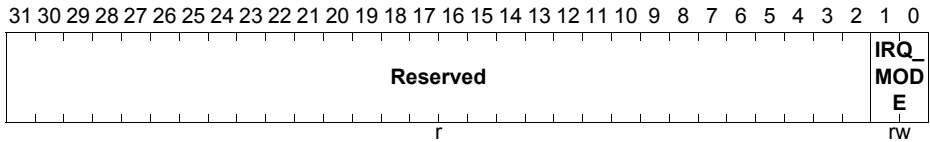
26.2.9.8 Register GTM_IRQ_MODE

GTM_IRQ_MODE

GTM Top Level Interrupts Mode Selection

(0001C_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
IRQ_MOD E	[1:0]	rw	Interrupt strategy mode selection for the AEI timeout and address monitoring interrupts 00 _B Level mode 01 _B Pulse mode 10 _B Pulse-Notify mode 11 _B Single-Pulse mode Note: The interrupt modes are described in Section 26.2.5 .
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

26.2.9.9 Register GTM_BRIDGE_MODE

Note: The content of this register should never be changed.

GTM_BRIDGE_MODE

GTM to SPB BRIDGE MODE

(00030_H)

Reset Value: 04001001_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUFF_DPT											Reserved				BRG _RS T
r											r				w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			SYN C_IN PUT _RE G	Reserved		BUF F_O VL	MOD E_U P_P GR	Reserved					MSK _WR _RS P	BRG _MO DE	
r			rw	r		rwh	r	r					rw	rw	

Field	Bits	Type	Description
BRG_MODE	0	rw	Defines the operation mode for the AEI bridge 0 _B AEI bridge operates in sync_bridge mode 1 _B AEI bridge operates in async_bridge mode
MSK_WR_RSP	1	rw	Mask write response 0 _B Do not mask the write response 1 _B Mask write response
Reserved	[7:2]	r	Reserved Read as zero, should be written as zero
MODE_UP_PGR	8	r	Mode update in progress 0 _B No update in progress. 1 _B Update in progress.
BUFF_OVERFLOW	9	rwh	Buffer overflow register 0 _B No buffer overflow occurred. 1 _B Buffer overflow occurred. Note: A buffer overflow can occur while multiple aborts are issued by the external bus or a pipelined instruction is started while FBC = 0 (see GTM_BRIDGE_PTR1 register). This bit always read as zero.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[11:10]	r	Reserved Read as zero, should be written as zero.
SYNC_INP UT_REG	12	rw	Additional Pipeline Stage in Synchronous Bridge Mode 0 _B No additional pipeline stage implemented 1 _B Additional pipeline stage implemented. All accesses in synchronous mode will be increased by one clock cycle.
Reserved	[15:13]	r	Reserved Read as zero, should be written as zero.
BRG_RST	16	w	Bridge software reset 0 _B No bridge reset request. 1 _B Bridge reset request. Note: This bit is cleared automatically after write. This bit always read as zero.
BUFF_DP T	[31:24]	r	Buffer depth of AEI bridge Signals the buffer depth of the GTM AEI bridge implementation.
Reserved	[23:17]	r	Reserved Read as zero, should be written as zero

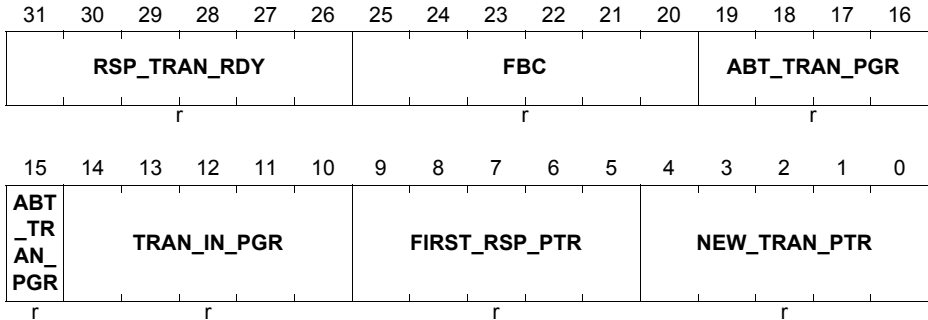
Generic Timer Module (GTM)

26.2.9.10 Register GTM_BRIDGE_PTR1

GTM_BRIDGE_PTR1

GTM to SPB BRIDGE PTR1

 (00034_H)

 Reset Value: 00X00000_H


Field	Bits	Type	Description
NEW_TRAN_PTR	[4:0]	r	New transaction pointer Signals the actual value of the new transaction pointer.
FIRST_RSP_PTR	[9:5]	r	First response pointer Signals the actual value of first response pointer.
TRAN_IN_PGR	[14:10]	r	Transaction in progress pointer (acquire) Transaction in progress pointer.
ABT_TRAN_PGR	[19:15]	r	Aborted transaction in progress pointer Aborted transaction in progress pointer.
FBC	[25:20]	r	Free buffer count Number of free buffer entries.
RSP_TRAN_RDY	[31:26]	r	Response transactions ready Amount of ready response transactions. Note: This register operates on the AEI_CLK domain.

Generic Timer Module (GTM)

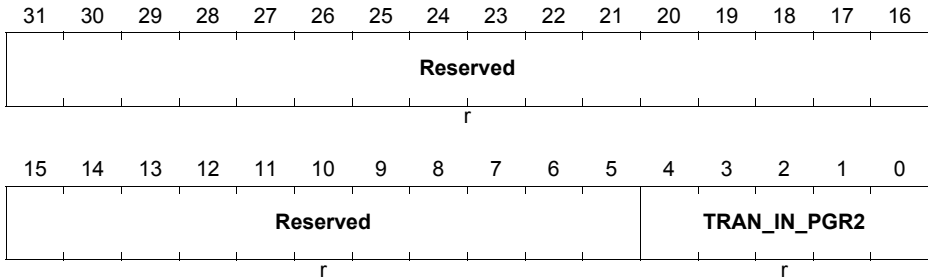
26.2.9.11 Register GTM_BRIDGE_PTR2

GTM_BRIDGE_PTR2

GTM to SPB BRIDGE PTR2

(00038_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
TRAN_IN_PGR2	[4:0]	r	Transaction in progress pointer (acquire2) Transaction in progress pointer 2.
Reserved	[31:5]	r	Reserved Read as zero, should be written as zero <i>Note: This register operates on the GTM_CLK domain</i>

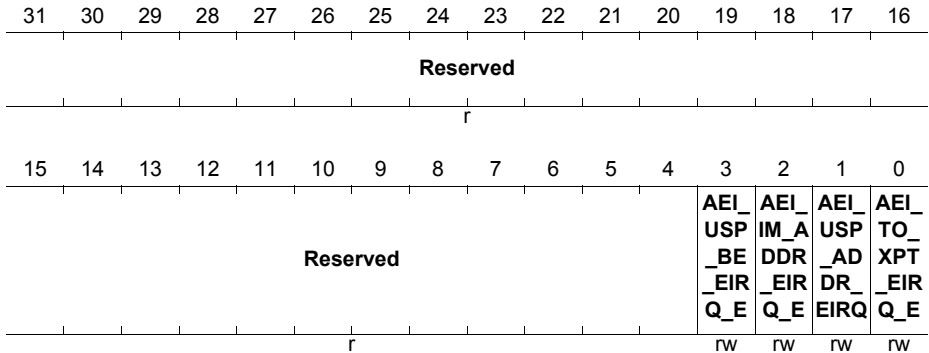
Generic Timer Module (GTM)

26.2.9.12 Register GTM_EIRQ_EN

GTM_EIRQ_EN

GTM Error Interrupt Enable Register(00020_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
AEI_TO_XPT_EIRQ_EN	0	rw	AEI_TO_XPT_EIRQ error interrupt enable 0 _B Disable error interrupt, interrupt is not visible outside GTM 1 _B Enable error interrupt, interrupt is visible outside GTM
AEI_USP_ADDR_EIRQ_EN	1	rw	AEI_USP_ADDR_EIRQ error interrupt enable 0 _B Disable error interrupt, interrupt is not visible outside GTM 1 _B Enable error interrupt, interrupt is visible outside GTM
AEI_IM_A_DDR_EIRQ_EN	2	rw	AEI_IM_A_DDR_EIRQ error interrupt enable 0 _B Disable error interrupt, interrupt is not visible outside GTM 1 _B Enable error interrupt, interrupt is visible outside GTM
AEI_USP_BE_EIRQ_EN	3	rw	AEI_USP_BE_EIRQ error interrupt enable 0 _B Disable error interrupt, interrupt is not visible outside GTM 1 _B Enable error interrupt, interrupt is visible outside GTM

Generic Timer Module (GTM)

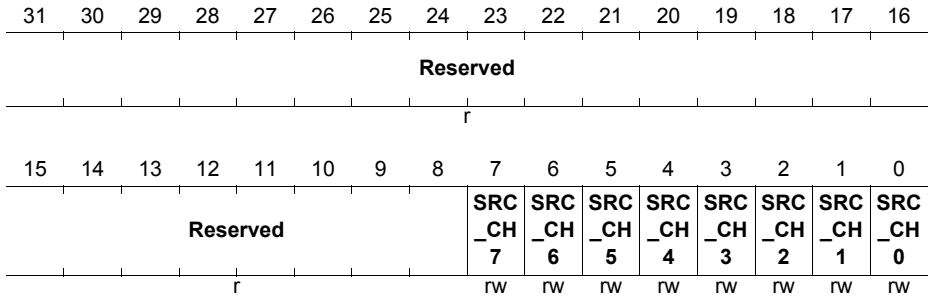
Field	Bits	Type	Description
Reserved	[31:4]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.2.9.13 Register GTM_TIMi_AUX_IN_SRC (i= 0...n)

GTM_TIMi_AUX_IN_SRC (i=0-2)

GTM TIMi AUX_IN_SRC (00040_H+i*04_H) Reset Value: 00000000_H



Field	Bits	Type	Description
SRC_CH0	0	rw	Defines AUX_IN source of TIMi channel 0 x=0 0 _B TOM Output selected TOM[a] channel [b] with a= (i* 8 +x) div 16; b=(i* 8 +x) mod 16; 1 _B ATOM Output selected ATOMi channel x
SRC_CH1	1	rw	Defines AUX_IN source of TIMi channel 1 x=1, see bit 0
SRC_CH2	2	rw	Defines AUX_IN source of TIMi channel 2 x=2, see bit 0
SRC_CH3	3	rw	Defines AUX_IN source of TIMi channel 3 x=3, see bit 0
SRC_CH4	4	rw	Defines AUX_IN source of TIMi channel 4 x=4, see bit 0
SRC_CH5	5	rw	Defines AUX_IN source of TIMi channel 5 x=5, see bit 0
SRC_CH6	6	rw	Defines AUX_IN source of TIMi channel 6 x=6, see bit 0
SRC_CH7	7	rw	Defines AUX_IN source of TIMi channel 7 x=7, see bit 0
Reserved	[31:8]	r	Reserved Read as zero, should be written as zero

26.3 Advanced Routing Unit (ARU)

26.3.1 Overview

The Advanced Routing Unit (ARU) is a flexible infrastructure component for transferring 53 bit wide data (five control bits and two 24 bit values) between several submodules of the GTM core in a configurable manner.

Since the concept of the ARU has already been described in [Section 26.2.3](#), this section only describes additional ARU features that can be used by the software for configuring and debugging ARU related data streams.

Also the definition of 'streams' and 'channels' in the ARU context is done in [Section 26.2.3](#).

26.3.2 Special Data Sources

Besides the addresses of the submodule related data sources as described in [Table 26.21.1](#), the ARU provides two special data sources that can be used for the configuration of data streams. These data sources are defined as follows:

Address 0x1FF: Data source that provides always a 53 bit data word with zeros. A read access to this memory location will never block a requesting data destination.

Address 0x1FE: Data source that never provides a data word. A read access to this memory location will always block a requesting data destination. This is the reset value of the read registers inside the data destinations.

Address 0x000: This address is reserved and can be used to bring data through the ARU registers ARU_DATA_H and ARU_DATA_L into the system by writing the write address 0x000 into the ARU_ACCESS register. This means that software test data can be brought into the GTM by the CPU.

26.3.3 ARU Access via AEI

Besides the data transfer between the connected submodules, there are two possibilities to access ARU data via the AEI.

26.3.3.1 Default ARU Access

The default ARU access incorporates the registers ARU_ACCESS, which is used for initiation of a read or write request and the registers ARU_DATA_H and ARU_DATA_L that provide the ARU data word to be transferred.

The status of a read or write transfer can be determined by polling specific bits in register ARU_ACCESS. Furthermore the acc_ack bit in the interrupt notify register is set after the read or write access is performed to avoid data loss e.g. on access cancellation.

Generic Timer Module (GTM)

A pending read or write request may also be cancelled by clearing the associated bit.

In the case of a read request, the AEI access behaves as a read request initiated by a data destination of a module. The read request is served by the ARU immediately when no other destination has a pending read request. This means, that an AEI read access does not take part in the scheduling of the destination channels and that the time between two consecutive read accesses is not limited by the round trip time.

On the other hand, the AEI access has the lowest priority behind the ARU scheduler that serves the destination channels. Thus, in worst case, the read request is served after one round trip of the ARU, when all destination channels would request data at the same point in time.

In the case of the write request, the ARU provides the write data at the address defined by the ADDR bit field inside the ARU_ACCESS register.

To avoid data loss, the reserved ARU address 0x0 has to be used to bring data into the system. Otherwise, in case the address specified inside the ADDR bit field is defined for another submodule that acts as a source at the ARU data loss may occur and no deterministic behaviour is guaranteed.

This is because the regular source submodule is not aware that its address is used by the ARU itself to provide data to a destination.

It is guaranteed that the ARU write data is send to the destination in case of both modules want to provide data at the same time.

Configuring both read and write request bits results in a read request, if the write request bit inside the register isn't already set. The following table describes the important cases of the bit 12 (RREQ) and bit 13 (WREQ) of the ARU_ACCESS register:

Table 26-8 ARU_ACCESS behaviour

AEI write access: aei_wdata(13:12)	actual value of ARU_ACCESS [13:12]	next value of ARU_ACCESS [13:12]	comment
00 _B	01 _B	00 _B	cancel read request
00 _B	10 _B	00 _B	cancel write request
01 _B	10 _B	10 _B	unchanged register
10 _B	01 _B	01 _B	unchanged register
11 _B	00 _B	01 _B	both read and write request results in a read request
11 _B	10 _B	10 _B	as before but WREQ bit is already set -> unchanged register

26.3.3.2 Debug Access

The debug access mode enables to inspect routed data of configured data streams during runtime.

The ARU provides two independent debug channels, whereas each is configured by a dedicated ARU read address in register ARU_DBG_ACCESS0 and ARU_DBG_ACCESS1 respectively.

The registers ARU_DBG_DATA0_H and ARU_DBG_DATA0_L (ARU_DBG_DATA1_H and ARU_DBG_DATA1_L) provide read access to the latest data word that the corresponding data source sent through the ARU.

Any time when data is transferred through the ARU from a data source to the destination requesting the data the interrupt signal ARU_NEW_DATA0_IRQ (ARU_NEW_DATA1_IRQ) is raised.

For advanced debugging purposes, the interrupt signal can also be triggered by software using the register ARU_IRQ_FORCINT.

Please note, that the debug mechanism should not be used by the application, when a HW-Debugger is used to trace the ARU communication. In that case, the debug registers are used by the HW-Debugger to specify the ARU streams that should be traced.

26.3.4 ARU Interrupt Signals

The following table describes ARU interrupt signals:

Table 26-9 ARU interrupt signals

Signal	Description
ARU_NEW_DATA0_IRQ	Indicates that data is transferred through the ARU using debug channel ARU_DBG_ACCESS0.
ARU_NEW_DATA1_IRQ	Indicates that data is transferred through the ARU using debug channel ARU_DBG_ACCESS1.
ACC_ACK_IRQ	ARU access acknowledge IRQ.

26.3.5 ARU Configuration Registers Overview

The following table shows a conclusion of configuration registers address offsets and initial values.

Table 26-10 ARU Configuration Registers Overview

Register name	Description	Details in Section
ARU_ACCESS	ARU access register	Section 26.3.6.1
ARU_DATA_H	ARU access register upper data word	Section 26.3.6.2

Generic Timer Module (GTM)

Table 26-10 ARU Configuration Registers Overview (cont'd)

Register name	Description	Details in Section
ARU_DATA_L	ARU access register lower data word	Section 26.3.6.3
ARU_DBG_ACCESS0	Debug access channel 0	Section 26.3.6.4
ARU_DBG_DATA0_H	Debug access 0 transfer register upper data word	Section 26.3.6.5
ARU_DBG_DATA0_L	Debug access 0 transfer register lower data word	Section 26.3.6.6
ARU_DBG_ACCESS1	Debug access channel 0	Section 26.3.6.7
ARU_DBG_DATA1_H	Debug access 1 transfer register upper data word	Section 26.3.6.8
ARU_DBG_DATA1_L	Debug access 1 transfer register lower data word	Section 26.3.6.9
ARU_IRQ_NOTIFY	ARU Interrupt notification register	Section 26.3.6.10
ARU_IRQ_EN	ARU Interrupt enable register	Section 26.3.6.11
ARU_IRQ_FORCINT	Register for forcing the ARU_NEW_DATA_IRQ interrupt	Section 26.3.6.12
ARU_IRQ_MODE	IRQ mode configuration register	Section 26.3.6.13

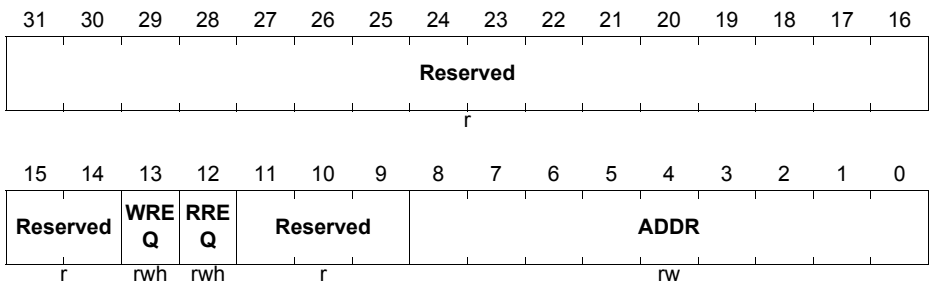
26.3.6 ARU Configuration Registers Description

All of the following registers are 32-bit only accessible.

26.3.6.1 Register ARU_ACCESS

GTM_ARU_ACCESS

ARU Access Register (00280_H) **Reset Value: 000001FE_H**



Generic Timer Module (GTM)

Field	Bits	Type	Description
ADDR	[8:0]	rw	ARU address Define the ARU address used for transferring data Note: For an ARU write request, the preferred address 0x0 have to be used. Note: A write request to the address 0x1FF (always full address) or 0x1FE (always empty address) are ignored and doesn't have any effect. Note: ARU address bits ADDR are only writable if RREQ and WREQ bits are zero
Reserved	[11:9]	r	Reserved Read as zero, should be written as zero
RREQ	12	rwh	Initiate read request 0 _B No read request is pending 1 _B Set read request to source channel addressed by ADDR Note: This bit is cleared automatically after transaction. Moreover, it can be cleared by software to cancel a read request. Note: RREQ bit is only writable if WREQ bit is zero, so to switch from RREQ to WREQ a cancel request has to be performed before. Note: Configuring both RREQ and WREQ bits results in a read request, so RREQ bit will be set if the WREQ bit of the register isn't already set. Note: The ARU read request on address ADDR is served immediately when no other destination has actually a read request when the RREQ bit is set by CPU. In a worst case scenario, the read request is served after one round trip of the ARU, but this is only the case when every destination channel issues a read request at consecutive points in time.

Generic Timer Module (GTM)

Field	Bits	Type	Description
WREQ	13	rwh	<p>Initiate write request</p> <p>0_B No write request is pending 1_B Mark data in registers ARU_DATA_H and ARU_DATA_L as valid</p> <p>Note: This bit is cleared automatically after transaction. Moreover, it can be cleared by software to cancel a write request.</p> <p>Note: WREQ bit is only writable if RREQ bit is zero, so to switch from WREQ to RREQ a cancel request has to be performed before.</p> <p>Note: Configuring both WREQ and RREQ bits result in a read request, so WREQ bit will not be set.</p> <p>Note: The data is provided at address ADDR. This address has to be programmed as the source address in the destination submodule channel. In worst case, the data is provided after one full ARU round trip.</p> <p>Note: It is strongly recommended that an address ADDR is used that is not reserved for another source submodule inside the GTM, for example the reserved address 0x000. Otherwise, data from another source submodule, that provides his data at the specified address ADDR may be lost. This can be avoided when the reserved ARU write address 0x0 is specified inside the ADDR bit field.</p>
Reserved	[31:14]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p> <p>Note: The register ARU_ACCESS can be used either for reading or for writing at the same point in time.</p>

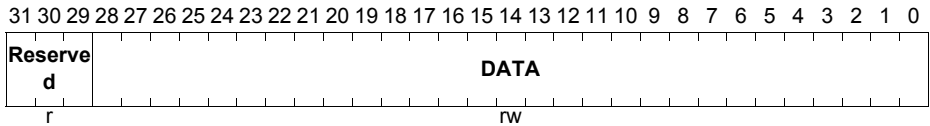
26.3.6.2 Register ARU_DATA_H

GTM_ARU_DATA_H

ARU Access Register Upper Data Word

(00284_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
DATA	[28:0]	rw	Upper ARU data word Note: Transfer upper ARU data word addressed by ADDR. The data bits 24 to 52 of an ARU word are mapped to the data bits 0 to 28 of this register
Reserved	[31:29]	r	Reserved Read as zero, should be written as zero

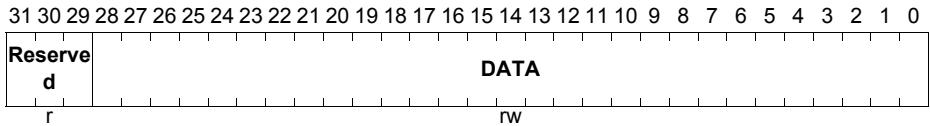
26.3.6.3 Register ARU_DATA_L

GTM_ARU_DATA_L

ARU Access Register Lower Data Word

(00288_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
DATA	[28:0]	rw	<p>Lower ARU data word</p> <p>Note: Transfer lower ARU data word addressed by ADDR. The data bits 0 to 23 of an ARU word are mapped to the data bits 0 to 23 of this register and the data bits 48 to 52 of an ARU word are mapped to the data bits 24 to 28 of this register when data is read by the CPU.</p> <p>Note: For writing data into the ARU by the CPU the bits 24 to 28 are not transferred to bit 48 to 52 of the ARU word. Only bits 0 to 23 are written to bits 0 to 23 of the ARU word</p>
Reserved	[31:29]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>

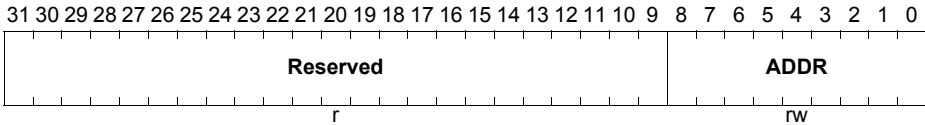
Generic Timer Module (GTM)

26.3.6.4 Register ARU_DBG_ACCESS0

GTM_ARU_DBG_ACCESS0

Debug Access Channel 0

 (0028C_H)

 Reset Value: 000001FE_H


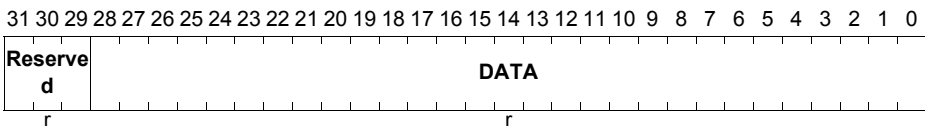
Field	Bits	Type	Description
ADDR	[8:0]	rw	ARU debugging address Note: Define address of ARU debugging channel 0.
Reserved	[31:9]	r	Reserved Read as zero, should be written as zero

26.3.6.5 Register ARU_DBG_DATA0_H

GTM_ARU_DBG_DATA0_H

Debug Access 0 Transfer Register Upper Data Word

 (00290_H)

 Reset Value: 00000000_H


Field	Bits	Type	Description
DATA	[28:0]	r	Upper debug data word Note: Transfer upper ARU data word addressed by register DBG_ACCESS0. The data bits 24 to 52 of an ARU word are mapped to the data bits 0 to 28 of this register Note: The interrupt ARU_NEW_DATA0_IRQ is raised if a new data word is available.
Reserved	[31:29]	r	Reserved Read as zero, should be written as zero

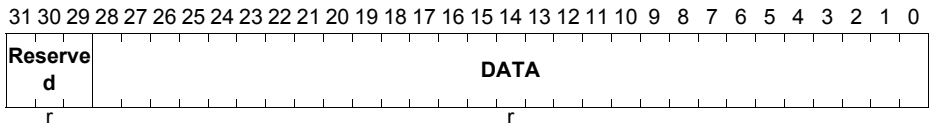
Generic Timer Module (GTM)

26.3.6.6 Register ARU_DBG_DATA0_L

GTM_ARU_DBG_DATA0_L

Debug Access 0 Transfer Register Lower Data Word
(00294_H)

Reset Value: 00000000_H



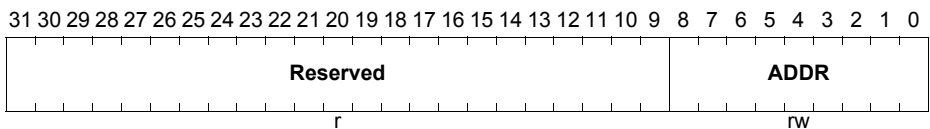
Field	Bits	Type	Description
DATA	[28:0]	r	Lower debug data word Note: Transfer lower ARU data word addressed by register DBG_ACCESS0. The data bits 0 to 23 of an ARU word are mapped to the data bits 0 to 23 of this register and the data bits 48 to 52 of an ARU word is mapped to the data bits 24 to 28 of this register. Note: The interrupt ARU_NEW_DATA0_IRQ is raised if a new data word is available.
Reserved	[31:29]	r	Reserved Read as zero, should be written as zero

26.3.6.7 Register ARU_DBG_ACCESS1

GTM_ARU_DBG_ACCESS1

Debug Access Channel 0 (00298_H)

Reset Value: 000001FE_H



Field	Bits	Type	Description
ADDR	[8:0]	rw	ARU debugging address Note: Define address of ARU debugging channel 1.
Reserved	[31:9]	r	Reserved Read as zero, should be written as zero

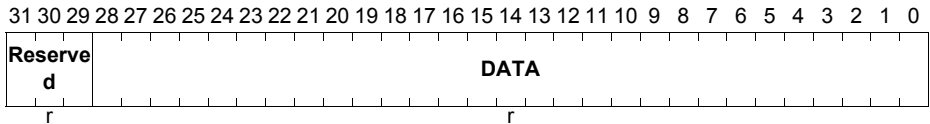
26.3.6.8 Register ARU_DBG_DATA1_H

GTM_ARU_DBG_DATA1_H

Debug Access 1 Transfer Register Upper Data Word

(0029C_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
DATA	[28:0]	r	Upper debug data word Note: Transfer upper ARU data word addressed by register DBG_ACCESS1. The data bits 24 to 52 of an ARU word are mapped to the data bits 0 to 28 of this register Note: The interrupt ARU_NEW_DATA1_IRQ is raised if a new data word is available.
Reserved	[31:29]	r	Reserved Read as zero, should be written as zero

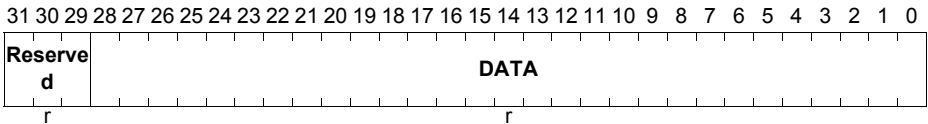
26.3.6.9 Register ARU_DBG_DATA1_L

GTM_ARU_DBG_DATA1_L

Debug Access 1 Transfer Register Lower Data Word

(002A0_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
DATA	[28:0]	r	<p>Lower debug data word</p> <p>Note: Transfer lower ARU data word addressed by register DBG_ACCESS1. The data bits 0 to 23 of an ARU word are mapped to the data bits 0 to 23 of this register and the data bits 48 to 52 of an ARU word is mapped to the data bits 24 to 28 of this register.</p> <p>Note: The interrupt ARU_NEW_DATA1_IRQ is raised if a new data word is available.</p>
Reserved	[31:29]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>

26.3.6.10 Register ARU_IRQ_NOTIFY

GTM_ARU_IRQ_NOTIFY

ARU Interrupt Notification Register (002A4_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
NEW_DAT A0	0	rwh	<p>Data was transferred for addr ARU_DBG_ACCESS0</p> <p>0_B No interrupt occurred 1_B ARU_NEW_DATA0_IRQ interrupt was raised by the ARU</p> <p>Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.</p>
NEW_DAT A1	1	rwh	<p>Data was transferred for addr ARU_DBG_ACCESS1</p> <p>0_B No interrupt occurred 1_B ARU_NEW_DATA1_IRQ interrupt was raised by the ARU</p> <p>Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.</p>
ACC_ACK	2	rwh	<p>AEI to ARU access finished, on read access data are valid</p> <p>Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.</p>
Reserved	[31:3]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>

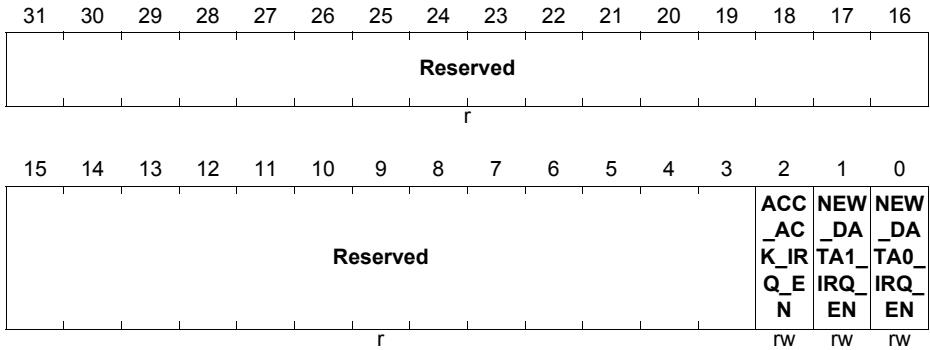
Generic Timer Module (GTM)

26.3.6.11 Register ARU_IRQ_EN

GTM_ARU_IRQ_EN

ARU Interrupt Enable Register (002A8_H)

Reset Value: 00000000_H



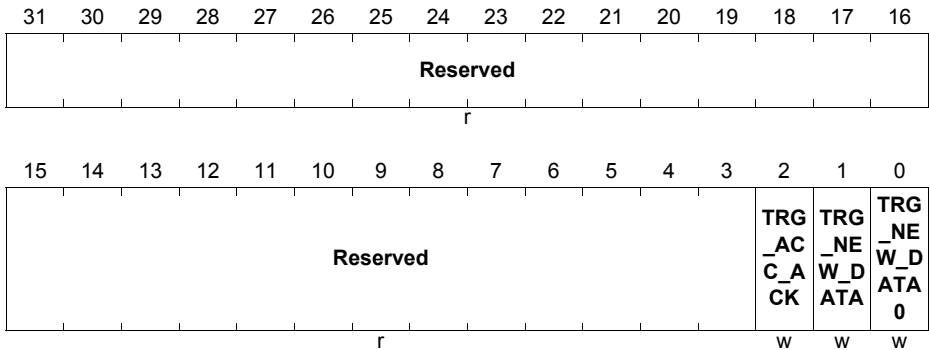
Field	Bits	Type	Description
NEW_DATA0_IRQ_EN	0	rw	ARU_NEW_DATA0_IRQ interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
NEW_DATA1_IRQ_EN	1	rw	ARU_NEW_DATA1_IRQ interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
ACC_ACK_IRQ_EN	2	rw	ACC_ACK_IRQ interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
Reserved	[31:3]	r	Reserved Read as zero, should be written as zero

26.3.6.12 Register ARU_IRQ_FORCINT

GTM_ARU_IRQ_FORCINT

ARU_NEW_DATA_IRQ Forcing Interrupt Register (002AC_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
TRG_NEW_DATA0	0	w	<p>Trigger new data 0 interrupt</p> <p>0_B corresponding bit in status register will not be forced</p> <p>1_B Assert corresponding field in ARU_IRQ_NOTIFY register</p> <p>Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.</p>
TRG_NEW_DATA1	1	w	<p>1 Trigger new data 1 interrupt</p> <p>0_B corresponding bit in status register will not be forced</p> <p>1_B Assert corresponding field in ARU_IRQ_NOTIFY register</p> <p>Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG_ACC_ACK	2	w	<p>Trigger ACC_ACK interrupt</p> <p>0_B corresponding bit in status register will not be forced</p> <p>1_B Assert corresponding field in ARU_IRQ_NOTIFY register</p> <p>Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.</p>
Reserved	[31:3]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>

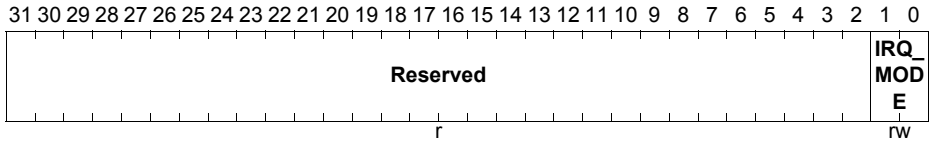
Generic Timer Module (GTM)

26.3.6.13 Register ARU_IRQ_MODE

GTM_ARU_IRQ_MODE

IRQ Mode Configuration Register (002B0_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
IRQ_MODE	[1:0]	rw	IRQ mode selection 00 _B Level mode 01 _B Pulse mode 10 _B Pulse-Notify mode 11 _B Single-Pulse mode Note: The interrupt modes are described in Section 26.2.5 .
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

26.4 Broadcast Module (BRC)

26.4.1 Overview

Since each write address for the submodule channels of the GTM that are able to write to the ARU can only be read by a single module, it is impossible to provide a data stream to different modules in parallel (This statement holds not for sources, which do not invalidate their data after the data were read by any consumer, e.g. DPLL).

To overcome this issue for regular modules, the submodule Broadcast (BRC) enables to duplicate data streams multiple times.

The BRC submodule provides 12 input channels as well as 22 output channels.

In order to clone an incoming data stream, the corresponding input channel can be mapped to zero or more output channels.

When mapped to zero no channel is read.

To destroy an incoming data stream, the EN_TRASHBIN bit inside the BRC_SRCx_DEST register has to be set.

The total number of output channels that are assigned to a single input channel is variable. However, the total number of assigned output channels must be less than or equal to 22.

26.4.2 BRC Configuration

As it is the case with all other submodules connected to the ARU, the input channels can read arbitrary ARU address locations and the output channels provide the broadcast data to fixed ARU write address locations.

The associated write addresses for the BRC submodule are fixed and can be obtained from [Section 26.21](#).

The read address for each input channel is defined by the corresponding register BRC_SRCx_ADDR (x: 0...1).

The mapping of an input channel to several output channels is defined by setting the appropriate bits in the register BRC_SRCx_DEST (x: 0...11). Each output channel is represented by a single bit in the register BRC_SRCx_DEST. The address of the output channel is defined in [Section 26.21](#).

If no output channel bit is set within a register BRC_SRCx_DEST, no data is provided to the corresponding ARU write address location from the defined read input specified by BRC_SRCx_ADDR. This means that the channel does not broadcast any data and is disabled (reset state).

Besides the possibility of mapping an input channel to several output channels, the bit EN_TRASHBIN of register BRC_SRCx_DEST may be set, which results in dropping an

Generic Timer Module (GTM)

incoming data stream. In this case the data of an input channel defined by `BRC_SRCx_ADDR` is consumed by the BRC module and not routed to any succeeding submodule. In consequence, the output channels defined in the register `BRC_SRCx_DEST` are ignored. Therefore, the bits 0 to 21 are set to zero (0) when trash bin functionality is enabled.

In general, the BRC submodule can work in two independent operation modes. In the first operation mode the data consistency is guaranteed since a BRC channel requests only new data from a source when all destination channels for the BRC have consumed the old data value. This mode is called Data Consistency Mode (DCM).

In a second operation mode the BRC channel always requests data from a source and distributes this data to the destination regardless whether all destinations have already consumed the old data. This mode is called Maximum Throughput Mode (MTM).

MTM ensures, that always the newest available data is routed through the system, while it is not guaranteed data consistency since some of the destination channels can be provided with the old data while some other destination channels are provided with the new data. If this is the case, the Data Inconsistency Detected Interrupt `BRC_DID_IRQ[x]` is raised but the channel continues to work.

Furthermore in MTM mode it is guaranteed that it is not possible to read a data twice by a read channel. This is blocked.

The channel mode can be configured inside the `BRC_SRCx_ADDR` register.

To avoid invalid configurations of the registers `BRC_SRCx_DEST`, the BRC also implements a plausibility check for these configurations. If the software assigns an already used output channel to a second input channel, BRC performs an auto correction of the lastly configured register `BRC_SRCx_DEST` and it triggers the interrupt `BRC_DEST_ERR`.

Consider the following example for clarification of the auto correction mechanism. Assume that the following configuration of the 22 lower significant bits for the registers `BRC_SRCx_DEST`:

`BRC_SRC_0_DEST`: 00 0000 0000 1000 1000 0000 (binary)

`BRC_SRC_1_DEST`: 00 0000 0000 0100 0000 0100 (binary)

`BRC_SRC_2_DEST`: 00 0000 0000 0001 0100 0010 (binary)

`BRC_SRC_3_DEST`: 00 0000 0000 0010 0001 1001 (binary)

If the software overwrites the value for register `BRC_SRC_2_DEST` with

`BRC_SRC_2_DEST`: 00 0000 0000 1001 0010 0010 (binary)

(changed bits are underlined), then the BRC releases a `BRC_DEST_ERR` interrupt since bit 11 is already assigned in register `BRC_SRC_0_DEST`. The auto correction forces bit 11 to be cleared. The modifications of the bits 5 and 6 are accepted, since there is no violation with previous configurations. So the result of the write access mentioned above results in the following modified register configuration:

Generic Timer Module (GTM)

BRC_SRC_2_DEST: 00 0000 0000 0001 0010 0010 (binary)

For debug purposes, the interrupt BRC_DEST_ERR can also be released by writing to register BRC_IRQ_FORCINT. Nevertheless, the interrupt has to be enabled to be visible outside of the GTM.

26.4.3 BRC Interrupt Signals

Interrupt signals are defined in following table:

Table 26-11 BRC Interrupts

Signal	Description
BRC_DEST_ERR_IRQ	Indicating configuration errors for BRC module
BRC_DID_IRQ[x]	Data inconsistency occurred in MTM mode (x=0-11)

26.4.4 BRC Configuration Registers Overview

Following table shows a conclusion of configuration registers address offsets and initial values.

Table 26-12 BRC Registers

Register Name	Description	Details in Section
BRC_SRCx_ADDR	Read address for input channel x (x=0-11)	Section 26.4.5.1
BRC_SRCx_DEST	Destination channels for input channel x (x=0-11)	Section 26.4.5.2
BRC_IRQ_NOTIFY	BRC Interrupt notification register	Section 26.4.5.3
BRC_IRQ_EN	BRC Interrupt enable register	Section 26.4.5.4
BRC_IRQ_EIRQ_EN	BRC Error interrupt enable register	Section 26.4.5.8
BRC_IRQ_FORCINT	Register for forcing the BRC_DEST_ERR interrupt	Section 26.4.5.5
BRC_RST	Software reset	Section 26.4.5.7
BRC_IRQ_MODE	IRQ mode configuration register	Section 26.4.5.6

26.4.5 BRC Configuration Registers Description

All of the following registers are 32-bit only accessible.

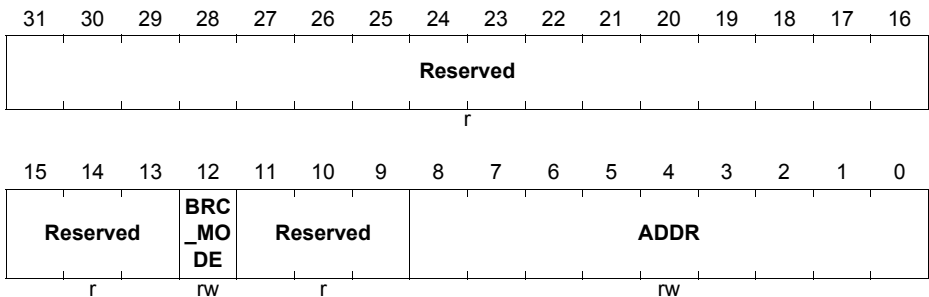
26.4.5.1 Register BRC_SRCx_ADDR (x=0-11)

GTM_BRC_SRCx_ADDR (x=0-11)

Read Address For Input Channel x

(00400_H+x*08_H)

Reset Value: 000001FE_H



Field	Bits	Type	Description
ADDR	[8:0]	rw	Source ARU address Define an ARU read address used as data source for input channel x (x011)
Reserved	[11:9]	r	Reserved Read as zero, should be written as zero
BRC_MODE	12	rw	BRC Operation mode select 0 _B Consistency Mode (DCM) selected 1 _B Maximum Throughput Mode (MTM) selected Note: this bit field is only writable if channel is disabled.
Reserved	[31:13]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.4.5.2 Register BRC_SRCx_DEST (x:0...11)

GTM_BRC_SRCx_DEST (x=0-11)

Destination Channels For Input Channel x

 $(00404_H + x \cdot 08_H)$

 Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									EN_TRA SHBI N	EN_DES T21	EN_DES T20	EN_DES T19	EN_DES T18	EN_DES T17	EN_DES T16
r									rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN_DES T15	EN_DES T14	EN_DES T13	EN_DES T12	EN_DES T11	EN_DES T10	EN_DES T9	EN_DES T8	EN_DES T7	EN_DES T6	EN_DES T5	EN_DES T4	EN_DES T3	EN_DES T2	EN_DES T1	EN_DES T0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
EN_DEST 0	0	rw	<p>Enable BRC destination address 0</p> <p>0_B Destination address 0 not mapped to source BRC_SRCx_ADDR</p> <p>1_B Destination address 0 mapped to source BRC_SRCx_ADDR</p> <p><i>Note:</i> Note: The destination address 0 for BRC channel is defined in Section 26.21.1</p> <p><i>Note:</i> The bit is cleared by auto correction mechanism if a destination channel is assigned to multiple source channels.</p> <p><i>Note:</i> When a BRC input channel is disabled (all EN_DESTx (x: 0...21) bits are cleared) the internal states are reset to their reset value.</p>
EN_DEST 1	1	rw	<p>Enable BRC destination address 1</p> <p>See bit 0</p>
EN_DEST 2	2	rw	<p>Enable BRC destination address 2</p> <p>See bit 0</p>
EN_DEST 3	3	rw	<p>Enable BRC destination address 3</p> <p>See bit 0</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
EN_DEST 4	4	rw	Enable BRC destination address 4 See bit 0
EN_DEST 5	5	rw	Enable BRC destination address 5 See bit 0
EN_DEST 6	6	rw	Enable BRC destination address 6 See bit 0
EN_DEST 7	7	rw	Enable BRC destination address 7 See bit 0
EN_DEST 8	8	rw	Enable BRC destination address 8 See bit 0
EN_DEST 9	9	rw	Enable BRC destination address 9 See bit 0
EN_DEST 10	10	rw	Enable BRC destination address 10 See bit 0
EN_DEST 11	11	rw	Enable BRC destination address 11 See bit 0
EN_DEST 12	12	rw	Enable BRC destination address 12 See bit 0
EN_DEST 13	13	rw	Enable BRC destination address 13 See bit 0
EN_DEST 14	14	rw	Enable BRC destination address 14 See bit 0
EN_DEST 15	15	rw	Enable BRC destination address 15 See bit 0
EN_DEST 16	16	rw	Enable BRC destination address 16 See bit 0
EN_DEST 17	17	rw	Enable BRC destination address 17 See bit 0
EN_DEST 18	18	rw	Enable BRC destination address 18 See bit 0
EN_DEST 19	19	rw	Enable BRC destination address 19 See bit 0
EN_DEST 20	20	rw	Enable BRC destination address 20 See bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
EN_DEST 21	21	rw	Enable BRC destination address 21 See bit 0
EN_TRAS HBIN	22	rw	Control trash bin functionality 0 _B Trash bin functionality disabled 1 _B Trash bin functionality enabled Note: When bit EN_TRASHBIN is enabled bits 0 to 21 are ignored for this input channel. Therefore, the bits 0 to 21 are set to zero (0) when trashbin functionality is enabled.
Reserved	[31:23]	r	Reserved Read as zero, should be written as zero

26.4.5.3 Register BRC_IRQ_NOTIFY
GTM_BRC_IRQ_NOTIFY
BRC Interrupt Notification Register (00460_H)
Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DID1 1	DID1 0	DID9	DID8	DID7	DID6	DID5	DID4	DID3	DID2	DID1	DID0	DES T_E RR	
r		rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
DEST_ER R	0	rwh	Configuration error interrupt for BRC submodule 0 _B No BRC configuration error occurred 1 _B BRC configuration error occurred Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
DIDx (x = 0-11)	x+1	rwh	Data inconsistency occurred in MTM mode Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:13]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.4.5.4 Register BRC_IRQ_EN

GTM_BRC_IRQ_EN

BRC Interrupt Enable Register

 (00464_H)

 Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DID_EN1 1	DID_EN1 0	DID_EN9	DID_EN8	DID_EN7	DID_EN6	DID_EN5	DID_EN4	DID_EN3	DID_EN2	DID_EN1	DID_EN0	DEST_ERR_EN
r			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
DEST_ERR_EN	0	rw	BRC_DEST_ERR_IRQ interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
DID_ENx (x = 0-11)	x+1	rw	BRC_DIDx_IRQ interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
Reserved	[31:13]	r	Reserved Read as zero, should be written as zero

26.4.5.5 Register BRC_IRQ_FORCINT

GTM_BRC_IRQ_FORCINT

BRC_DEST_ERR Forcing Interrupt Register (00468_H)

 Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			TRG _DID 11	TRG _DID 10	TRG _DID 9	TRG _DID 8	TRG _DID 7	TRG _DID 6	TRG _DID 5	TRG _DID 4	TRG _DID 3	TRG _DID 2	TRG _DID 1	TRG _DID 0	TRG _DE ST ERR
r			w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
TRG_DEST_ERR	0	w	Trigger destination error interrupt 0 _B corresponding bit in status register will not be forced 1 _B Assert corresponding field in BRC_IRQ_NOTIFY register Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
TRG_DIDx (x = 0-11)	x+1	w	Trigger data inconsistency error interrupt 0 _B corresponding bit in status register will not be forced 1 _B Assert corresponding field in BRC_IRQ_NOTIFY register Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
Reserved	[31:13]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

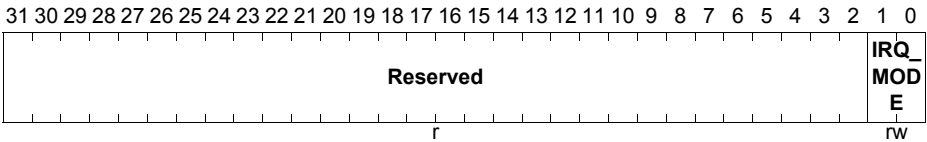
26.4.5.6 Register BRC_IRQ_MODE

GTM_BRC_IRQ_MODE

BRC IRQ Mode Configuration Register

(0046C_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
IRQ_MODE	[1:0]	rw	IRQ mode selection 00 _B Level mode 01 _B Pulse mode 10 _B Pulse-Notify mode 11 _B Single-Pulse mode Note: The interrupt modes are described in Section 26.2.5 .
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

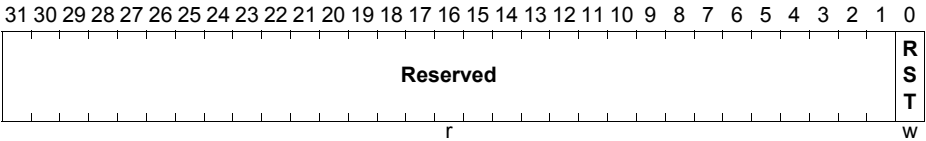
Generic Timer Module (GTM)

26.4.5.7 Register BRC_RST

GTM_BRC_RST

BRC Software Reset Register (00470_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
RST	0	w	Software reset 0 _B No action 1 _B Reset BRC Note: This bit is cleared automatically after write by CPU. The channel registers are set to their reset values and channel operation is stopped immediately.
Reserved	[31:1]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.4.5.8 Register BRC_EIRQ_EN

GTM_BRC_EIRQ_EN

BRC Error Interrupt Enable Register (00474_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DID_ EN1 1	DID_ EN1 0	DID_ EN9	DID_ EN8	DID_ EN7	DID_ EN6	DID_ EN5	DID_ EN4	DID_ EN3	DID_ EN2	DID_ EN1	DID_ EN0	DES T_E RR_ EN
r			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
DEST_ERR_EN	0	rw	BRC_DEST_ERR_EIRQ error interrupt enable 0 _B Disable error interrupt, error interrupt is not visible outside GTM 1 _B Enable error interrupt, error interrupt is visible outside GTM
DID_ENx (x = 0-11)	x+1	rw	BRC_DIDx_EIRQ error interrupt enable 0 _B Disable error interrupt, error interrupt is not visible outside GTM 1 _B Enable error interrupt, error interrupt is visible outside GTM
Reserved	[31:13]	r	Reserved Read as zero, should be written as zero

26.5 First In First Out Module (FIFO)

26.5.1 Overview

The FIFO unit is the storage part of the FIFO submodule. The F2A described in [Section 26.7](#) and the AFD described in [Section 26.6](#) implement the interface part of the FIFO submodule to the ARU and the AEI bus. Each FIFO unit embeds eight logical FIFOs. These logical FIFOs are configurable in the following manner:

- FIFO size (defines start and end address)
- FIFO operation modes (normal mode or ring buffer operation mode)
- Fill level control / memory region read protection

Each logical FIFO represents a data stream between the submodules of the GTM and the microcontroller connected to AFD submodule (see [Section 26.6](#)). The FIFO RAM counts 1K words, where the word size is 29 bit. This gives the freedom to program or receive 24 bit of data together with the five control bits inside an ARU data word.

The FIFO unit provides three ports for accessing its content. One port is connected to the F2A interface, one port is connected to the AFD interface and one port has its own AEI bus interface.

The AFD interface has always the highest priority. Accesses to the FIFO from AFD interface and direct AEI interface in parallel - which means at the same time - is not possible, because both interfaces are driven from the same AEI bus interface of the GTM.

The priority between F2A and direct AEI interface can be defined by software. This can be done by using the register FIFO0_CHx_CTRL for all FIFO channels of the submodule.

The FIFO is organized as a single RAM that is also accessible through the FIFO AEI interface connected to one of the FIFO ports. To provide the direct RAM access, the RAM is mapped into the address space of the microcontroller.

After reset, the FIFO RAM isn't initialized.

The FIFO channels can be flushed individually. Each of the eight FIFO channels can be used whether in normal FIFO operation mode or in ring buffer operation mode.

Beside the possibility of flushing each FIFO channel directly, a write access to FIFOi_CHx_END_ADDR or to FIFOi_CHx_START_ADDR will also flush the regarding channel which means that the read and write pointer and also the fill level of the regarding channel will be reset. In consequence of this existing data in the concerned FIFO channel are not longer valid- thereafter the channel is empty.

26.5.2 Operation Modes

26.5.2.1 Normal Operation Mode

In normal FIFO operation mode the content of the FIFO is written and read in first-in first-out order, where the data is destroyed after it is delivered to the system bus or the F2A submodule (see [Section 26.7](#)).

The upper and lower watermark registers (registers FIFO0_CHx_UPPER_WM and FIFO0_CHx_LOWER_WM) are used for controlling the FIFO's fill level. If the fill level declines the lower watermark or it exceeds the upper watermark, an interrupt signal is triggered by the FIFO submodule if enabled inside the FIFO0_IRQ_EN.

The interrupt signals are sending to the Interrupt Concentrator Module (ICM) (see [Section 26.18](#)). The ICM can also initiate specific DMA transfers.

26.5.2.2 Ring Buffer Operation Mode

The ring buffer mode is a powerful tool to provide a continuous data or configuration stream to the other GTM submodules without CPU interaction. In ring buffer mode the FIFO provides a continuous data stream to the F2A submodule. The first word of the FIFO is delivered first and after the last word is provided by the FIFO to the ARU, the first word can be obtained again.

If in ring buffer mode the read pointer reaches the write pointer it will be set again to the configured start address. So the read pointer always rotates cyclic between the configured start address of the regarding FIFO channel (first written data) and the write pointer which points to the last written data of the channel.

It is possible to add data via the AEI to FIFO interface (AFD) to the FIFO channel while running in ring buffer mode. The new written data will be added in the next ring buffer cycle.

It is recommended to fill the FIFO channel first before enabled the data stream in the FIFO to ARU interface (F2A).

There could be the requirement that the user must be able to change some data inside the continuous data stream to the GTM submodules or system bus. This is possible through direct memory access provided by the FIFO AEI interface.

26.5.3 FIFO Interrupt Signals

Interrupt signals are defined in following table:

Table 26-13 FIFO Interrupt Signals

Signal	Description
FIFOi_CH[x]_EMPTY	Indicating empty FIFOi x (x:0...7) was reached
FIFOi_CH[x]_FULL	Indicating full FIFOi x (x:0...7) was reached
FIFOi_CHx_LOWER_WM	Indicating FIFOi x (x:0...7) reached lower watermark.
FIFOi_CHx_UPPER_WM	Indicating FIFOi x (x:0...7) reached upper watermark.

26.5.4 FIFOi Configuration Registers Overview

The following table shows a conclusion of configuration registers address offsets and initial values:

Table 26-14 FIFOi Configuration Registers Overview

Register Name	Description	Details in Section
FIFOi_CHx_CTRL	FIFOi Channel x control register (x: 0...7)	Section 26.5.5.1
FIFOi_CHx_END_ADDR	FIFOi Channel x end address register (x: 0...7)	Section 26.5.5.2
FIFOi_CHx_START_ADDR	FIFOi Channel x start address register (x: 0...7)	Section 26.5.5.3
FIFOi_CHx_UPPER_WM	FIFOi Channel x upper watermark register (x: 0...7)	Section 26.5.5.4
FIFOi_CHx_LOWER_WM	FIFOi Channel x lower watermark register (x: 0...7)	Section 26.5.5.5
FIFOi_CHx_STATUS	FIFOi Channel x status register (x: 0...7)	Section 26.5.5.6
FIFOi_CHx_FILL_LEVEL	FIFOi Channel x fill level register (x: 0...7)	Section 26.5.5.7
FIFOi_CHx_WR_PTR	FIFOi Channel x write pointer register (x: 0...7)	Section 26.5.5.8
FIFOi_CHx_RD_PTR	FIFOi Channel x read pointer register (x: 0...7)	Section 26.5.5.9
FIFOi_CHx_IRQ_NOTIFY	FIFOi x interrupt notification register (x: 0...7)	Section 26.5.5.10

Generic Timer Module (GTM)

Table 26-14 FIFOi Configuration Registers Overview (cont'd)

Register Name	Description	Details in Section
FIFOi_CHx_IRQ_EN	FIFOi x interrupt enable register (x: 0...7)	Section 26.5.5.11
FIFOi_CHx_EIRQ_EN	FIFO x Error interrupt enable register	Section 26.5.5.14
FIFOi_CHx_IRQ_FORCINT	FIFOi x register to force interrupt by software (x: 0...7)	Section 26.5.5.12
FIFOi_CHx_IRQ_MODE	FIFOi x IRQ mode control register (x: 0...7)	Section 26.5.5.13

26.5.5 FIFOi Configuration Registers Description

All of the following registers are 32-bit only accessible.

26.5.5.1 Register FIFOi_CHx_CTRL (x:0...7)

GTM_FIFO0_CHx_CTRL (x=0-7)

FIFO0 Channel x Control Register(18400_H+x*40_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
RBM	0	rw	Ring buffer mode enable 0 _B Normal FIFO operation mode 1 _B Ring buffer mode
RAP	1	rw	RAM access priority 0 _B FIFO ports have higher access priority than AEI-IF 1 _B AEI-IF has higher access priority than FIFO ports Note: RAP bit is only functional in register FIFOi_CHx_CTRL. The priority is defined for all FIFO channels there
FLUSH	2	w	FIFO Flush control 0 _B Normal operation 1 _B Execute FIFO flush (bit is automatically cleared after flush). Note: A FIFO Flush operation resets the FIFOi_CHx_FILL_LEVEL, FIFOi_CHx_WR_PTR and FIFOi_CHx_RD_PTR registers to their initial values.

Generic Timer Module (GTM)

Field	Bits	Type	Description
WULOCK	3	rw	RAM write unlock Enable/disable direct RAM write access to the memory mapped FIFO region 0_B Direct RAM write access disabled 1_B Direct RAM write access enabled Note: Only the bit WULOCK of register FIFOi_CH0_CTRL enables/disables the direct RAM write access for all FIFO channels (whole FIFO RAM). The WULOCK bits of the other channels are writable but have no effect.
Reserved	[31:4]	r	Reserved Read as zero, should be written as zero

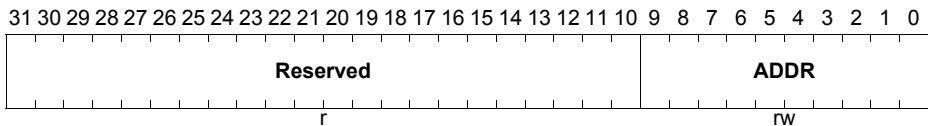
26.5.5.2 Register FIFOi_CHx_END_ADDR (x:0...7)

GTM_FIFO0_CHx_END_ADDR (x=0-7)

FIFO0 Channel x End Address Register

$$(18404_H + x * 40_H)$$

Reset Value: 00000XXX_H



Field	Bits	Type	Description
ADDR	[9:0]	rw	End address for FIFOi channel x, (x:0...7) Note: Note: value for ADDR is calculated as $ADDR = 128 * (x + 1) - 1$. Note: A write access will flush the regarding channel.
Reserved	[31:10]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

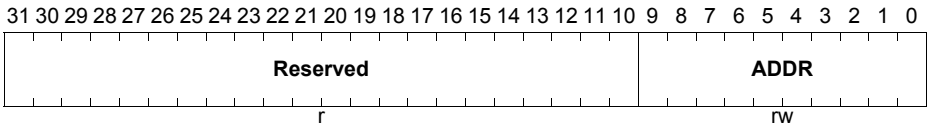
26.5.5.3 Register FIFOi_CHx_START_ADDR (x:0...7)

GTM_FIFO0_CHx_START_ADDR (x=0-7)

FIFO0 Channel x Start Address Register

(18408_H+x*40_H)

Reset Value: 00000XXX_H



Field	Bits	Type	Description
ADDR	[9:0]	rw	Start address for FIFOi channel x, (x07) <i>Note: Note: Initial value for ADDR is calculated as ADDR = 128*x</i> <i>Note: A write access will flush the regarding channel.</i>
Reserved	[31:10]	r	Reserved Read as zero, should be written as zero

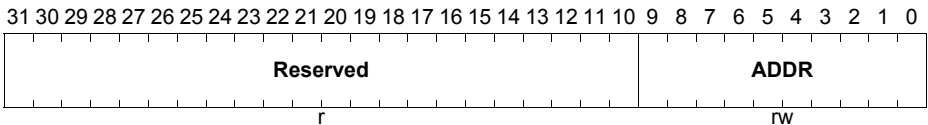
26.5.5.4 Register FIFOi_CHx_UPPER_WM (x:0...7)

GTM_FIFO0_CHx_UPPER_WM (x=0-7)

FIFO0 Channel x Upper Watermark Register

(1840C_H+x*40_H)

Reset Value: 00000060_H



Field	Bits	Type	Description
ADDR	[9:0]	rw	Upper watermark address Note: The upper watermark is configured as a relative fill level of the FIFOi. ADDR must be in range: 0 <= ADDR <= FIFOi_CHx_END_ADDR - FIFOi_CHx_START_ADDR. Initial value for ADDR is defined as ADDR = 0x60.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:10]	r	Reserved Read as zero, should be written as zero

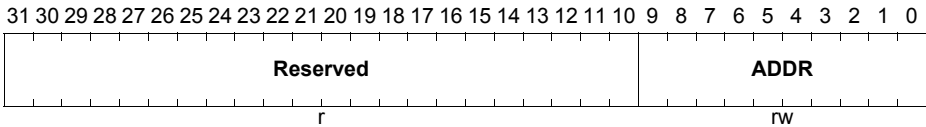
26.5.5.5 Register FIFOi_CHx_LOWER_WM (x:0...7)

GTM_FIFO0_CHx_LOWER_WM (x=0-7)

FIFO0 Channel x Lower Watermark Register

(18410_H+x*40_H)

Reset Value: 00000020_H



Field	Bits	Type	Description
ADDR	[9:0]	rw	Lower watermark address Note: The lower watermark is configured as a relative fill level of the FIFOi. ADDR must be in range: 0 <= ADDR <= FIFOi_CHx_END_ADDR - FIFOi_CHx_START_ADDR. Initial value for ADDR is defined as ADDR = 0x20.
Reserved	[31:10]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)
26.5.5.6 Register FIFOi_CHx_STATUS (x:0...7)
GTM_FIFO0_CHx_STATUS (x=0-7)
FIFO0 Channel x Status Register (18414_H+x*40_H)
Reset Value: 00000005_H


Field	Bits	Type	Description
EMPTY	0	r	FIFO is empty 0 _B Fill level > 0 1 _B Fill level = 0 Note: Bit only applicable in normal mode
FULL	1	r	FIFO is full 0 _B Fill level < FIFOi_CHx_END_ADDR – FIFOi_CHx_START_ADDR + 1 1 _B Fill level = FIFOi_CHx_END_ADDR – FIFOi_CHx_START_ADDR + 1 Note: Bit only applicable in normal mode
LOW_WM	2	r	Lower watermark reached 0 _B Fill level > lower watermark 1 _B Fill level <= lower watermark Note: Bit only applicable in normal mode
UP_WM	3	r	Upper watermark reached 0 _B Fill level < upper watermark 1 _B Fill level >= upper watermark Note: Bit only applicable in normal mode
Reserved	[31:4]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

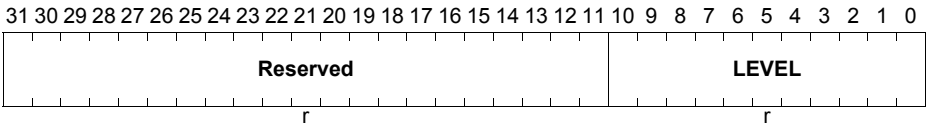
26.5.5.7 Register FIFOi_CHx_FILL_LEVEL (x:0...7)

GTM_FIFO0_CHx_FILL_LEVEL (x=0-7)

FIFO0 Channel x Fill Level Register

(18418_H+x*40_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
LEVEL	[10:0]	r	Fill level of the current FIFO Note: LEVEL is in range: $0 \leq \text{LEVEL} \leq \text{FIFOi_CHx_END_ADDR} - \text{FIFOi_CHx_START_ADDR} + 1$. Register content is compared to the upper and lower watermark values for this channel to detect watermark over- and underflow.
Reserved	[31:11]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

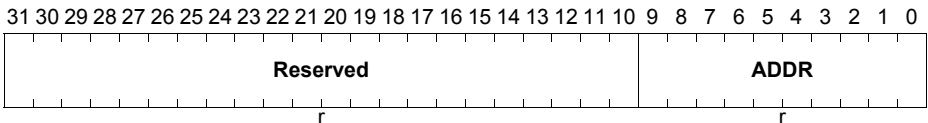
26.5.5.8 Register FIFOi_CHx_WR_PTR (x:0...7)

GTM_FIFO0_CHx_WR_PTR (x=0-7)

FIFO0 Channel x Write Pointer Register

(1841C_H+x*40_H)

Reset Value: 00000XXX_H



Field	Bits	Type	Description
ADDR	[9:0]	r	Position of the write pointer Note: ADDR must be in range $0 \leq \text{ADDR} \leq 1023$. Initial value for ADDR is defined as ADDR = FIFOi_CHx_START_ADDR
Reserved	[31:10]	r	Reserved Read as zero, should be written as zero

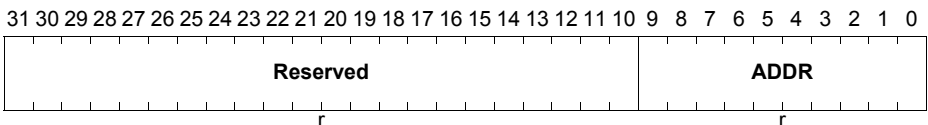
26.5.5.9 Register FIFOi_CHx_RD_PTR (x:0...7)

GTM_FIFO0_CHx_RD_PTR (x=0-7)

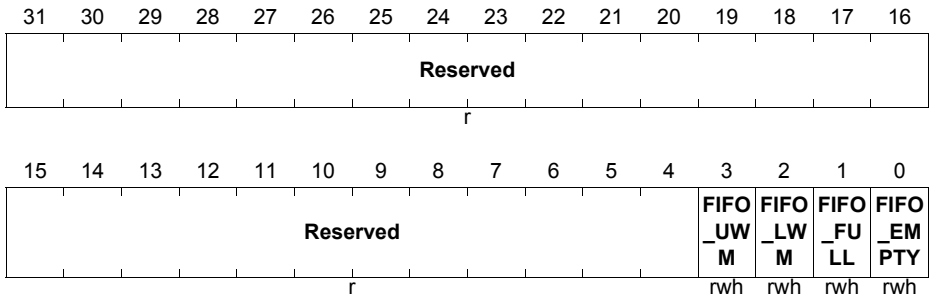
FIFO0 Channel x Read Pointer Register

(18420_H+x*40_H)

Reset Value: 00000XXX_H



Field	Bits	Type	Description
ADDR	[9:0]	r	Position of the read pointer Note: ADDR must be in range $0 \leq \text{ADDR} \leq 1023$. Initial value for ADDR is defined as ADDR = FIFOi_CHx_START_ADDR
Reserved	[31:10]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)
26.5.5.10 Register FIFO_i_CH_x_IRQ_NOTIFY (x:0...7)
GTM_FIFO0_CH_x_IRQ_NOTIFY (x=0-7)
FIFO0 Channel x Interrupt Notification Register
(18424_H+x*40_H)
Reset Value: 00000005_H


Field	Bits	Type	Description
FIFO_EMPTY	0	rwh	FIFO is empty 0 _B No interrupt occurred. 1 _B FIFO is empty interrupt occurred. Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
FIFO_FULL	1	rwh	FIFO is full See bit 0
FIFO_LWM	2	rwh	FIFO Lower watermark was under-run See bit 0
FIFO_UWM	3	rwh	FIFO Upper watermark was over-run See bit 0
Reserved	[31:4]	r	Reserved Read as zero, should be written as zero

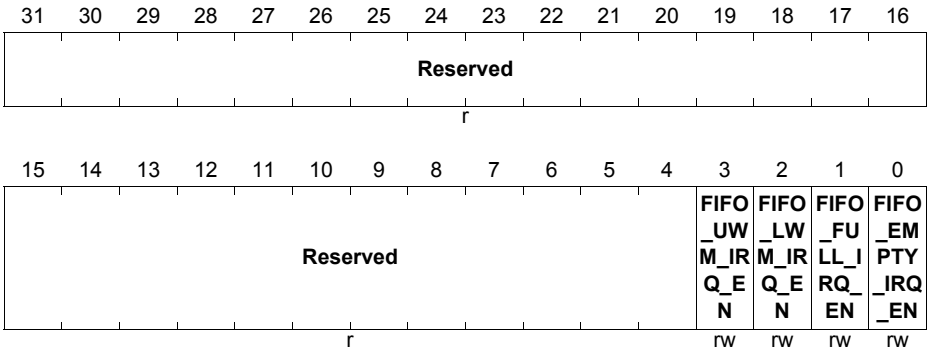
26.5.5.11 Register FIFO_i_CH_x_IRQ_EN (x:0...7)

GTM_FIFO0_CH_x_IRQ_EN (x=0-7)

FIFO0 Channel x Interrupt Enable Register

(18428_H+x*40_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
FIFO_EMPTY_IRQ_EN	0	rw	interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM. 1 _B Enable interrupt, interrupt is visible outside GTM.
FIFO_FULL_IRQ_EN	1	rw	interrupt enable See bit 0
FIFO_LWM_IRQ_EN	2	rw	interrupt enable See bit 0
FIFO_UWM_IRQ_EN	3	rw	interrupt enable See bit 0
Reserved	[31:4]	r	Reserved Read as zero, should be written as zero

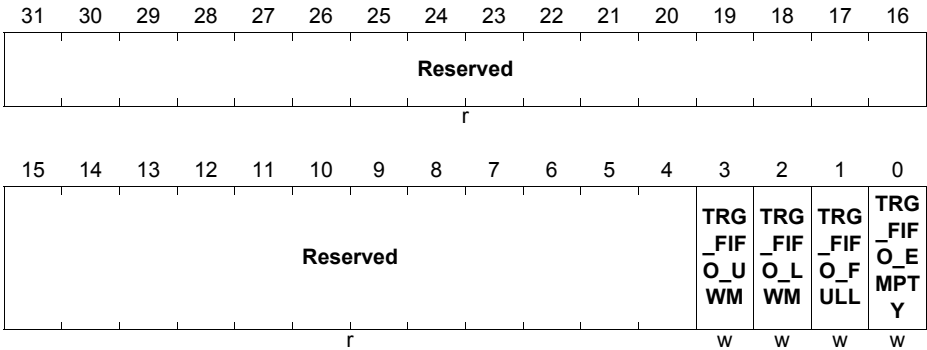
26.5.5.12 Register FIFOi_CHx_IRQ_FORCINT

GTM_FIFO0_CHx_IRQ_FORCINT (x=0-7)

FIFO0 Channel x Force Interrupt By Software Register

(1842C_H+x*40_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
TRG_FIFO_EMPTY	0	w	Force interrupt of FIFO empty status 0 _B corresponding bit in status register will not be forced 1 _B Assert corresponding field in FIFOi_CHi_IRQ_NOTIFY register Note: This bit is cleared automatically after write.
TRG_FIFO_FULL	1	w	Force interrupt of FIFO full status See bit 0
TRG_FIFO_LWM	2	w	Force interrupt of lower watermark See bit 0
TRG_FIFO_UWM	3	w	Force interrupt of upper watermark See bit 0
Reserved	[31:4]	r	Reserved Read as zero, should be written as zero

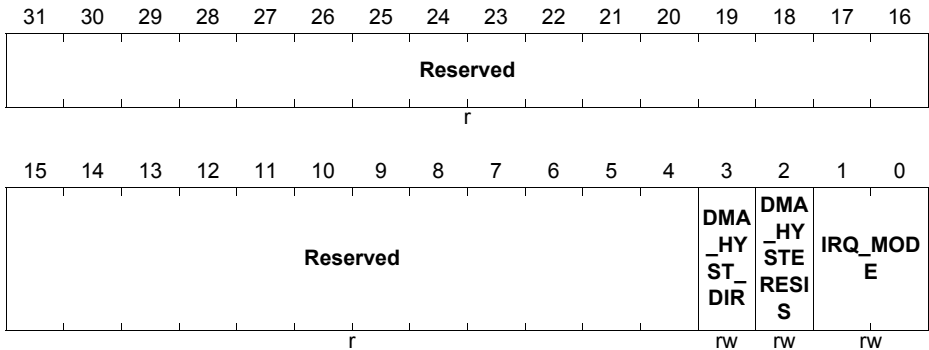
26.5.5.13 Register FIFO_i_CH_x_IRQ_MODE

GTM_FIFO0_CH_x_IRQ_MODE (x=0-7)

FIFO0 Channel x IRQ Mode Control Register

(18430_H+x*40_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
IRQ_MODE	[1:0]	rw	IRQ mode selection 00 _B Level mode 01 _B Pulse mode 10 _B Pulse-Notify mode 11 _B Single-Pulse mode Note: The interrupt modes are described in Section 26.2.5 .
DMA_HYSTERESIS	2	rw	Enable DMA hysteresis mode 0 _B Disable FIFO hysteresis for DMA access. 1 _B Enable FIFO hysteresis for DMA access.

Generic Timer Module (GTM)

Field	Bits	Type	Description
DMA_HYS T_DIR	3	rw	<p>DMA direction in hysteresis mode</p> <p>0_B DMA direction read in hysteresis mode. 1_B DMA direction write in hysteresis mode.</p> <p>Note: In the case of DMA writing data to a FIFO the DMA requests must be generated by the lower watermark. If the DMA hysteresis is enabled, the FIFO does not generate a new DMA request until the upper watermark is reached.</p> <p>Note: In the case of DMA reading data from FIFO the DMA requests must be generated by the upper watermark. If the DMA hysteresis is enabled, the FIFO does not generate a new DMA request until the lower watermark is reached.</p>
Reserved	[31:4]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>

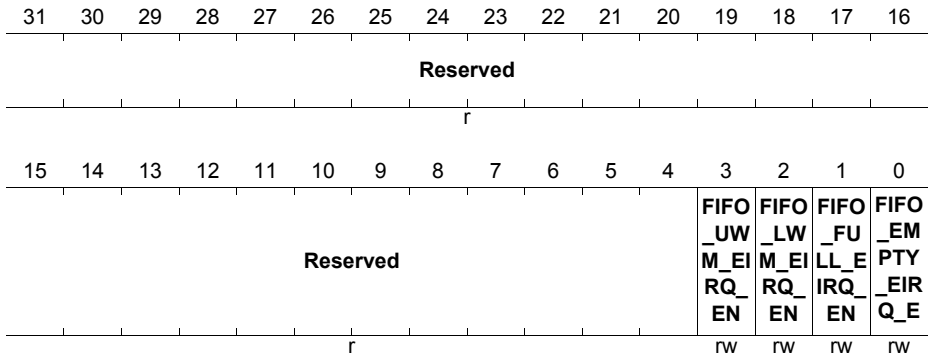
Generic Timer Module (GTM)

26.5.5.14 Register FIFOi_CHx_EIRQ_EN (x:0...7)

GTM_FIFO0_CHx_EIRQ_EN (x=0-7)

FIFO0 Channel x Error Interrupt Enable Register
(18434_H+x*40_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
FIFO_EMPTY_EIRQ_EN	0	rw	Error interrupt enable 0 _B Disable error interrupt, error interrupt is not visible outside GTM 1 _B Enable error interrupt, error interrupt is visible outside GTM
FIFO_FULL_EIRQ_EN	1	rw	Interrupt enable See bit 0
FIFO_LWM_EIRQ_EN	2	rw	Interrupt enable See bit 0
FIFO_UWM_EIRQ_EN	3	rw	Interrupt enable See bit 0
Reserved	[31:4]	r	reserved Note: read as zero, should be written as zero

26.6 AEI to FIFO Data Interface (AFD)

26.6.1 Overview

The AFD submodule implements a data interface between the AEI bus and the FIFO submodule, which consists of eight logical FIFO channels.

The AFD submodule provides one buffer registers that are dedicated to the logical channels of the FIFO. Access to the corresponding FIFO channel is given by reading or writing this buffer registers AFD0_CHx_BUF_ACC.

An AEI write access to the buffer register where the corresponding FIFO channel is full will be ignored. The data will be lost.

An AEI read access to the buffer register where the corresponding FIFO channel is empty will be served with zero data.

26.6.2 AFD Register overview

Following table shows a conclusion of configuration registers address offsets and initial values.

Table 26-15 AFD Register overview

Register Name	Description	Details in Section
AFD0_CHx_BUF_ACC	AFD FIFO x buffer access register (x: 0...7)	Section 26.6.3.1

26.6.3 AFD Register description

All of the following registers are 32-bit only accessible.

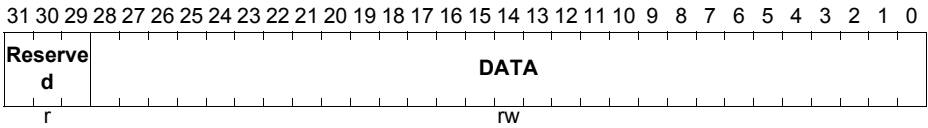
26.6.3.1 Register AFD0_CHx_BUF_ACC (x:0...7)

GTM_AFD0_CHx_BUF_ACC (x=0-7)

AFD0 FIFO0 Channel x Buffer Access Register

(18080_H+x*10_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
DATA	[28:0]	rw	Read/write data from/to FIFO
Reserved	[31:29]	r	Reserved Read as zero, should be written as zero

26.7 FIFO to ARU Unit (F2A)

26.7.1 Overview

The F2A is the interface between the ARU and the FIFO submodule. Since the data width of the ARU (ARU word) is 53 bit (two 24 bit values and five control bits) and the data width of the FIFO is only 29 bit, the F2A has to distribute the data from and to the FIFO channels in a configurable manner.

The data transfer between FIFO and ARU is organized with eight different streams that are connected to the eight different channels of the corresponding FIFO module. A stream represents a data flow from/to ARU to/from the FIFO via the F2A.

The general definition of 'channels' and 'streams' in the ARU context is done in [Section 26.2.3](#).

Each FIFO channel can act as a write stream (data flow from FIFO to ARU) or as a read stream (data flow from ARU to FIFO).

Within these streams the F2A can transmit/receive the lower, the upper or both 24 bit values of the ARU together with the ARU control bits according to the configured transfer modes as described in [Section 26.7.2](#)

26.7.2 Transfer modes

The F2A unit provides several transfer modes to map 29 bit data of the FIFO from/to 53 bit data of the ARU. E.g. it is configurable that the 24 bit FIFO data is written to the lower ARU data entry (means bits 0 to 23) or to the higher 24 bit ARU data entry (means bits 24 to 47). Bits 24 to 28 of the FIFO data entry (the five control bits) are written/read in both cases to/from bits 48 to 52 of the ARU entry.

When both values of the ARU have to be stored in the FIFO the values are stored behind each other inside the FIFO if the FIFO is not full.

If there is only space for one 24 bit data word plus the five control bits, the F2A transfers one part of the 53 bits first and then waits for transferring the second part before new data is requested from the ARU.

When two values from the FIFO have to be written to one ARU location the words have to be located behind each other inside the FIFO.

The transfer to ARU is only established when both parts could be read out of the FIFO otherwise if only one 29 bit word was provided by the FIFO the F2A waits until the second part is available before the data is made available at the ARU.

[Figure 26-16](#) shows the data ordering of the FIFO when both ARU values must be transferred between ARU and FIFO.

When reading from the ARU the F2A first writes the lower word to the FIFO.

Generic Timer Module (GTM)

In case of writing to the ARU the F2A reads the lower word first from the FIFO, thus the lower word must be written first to the FIFO through the AFD interface.

Please note, that the five control bits (bits 48 to 52 of the ARU data word) are duplicated as bit 24 to 28 of both FIFO words in case of reading from ARU.

In the case of writing to the ARU, bits 24 to 28 of the last written FIFO word (the higher ARU word) are copied to bits 48 to 52 of the corresponding ARU location.

The transfer modes can be configured with the TMODE bits of registers F2A0_CHx_STR_CFG (x: 0...7).

26.7.2.1 Data transfer of both ARU words between ARU and FIFO

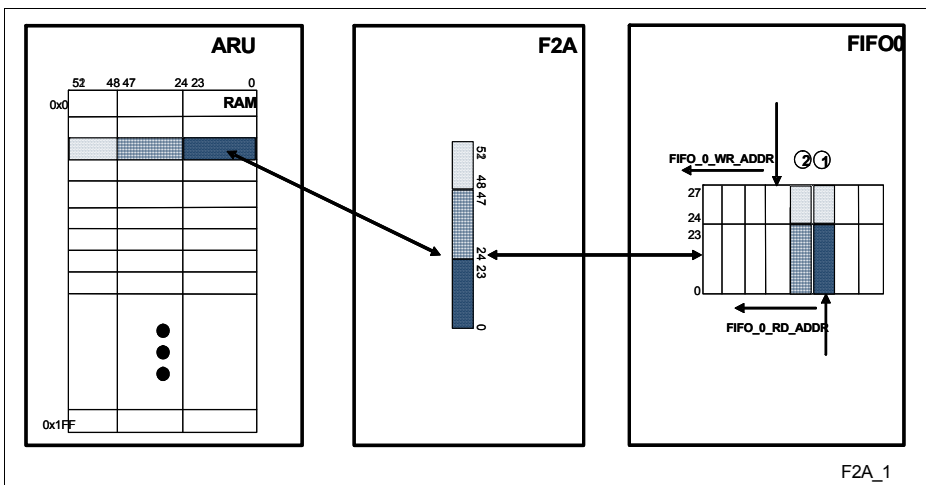


Figure 26-16 Data transfer of both ARU words between ARU and FIFO

26.7.3 F2A Configuration Registers Overview

The following table shows a conclusion of configuration registers address offsets and initial values.

Table 26-16 F2A Configuration Registers Overview

Register name	Description	Details in Section
F2Ai_ENABLE	F2A stream activation register	Section 26.7.4.1
F2Ai_CHx_ARU_RD_FIFO	F2A read channel address register (x: 0...7)	Section 26.7.4.2
F2Ai_CHx_STR_CFG	F2A stream x configuration register (x: 0...7)	Section 26.7.4.3

26.7.4 F2A Configuration Registers description

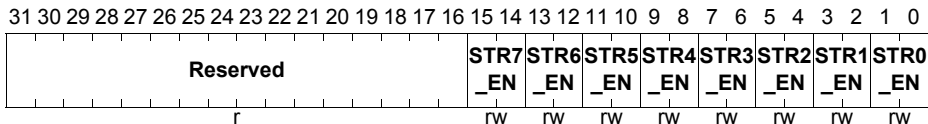
All of the following registers are 32-bit only accessible.

26.7.4.1 Register F2Ai_ENABLE

GTM_F2A0_ENABLE

F2A0 Stream Activation Register (18040_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
STR0_EN	[1:0]	rw	<p>Enable/disable stream 0</p> <p>Write of following double bit values is possible:</p> <p>00_B Don't care, bits 1:0 will not be changed</p> <p>01_B Stream 0 is disabled and internal states are reset</p> <p>10_B Stream 0 is enabled</p> <p>11_B Don't care, bits 1:0 will not be changed</p> <p>Read of following double values means:</p> <p>00_B Stream disabled</p> <p>11_B Stream enabled</p>
STR1_EN	[3:2]	rw	<p>Enable/disable stream 1</p> <p>See bits 1:0</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
STR2_EN	[5:4]	rw	Enable/disable stream 2 See bits 1:0
STR3_EN	[7:6]	rw	Enable/disable stream 3 See bits 1:0
STR4_EN	[9:8]	rw	Enable/disable stream 4 See bits 1:0
STR5_EN	[11:10]	rw	Enable/disable stream 5 See bits 1:0
STR6_EN	[13:12]	rw	Enable/disable stream 6 See bits 1:0
STR7_EN	[15:14]	rw	Enable/disable stream 7 See bits 1:0
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

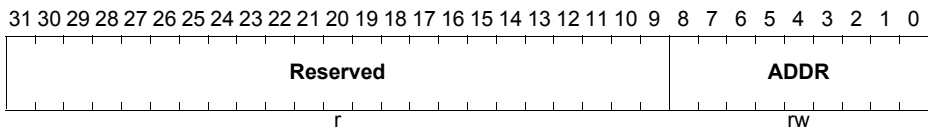
26.7.4.2 Register F2Ai_CHx_ARU_RD_FIFO (x: 0...7)

GTM_F2A0_CHx_ARU_RD_FIFO (x=0-7)

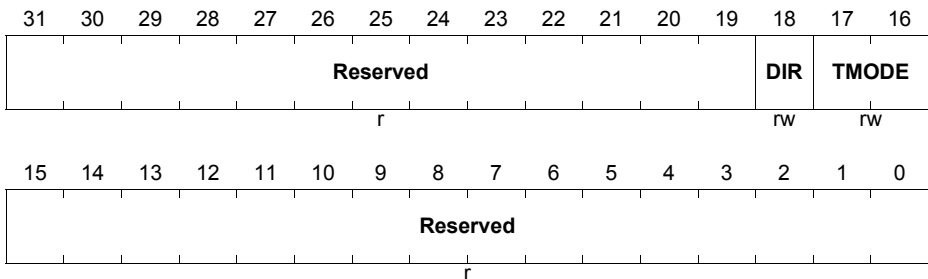
F2A Read Channel Address Register

(18000_H+x*04_H)

Reset Value: 000001FE_H



Field	Bits	Type	Description
ADDR	[8:0]	rw	ARU Read address
Reserved	[31:9]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)
26.7.4.3 Register F2Ai_CHx_STR_CFG (x: 0...7)
GTM_F2A0_CHx_STR_CFG (x=0-7)
F2A Stream x Configuration Register
 $(18020_H + x \cdot 04_H)$
Reset Value: 00000000_H


Field	Bits	Type	Description
Reserved	[15:0]	r	Reserved Read as zero, should be written as zero
TMODE	[17:16]	rw	Transfer mode for 53 bit ARU data from/to FIFO 00 _B Transfer low word (ARU bits 23:0) from/to FIFO 01 _B Transfer high word (ARU bits 47:24) from/to FIFO 10 _B Transfer both words from/to FIFO 11 _B Reserved Note: This bit is only writable if the corresponding enable bit STRx_EN of register F2A_ENABLE is cleared.
DIR	18	rw	Data transfer direction 0 _B Transport from ARU to FIFO 1 _B Transport from FIFO to ARU Note: This bit is only writable if the corresponding enable bit STRx_EN of register F2A_ENABLE is cleared.
Reserved	[31:19]	r	Reserved Read as zero, should be written as zero

26.8 Clock Management Unit (CMU)

26.8.1 Overview

The Clock Management Unit (CMU) is responsible for clock generation of the counters and of the GTM. The CMU consists of three subunits that generate different clock sources for the whole GTM. [Figure 26-17](#) shows a block diagram of the CMU.

The Configurable Clock Generation (CFGU) subunit provides eight dedicated clock sources for the following GTM submodules: TIM, ATOM, TBU, and MON. Each instance of such a submodule can choose an arbitrary clock source, in order to specify wide-ranging time bases.

The Fixed Clock Generation (FXU) subunit generates predefined non-configurable clocks `CMU_FXCLK[y]` ($y: 0 \dots 4$) for the TOM submodules and the MON submodule. The `CMU_FXCLK[y]` signals are derived from the `CMU_GCLK_EN` signal generated by the Global Clock Divider. The dividing factors are defined as 2^0 , 2^4 , 2^8 , 2^{12} , and 2^{16} .

The External Clock Generation (EGU) subunit is able to generate up to three GTM external clock signals visible at `CMU_ECLK[z]` ($z: 0 \dots 2$) with a duty cycle of about 50%.

The clock source signals `CMU_CLK[x]` ($x: 0 \dots 7$) and `CMU_FXCLK[y]` are implemented in form of enable signals for the corresponding registers, which means that the actual clock signal of all registers always use the `SYS_CLK` signal.

The four configurable clock signals `CMU_CLK0`, `CMU_CLK1`, `CMU_CLK6` and `CMU_CLK7` are connected to the TIM filter counters.

26.8.1.1 CMU Block Diagram

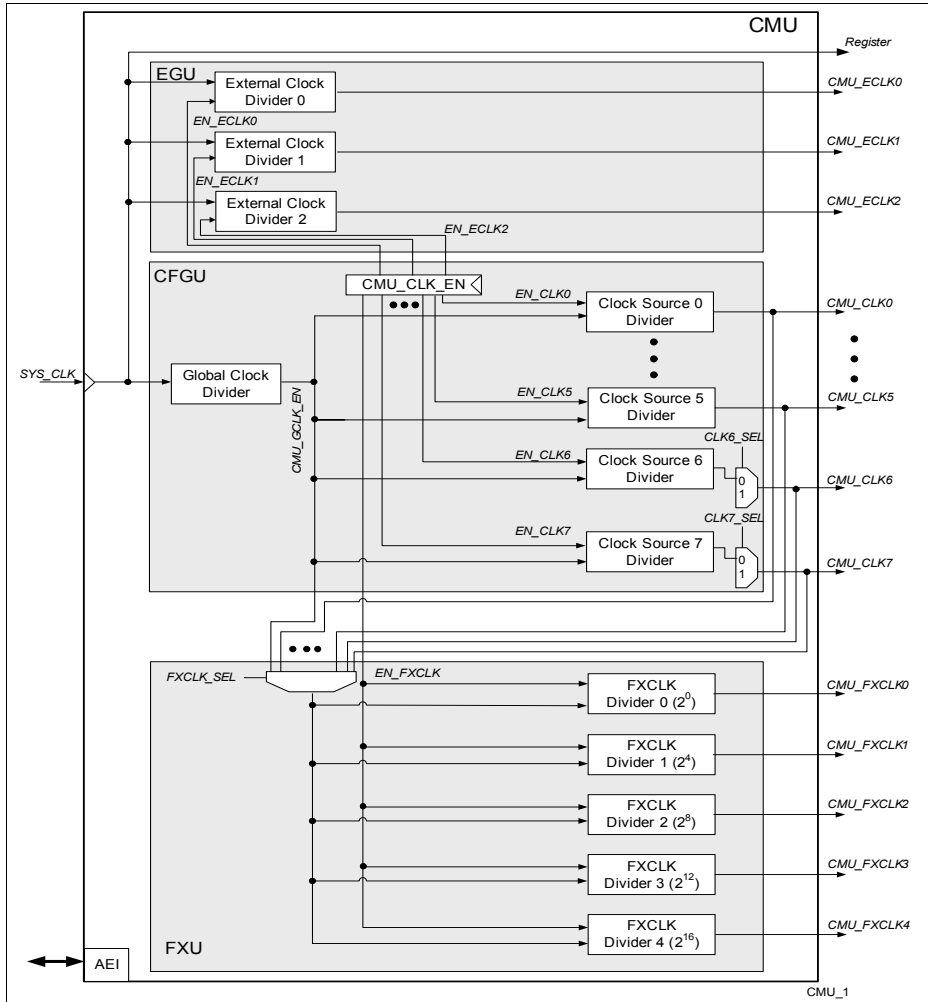


Figure 26-17 CMU Block Diagram

26.8.2 Global Clock Divider

The subblock Global Clock Divider can be used to divide the GTM global input clock signal SYS_CLK into a common subdivided clock signal.

The divided clock signal of the subblock Global Clock Divider is implemented as an enable signal that enables dedicated clocks from the SYS_CLK signal to generate the user specified divided clock frequency.

The resulting fractional divider (Z/N) specified through equation:

$$\text{TCMU_GCLK_EN} = (\text{Z}/\text{N}) * \text{TSYS_CLK}$$

is implemented according the following algorithm

(Z: CMU_GCLK_NUM(23:0); N: CMU_GCLK_DEN(23:0); Z,N >0):

(1) Set remainder (r), operand1 (OP1) and operand2 (OP2) register during init-phase (with implicit conversion to signed):

$$r = Z, \text{OP1} = N, \text{OP2} = N - Z;$$

(2) After leaving init-phase (at least one CMU_CLK[x] has been enabled) the sign of remainder r for each SYS_CLK cycle will be checked:

(3) If $r > 0$ keep updating remainder and keep CMU_GCLK_EN='0':

$$r = r - \text{OP1};$$

(4) If $r < 0$ update remainder and set CMU_GCLK_EN='1':

$$r = r - \text{OP2};$$

After at most $(Z/N+1)$ subtractions (3) there will be a negative r and an active phase of the generated clock enable (for one cycle) will be triggered (4). The remainder r is a measure for the distance to a real Z/N clock and will be regarded for the next generated clock enable cycle phase. The new r value will be $r = r + (Z - N)$. In the worst case the remainder r will sum up to an additional cycle in the generated clock enable period after Z-cycles. In the other cases equally distributed additional cycles will be inserted for the generated clock enable. If Z is an integer multiple of N no additional cycles will be included for the generated clock enable at all.

Note that for a better resource sharing all arithmetic has been reduced to subtractions and the initialization of the remainder r uses the complement of $(Z - N)$.

26.8.3 Configurable Clock Generation Subunit (CFGU)

The CMU subunit CFGU provides up to eight configurable clock divider blocks that divide the common CMU_GCLK_EN signal into dedicated enable signals for the GTM subblocks.

The configuration of the eight different clock signals CMU_CLK[x] (x: 0...7) always depends on the configuration of the global clock enable signal CMU_GCLK_EN. Additionally, each clock source has its own configuration data, provided by the control register CMU_CLK_x_CTRL (x: 0...7).

Generic Timer Module (GTM)

According to the configuration of the Global Clock Divider, the configuration of the Clock Source x Divider is done by setting an appropriate value in the bit field CLK_CNT[x] of the register CMU_CLK_x_CTRL.

The frequency $f_x = 1/T_x$ of the corresponding clock enable signal CMU_CLK[x] can be determined by the unsigned representation of CLK_CNT[x] of the register CMU_CLK_x_CTRL in the following way:

$$TCMU_CLK[x] = (CLK_CNT[x] + 1) * TCMU_GCLK_EN$$

The corresponding wave form is shown in [Figure 26-18](#).

Each clock signal CMU_CLK[x] can be enabled individually by setting the appropriate bit field EN_CLK[x] in the register CMU_CLK_EN. Except for CMU_CLK6 and CMU_CLK7 individual enabling and disabling is active only if CLK6_SEL and CLK7_SEL is cleared.

Alternatively, clock source six and seven (CMU_CLK6 and CMU_CLK7) may provide the signal SUB_INC1 and SUB_INC2 coming from submodule DPPLL as clock enable signal depending on the bit field CLK6_SEL of the register CMU_CLK_6_CTRL and on the bit field CLK7_SEL of the register CMU_CLK_7_CTRL.

To avoid unexpected behaviour of the hardware, the configuration of a register CMU_CLK_x_CTRL can only be changed, when the corresponding clock signal CMU_CLK[x] is disabled.

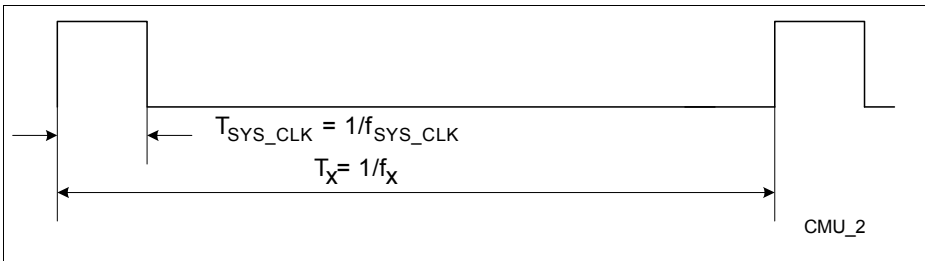
Further, any changes to the registers CMU_GCLK_NUM and CMU_GCLK_DEN can only be performed, when all clock enable signals CMU_CLK[x] and the EN_FXCLK bit inside the CMU_CLK_EN register are disabled.

The clock source signals CMU_CLK[x] (x:0...7) and CMU_FXCLK[y] are implemented in form of enable signals for the corresponding registers, which means that the actual clock signal of all registers always use the SYS_CLK signal.

The hardware guarantees that all clock signals CMU_CLK[x], which were enabled simultaneous, are synchronized to each other. Simultaneous enabling does mean that the bits EN_CLK[x] in the register CMU_CLK_EN are set by the same write access.

After EN_CLK[x] entries in CMU_CLK_EN have been disabled, internal states of these EN_CLK[x] paths will be reset to start with the same behaviour after enable.

26.8.4 Wave Form of Generated Clock Signal CMU_CLK[x]


Figure 26-18 Example clock signal

26.8.5 Fixed Clock Generation (FXU)

The FXU subunit generates fixed clock enables out of the CMU_GCLK_EN or one of the eight CMU_CLKx enable signal depending on the CMU_CLK_5.FXCLK_SEL bit field of the CMU_FXCLK_CTRL register. These clock enables are used for the PWM generation inside the TOM submodules.

All clock enables CMU_FXCLK[y] can be enabled or disabled simultaneously by setting the appropriate bit field EN_FXCLK in the register CMU_CLK_EN.

The dividing factors are defined as 2^0 , 2^4 , 2^8 , 2^{12} , and 2^{16} . The signals CMU_FXCLK[y] are implemented in form of enable signals for the corresponding registers (see also [Section 26.8.4](#))

After EN_FXCLK entry in CMU_CLK_EN have been disabled, internal states of EN_FXCLK paths will be reset to start with the same behaviour after enable.

26.8.6 External Generation Unit (EGU)

The EGU subunit generate up to three separate clock output signals CMU_ECLK[z] (z: 0...2).

Each of these clock signals is derived from the corresponding External Clock Divider z subblock, which generates a clock signal derived from the GTM input clock SYS_CLK.

In contrast to the signals CMU_CLK[x] and CMU_FXCLK[y], which are treated as simple enable signals for the registers, the signals CMU_ECLK[z] have a duty cycle of about 50% that is used as a true clock signal for external peripheral components.

Each of the external clocks are enabled and disabled by setting the appropriate bit field EN_ECLK[z] in the register CMU_CLK_EN.

After EN_ECLK[z] entries in CMU_CLK_EN have been disabled, internal states of these EN_ECLK[z] paths will be reset to start with the same behaviour after enable.

The clock frequencies $f_{\text{CMU_ECLK}[z]} = 1/T_{\text{CMU_ECLK}[z]}$ of the external clocks are controlled with the registers CMU_ECLK_z_NUM and CMU_ECLK_z_DEN as follows:

$$T_{\text{CMU_ECLK}[z]} = 2 * (\text{ECLK}[z]_{\text{NUM}} / \text{ECLK}[z]_{\text{DEN}}) * T_{\text{SYS_CLK}}$$

Generic Timer Module (GTM)

and is implemented according the following algorithm

(Z: CMU_ECLK_z_NUM(23:0); N: CMU_ECLK_z_DEN(23:0); $Z, N > 0$; $Z \geq N$; CMU_ECLK[z]='0');

(1) Set remainder (r), operand1 (OP1) and operand2 (OP2) register during init-phase (with implicit conversion to signed):

$r=Z$, $OP1=N$, $OP2=N-Z$;

(2) After leaving init-phase (CMU_ECLK[z] has been enabled) the sign of remainder r for each SYS_CLK cycle will be checked:

(3) If $r > 0$ keep updating remainder and keep CMU_ECLK[z]:

$r=r-OP1$;

(4) If $r < 0$ update remainder and toggle CMU_ECLK[z]:

$r=r-OP2$;

After at most $(Z/N+1)$ subtractions (3) there will be a negative r and an active phase of the generated clock enable (for one cycle) will be triggered (4). The remainder r is a measure for the distance to a real Z/N clock and will be regarded for the next generated clock toggle phase. The new r value will be $r=r+(Z-N)$. In the worst case the remainder r will sum up to an additional cycle in the generated clock toggle period after Z-cycles. In the other cases equally distributed additional cycles will be inserted for the generated clock toggle. If Z is an integer multiple of N no additional cycles will be included for the generated clock toggle at all.

Note that for a better resource sharing all arithmetic has been reduced to subtractions and the initialization of the remainder r uses the complement of $(Z-N)$.

The default value of the CMU_ECLK[z] output is low.

26.8.7 CMU Configuration Registers Overview

Following configuration registers are considered in CMU submodule:

Table 26-17 CMU Configuration Registers Overview

Register Name	Description	Details in Section
CMU_CLK_EN	CMU Clock enable register	Section 26.8.8.1
CMU_GCLK_NUM	CMU Global clock control numerator register	Section 26.8.8.2
CMU_GCLK_DEN	CMU Global clock control denominator register	Section 26.8.8.3
CMU_CLK_x_CTRL	CMU Control for clock source x register (x:0...5)	Section 26.8.8.4
CMU_CLK_6_CTRL	CMU Control for clock source 6 register	Section 26.8.8.5

Generic Timer Module (GTM)

Table 26-17 CMU Configuration Registers Overview (cont'd)

Register Name	Description	Details in Section
CMU_CLK_7_CTRL	CMU Control for clock source 7 register	Section 26.8.8.6
CMU_ECLK_z_NUM	CMU External clock z control numerator register (x:0...2)	Section 26.8.8.7
CMU_ECLK_z_DEN	CMU External clock z control denominator register (z:0...2)	Section 26.8.8.8
CMU_FXCLK_CTRL	Control FXCLK subunit input clock	Section 26.8.8.9

26.8.8 CMU Configuration Register Description

All of the following registers are 32-bit only accessible.

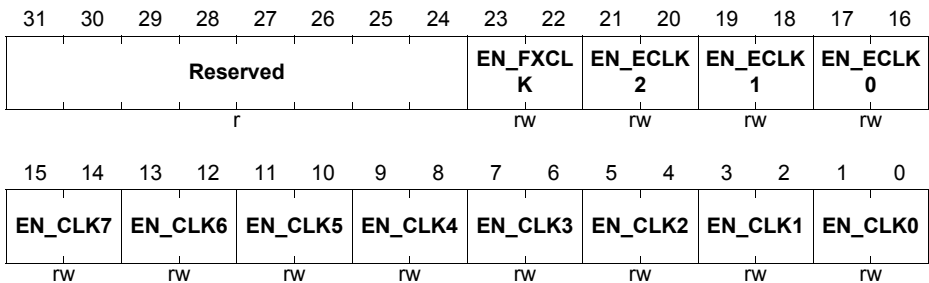
26.8.8.1 Register CMU_CLK_EN

GTM_CMU_CLK_EN

CMU Clock Enable Register

(00300_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
EN_CLK0	[1:0]	rw	<p>Enable clock source 0</p> <p>00_B clock source is disabled (ignore write access) 01_B disable clock signal and reset internal states 10_B enable clock signal 11_B clock signal enabled (ignore write access)</p> <p><i>Note: Any read access to an EN_CLKx, EN_ECLKz or EN_FXCLK bit field will always result in a value 00 or 11 indicating current state. A modification of the state is only performed with the values 01 and 10. Writing the values 00 and 11 is always ignored.</i></p> <p><i>Note: Any disabling to EN_CLKx will be reset internal counters for configurable clocks.</i></p>
EN_CLK1	[3:2]	rw	<p>Enable clock source 1</p> <p>see bits [1:0]</p>
EN_CLK2	[5:4]	rw	<p>Enable clock source 2</p> <p>see bits [1:0]</p>
EN_CLK3	[7:6]	rw	<p>Enable clock source 3</p> <p>see bits [1:0]</p>

Generic Timer Module (GTM)

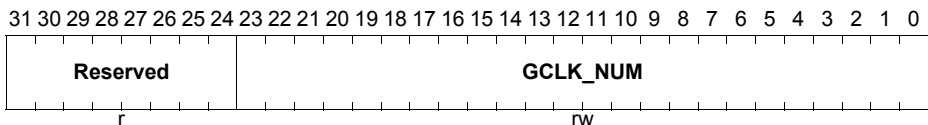
Field	Bits	Type	Description
EN_CLK4	[9:8]	rw	Enable clock source 4 see bits [1:0]
EN_CLK5	[11:10]	rw	Enable clock source 5 see bits [1:0]
EN_CLK6	[13:12]	rw	Enable clock source 6 see bits [1:0]
EN_CLK7	[15:14]	rw	Enable clock source 7 see bits [1:0]
EN_ECLK 0	[17:16]	rw	Enable ECLK 0 generation subunit see bits [1:0]
EN_ECLK 1	[19:18]	rw	Enable ECLK 1 generation subunit see bits [1:0]
EN_ECLK 2	[21:20]	rw	Enable ECLK 2 generation subunit see bits [1:0]
EN_FXCLK K	[23:22]	rw	Enable all CMU_FXCLK see bits [1:0] <i>Note: An enable to EN_FXCLK from disable state will be reset internal fixed clock counters.</i>
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

26.8.8.2 Register CMU_GCLK_NUM

GTM_CMU_GCLK_NUM

CMU Global Clock Control Numerator Register

 (00304_H)

 Reset Value: 00000001_H


Generic Timer Module (GTM)

Field	Bits	Type	Description
GCLK_NUM	[23:0]	rw	<p>Numerator for global clock divider Defines numerator of the fractional divider.</p> <p><i>Note: Value can only be modified when all clock enables EN_CLK[x] and the EN_FXCLK are disabled.</i></p> <p><i>Note: The CMU hardware alters the content of CMU_GCLK_NUM and CMU_GCLK_DEN automatically to 0x1, if CMU_GCLK_NUM is specified less than CMU_GCLK_DEN or one of the values is specified with a value zero. Thus, a secure way for altering the values is writing twice to the register CMU_GCLK_NUM followed by a single write to register CMU_GCLK_DEN.</i></p>
Reserved	[31:24]	r	<p>Reserved Read as zero, should be written as zero</p>

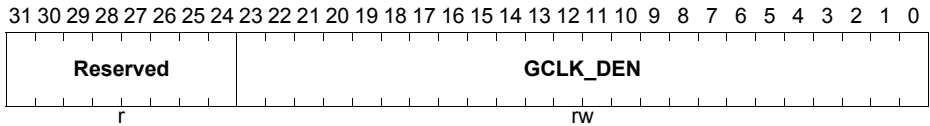
Generic Timer Module (GTM)

26.8.8.3 Register CMU_GCLK_DEN

GTM_CMU_GCLK_DEN

CMU Global Clock Control Denominator Register
(00308_H)

Reset Value: 00000001_H



Field	Bits	Type	Description
GCLK_DEN	[23:0]	rw	<p>Denominator for global clock divider Defines denominator of the fractional divider.</p> <p><i>Note: Value can only be modified when all clock enables EN_CLK[x] and the EN_FXCLK are disabled.</i></p> <p><i>Note: The CMU hardware alters the content of CMU_GCLK_NUM and CMU_GCLK_DEN automatically to 0x1, if CMU_GCLK_NUM is specified less than CMU_GCLK_DEN or one of the values is specified with a value zero. Thus, a secure way for altering the values is writing twice to the register CMU_GCLK_NUM followed by a single write to register CMU_GCLK_DEN.</i></p>
Reserved	[31:24]	r	<p>Reserved Read as zero, should be written as zero</p>

Generic Timer Module (GTM)

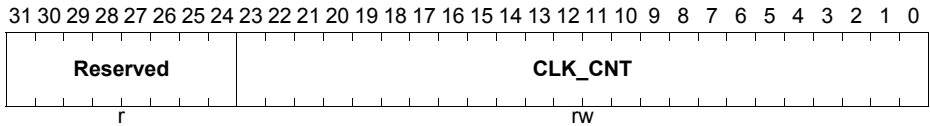
26.8.8.4 Register CMU_CLK_x_CTRL (x:0...5)

GTM_CMU_CLK_x_CTRL (x=0-5)

CMU Control For Clock Source x Register

(0030C_H+x*04_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CLK_CNT	[23:0]	rw	Clock count Defines count value for the clock divider of clock source CMU_CLK[x] (x:0...5) Note: Value can only be modified when clock enable EN_CLKx (x:0...5) is disabled.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

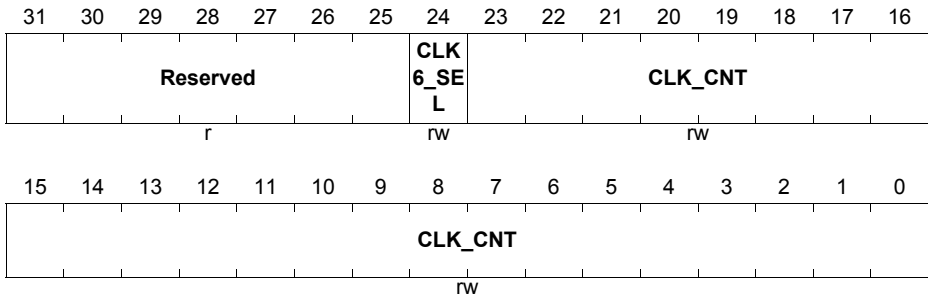
26.8.8.5 Register CMU_CLK_6_CTRL

GTM_CMU_CLK_6_CTRL

CMU Control For Clock Source 6 Register

(00324_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CLK_CNT	[23:0]	rw	Clock count. Define count value for the clock divider of clock source CMU_CLK6 Note: Value can only be modified when clock enable EN_CLK6 is disabled
CLK6_SE L	24	rw	Clock source selection for CMU_CLK6 0 _B use Clock Source 6 Divider 1 _B use signal SUB_INC2 of submodule DPLL Note: Value can only be modified when clock enable EN_CLK6 is disabled.
Reserved	[31:25]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

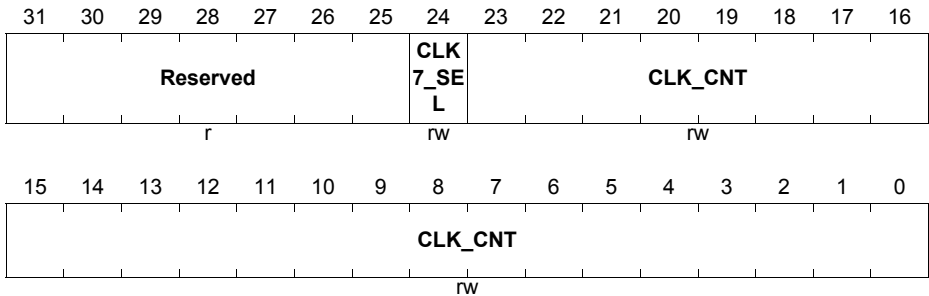
26.8.8.6 Register CMU_CLK_7_CTRL

GTM_CMU_CLK_7_CTRL

CMU Control For Clock Source 7 Register

(00328_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CLK_CNT	[23:0]	rw	Clock count. Define count value for the clock divider of clock source CMU_CLK7 Note: Value can only be modified when clock enable EN_CLK7 is disabled
CLK7_SE L	24	rw	Clock source selection for CMU_CLK7 0 _B use Clock Source 7 Divider 1 _B use signal SUB_INC1 of submodule DPLL Note: Value can only be modified when clock enable EN_CLK7 is disabled.
Reserved	[31:25]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

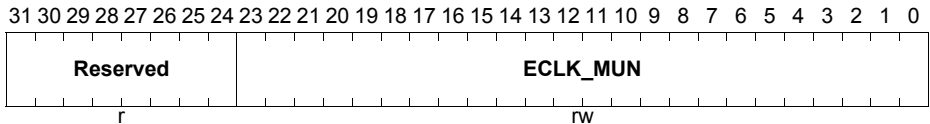
26.8.8.7 Register CMU_ECLK_z_NUM (z:0...2)

GTM_CMU_ECLK_z_NUM (z=0-2)

CMU External Clock z Control Numerator Register

(0032C_H+z*08_H)

Reset Value: 00000001_H



Field	Bits	Type	Description
ECLK_NUM	[23:0]	rw	<p>Numerator for external clock divider Defines numerator of the fractional divider.</p> <p><i>Note: Value can only be modified when clock enable EN_ECLK[z] is disabled.</i></p> <p><i>Note: The CMU hardware alters the content of CMU_ECLK_z_NUM and CMU_ECLK_z_DEN automatically to 0x1, if CMU_ECLK_z_NUM is specified less than CMU_ECLK_z_DEN or one of the values is specified with a value zero. Thus, a secure way for altering the values is writing twice to the register CMU_ECLK_z_NUM followed by a single write to register CMU_ECLK_z_DEN.</i></p>
Reserved	[31:24]	r	<p>Reserved Read as zero, should be written as zero</p>

Generic Timer Module (GTM)

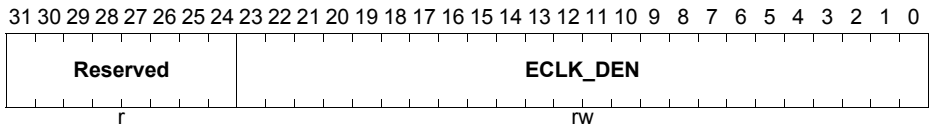
26.8.8.8 Register CMU_ECLK_z_DEN (z:0...2)

GTM_CMU_ECLK_z_DEN (z=0-2)

CMU External Clock z Control Denominator Register

(00330_H + z*08_H)

Reset Value: 00000001_H

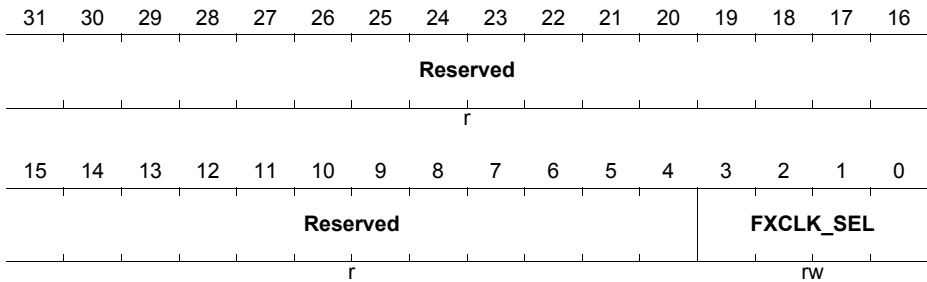


Field	Bits	Type	Description
ECLK_DEN	[23:0]	rw	<p>Denominator for external clock divider Defines denominator of the fractional divider.</p> <p><i>Note: Value can only be modified when clock enable EN_ECLK[z] is disabled.</i></p> <p><i>Note: The CMU hardware alters the content of CMU_ECLK_z_NUM and CMU_ECLK_z_DEN automatically to 0x1, if CMU_ECLK_z_NUM is specified less than CMU_ECLK_z_DEN or one of the values is specified with a value zero. Thus, a secure way for altering the values is writing twice to the register CMU_ECLK_z_NUM followed by a single write to register CMU_ECLK_z_DEN.</i></p>
Reserved	[31:24]	r	<p>Reserved Read as zero, should be written as zero</p>

Generic Timer Module (GTM)

26.8.8.9 Register CMU_FXCLK_CTRL

GTM_CMU_FXCLK_CTRL
 CMU FXCLK Control Register (00344_H) Reset Value: 00000000_H



Field	Bits	Type	Description
FXCLK_SEL	[3:0]	rw	<p>Input clock selection for EN_FXCLK line</p> <p>0000_BCMU_GCLK_EN selected 0001_BCMU_CLK0 selected 0010_BCMU_CLK1 selected 0011_BCMU_CLK2 selected 0100_BCMU_CLK3 selected 0101_BCMU_CLK4 selected 0110_BCMU_CLK5 selected 0111_BCMU_CLK6 selected 1000_BCMU_CLK7 selected</p> <p><i>Note: This value can only be written, when the CMU_FXCLK generation is disabled. See bits 23...22 in register CMU_CLK_EN.</i></p> <p><i>Note: Other values for FXCLK_SEL are reserved and should not be used, but they behave like FXCLK_SEL = 0.</i></p>
Reserved	[31:4]	r	<p>Reserved bits</p> <p>Read as zero, should be written as zero</p>

26.9 Time Base Unit (TBU)

26.9.1 Overview

The Time Base Unit TBU provides common time bases for the GTM. The TBU submodule is organized in channels, where the number of channels is device dependent. There are at most three channels implemented inside the TBU. The TBU channel 0 time base register TBU_CHO_BASE is 27 bits and it is configurable whether the lower 24 bit or the upper 24 bit are provided to the GTM as signal TBU_TS0. The two TBU channels 1 and 2 have a time base register TBU_CHy_BASE (y: 1, 2) of 24 bit length. The time base register value TBU_TS[y] and the time base register update event TBU_UP[y] are provided to subsequent submodules of the GTM.

The TBU_UP[z] (z:1...2) signals are set to high for a single SYS_CLK period, whenever the corresponding signal TBU_TS[z] (z:1...2) or TBU_TS0 is getting updated. The signal TBU_UP0_L is set to high for a single SYS_CLK period if the signal TBU_TS0 and TBU_TS0 is getting updated and TBU_UP0_H is set to high for a single SYS_CLK period, whenever if the upper 24 bit of TBU_TS0 are updated.

The time base channels can run independently of each other and can be enabled and disabled synchronously by control bits in a global TBU channel enable register TBU_CHEN. [Section 26.9.1.1](#) shows a block diagram of the Time Base Unit.

26.9.1.1 TBU Block Diagram

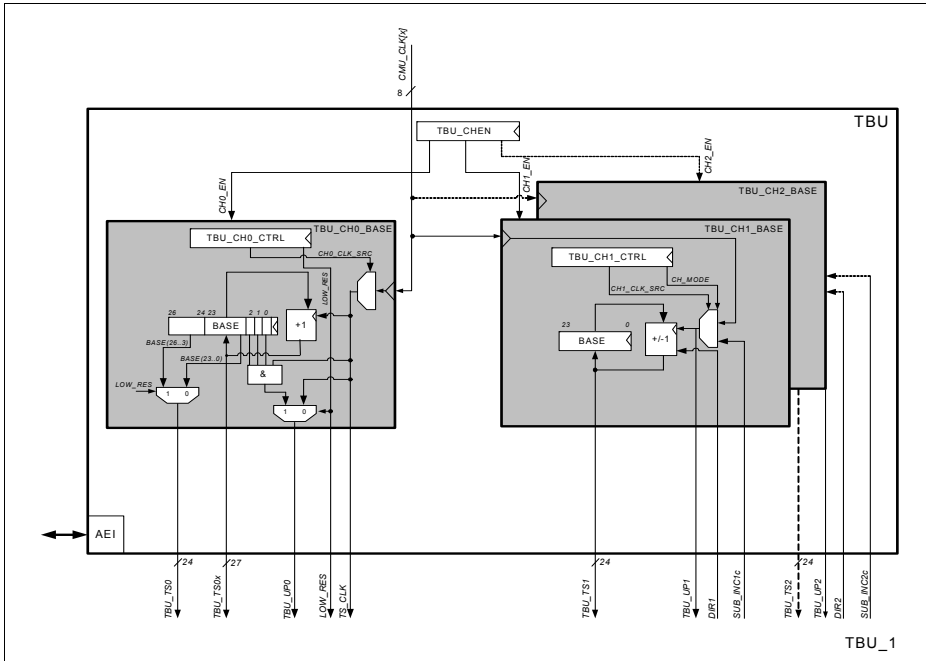


Figure 26-19 TBU Block Diagram

Dependent on the device a third TBU channel exists which offers the same functionality as the time base channel 1.

The configuration of the independent time base channels TBU_BASE_[z] is done via the AEI interface. Each TBU channel may select one of the eight CMU_CLK[x] (x: 0...7) signals coming from the CMU submodule.

For TBU channels 1 and 2 an additional clock signal SUB_INC[y]c (y: 1, 2) coming from the DPLL can be selected as input clock for the TBU_BASE_[y]. This clock in combination with the DIR[y] signals determines the counter direction of the TBU_BASE_[y].

The selected time stamp clock signal for the TBU_BASE_0 subunit is served via the TS_CLK signal line to the DPLL submodule. The TS_CLK signal equals the signal TBU_UP0.

26.9.2 TBU Time Base Channels

The time base values are generated within the TBU time base channels in two independent operation modes.

26.9.2.1 TBU Channel Modes

TBU channel 0 provides a 27 bit counter in a free running counter mode. Dependent on the bit field `LOW_RES` of register `TBU_CH0_CTRL`, the lower 24 bits (bit 0 to 23) or the upper 24 bits (bits 3 to 26) are provided to the GTM submodules.

TBU channel 1 and channel 2 can run in two modes; the free running counter mode and forward/backward counter mode, where the time base can run backwards dependent on the `DIR[y]` input signal values.

In both modes, the time base register `TBU_CH[z]_BASE` can be initialized with a start value just before enabling the corresponding TBU channel.

Moreover, the time base register `TBU_CH[z]_BASE` can always be read in order to determine the actual value of the counter.

Free Running Counter Mode

In TBU Free running counter mode, the time base register `TBU_CHy_BASE` is updated on every specified incoming clock event by the selected signal `CMU_CLK[x]` (dependent on `TBU_CH[z]_CTRL` register). In general the time base register `TBU_CHy_BASE` is increment on every `CMU_CLK[x]` clock tick.

Forward/Backward Counter Mode

As mentioned above TBU channels 1 and 2 can also be configured to run in Forward/Backward Counter Mode. In this mode the `DIR[y]` signal provided by the DPLL is taken into account.

The value of the time base register `TBU_CHy_BASE` is increment in case when the `DIR[y]` signal equals '0' and decremented in case when the `DIR[y]` signal is '1'.

26.9.3 TBU Configuration Registers Overview

Following table shows a conclusion of configuration registers address offsets and initial values.

Table 26-18 TBU Configuration Registers Overview

Register Name	Description	Details in Section
<code>TBU_CHEN</code>	TBU global channel enable register	Section 26.9.4.1
<code>TBU_CH0_CTRL</code>	TBU channel 0 control register	Section 26.9.4.2

Generic Timer Module (GTM)

Table 26-18 TBU Configuration Registers Overview (cont'd)

Register Name	Description	Details in Section
TBU_CH0_BASE	TBU channel 0 base register	Section 26.9.4.3
TBU_CHy_CTRL	TBU channel y control register (y:1,2)	Section 26.9.4.4
TBU_CHy_BASE	TBU channel y base register (y:1,2)	Section 26.9.4.5

Note: In a typical application the Time Base Unit (TBU) considers channels 0 and 1 only. In this case register addresses 0x20...0x2C are reserved and shall be read as zero. Channel 2 can be additionally implemented on special high-end application requirements.

26.9.4 TBU Registers description

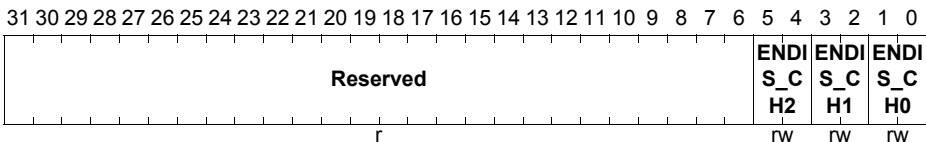
All of the following registers are 32-bit only accessible.

26.9.4.1 Register TBU_CHEN

GTM_TBU_CHEN

TBU Global Channel Enable Register(00100_H)

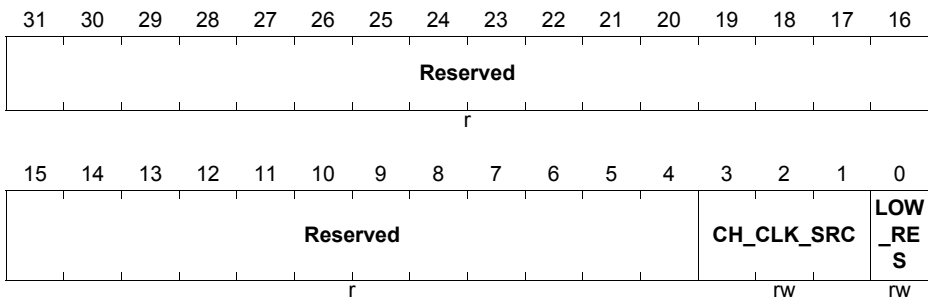
Reset Value: 0000000_H



Field	Bits	Type	Description
ENDIS_CH 0	[1:0]	rw	TBU channel 0 enable/disable control Write of following double bit values is possible: 00 _B don't care, bits 1:0 will not be changed 01 _B channel disabled: is read as 00 (see below) 10 _B channel enabled: is read as 11 (see below) 11 _B don't care, bits 1:0 will not be changed Note: Read of following double values means: 00 _B channel disabled 11 _B channel enabled
ENDIS_CH 1	[3:2]	rw	TBU channel 1 enable/disable control See bits 1:0

Generic Timer Module (GTM)

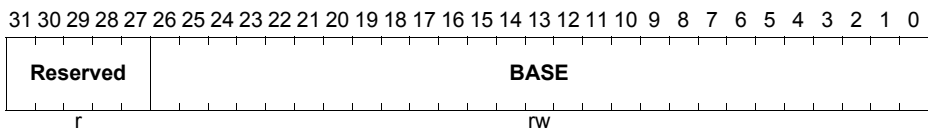
Field	Bits	Type	Description
ENDIS_CH 2	[5:4]	rw	TBU channel 2 enable/disable control See bits 1:0 Note: These bits are only applicable if channel is implemented for this device, otherwise read and write as zero
Reserved	[31:6]	r	Reserved Read as zero, should be written as zero

26.9.4.2 Register TBU_CH0_CTRL
GTM_TBU_CH0_CTRL
TBU Channel 0 Control Register (00104_H) **Reset Value: 00000000_H**


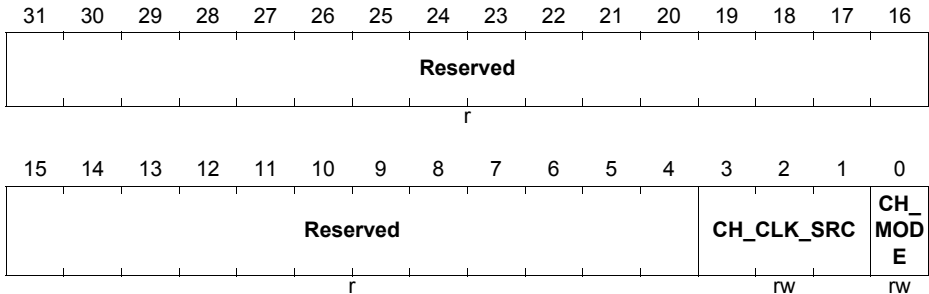
Field	Bits	Type	Description
LOW_RES	0	rw	TBU_CH0_BASE register resolution 0 _B TBU channel uses lower counter bits (bit 0 to 23) 1 _B TBU channel uses upper counter bits (bit 3 to 26) Note: The two resolutions for the TBU channel 0 can be used in the TIM channel 0 and the DPLL submodules. Note: This value can only be modified if channel 0 is disabled.

Generic Timer Module (GTM)

Field	Bits	Type	Description
CH_CLK_SRC	[3:1]	rw	Clock source for channel x (x:0...2) time base counter 000 _B CMU_CLK0 selected 001 _B CMU_CLK1 selected 010 _B CMU_CLK2 selected 011 _B CMU_CLK3 selected 100 _B CMU_CLK4 selected 101 _B CMU_CLK5 selected 110 _B CMU_CLK6 selected 111 _B CMU_CLK7 selected Note: This value can only be modified (written) if channel 0 was disabled
Reserved	[31:4]	r	Reserved Read as zero, should be written as zero

26.9.4.3 Register TBU_CH0_BASE
GTM_TBU_CH0_BASE
TBU Channel 0 Base Register
(00108_H)
Reset Value: 00000000_H


Field	Bits	Type	Description
BASE	[26:0]	rw	Time base value for channel 0 Note: The value of BASE can only be written if the TBU channel 0 is disabled Note: If channel 0 is enabled, a read access to this register provides the current value of the underlying 27 bit counter.
Reserved	[31:27]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)
26.9.4.4 Register TBU_CHy_CTRL (y:1, 2)
GTM_TBU_CHy_CTRL (y=1-2)
TBU Channel y Control Register (00104_H+y*08_H)
Reset Value: 00000000_H


Field	Bits	Type	Description
CH_MODE	0	rw	Channel mode 0 _B Free running counter mode 1 _B Forward/backward counter mode Note: This value can only be modified if channel y (y:1,2) was disabled. In Free running counter mode the CMU clock source specified by CH_CLK_SRC is used for the counter. In Forward/Backward counter mode the SUB_INC[y]c clock signal in combination with the DIR[y] input signal is used to determine the counter direction and clock frequency.
CH_CLK_SRC	[3:1]	rw	Clock source for channel x (x1...2) time base counter 000 _B CMU_CLK0 selected 001 _B CMU_CLK1 selected 010 _B CMU_CLK2 selected 011 _B CMU_CLK3 selected 100 _B CMU_CLK4 selected 101 _B CMU_CLK5 selected 110 _B CMU_CLK6 selected 111 _B CMU_CLK7 selected Note: This value can only be modified if channel y was disabled
Reserved	[31:4]	r	Reserved Read as zero, should be written as zero

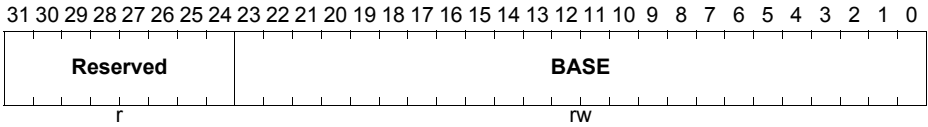
Generic Timer Module (GTM)

26.9.4.5 Register TBU_CHy_BASE (y:1,2)

GTM_TBU_CHy_BASE (y=1-2)

TBU Channel y Base Register (00108_H+y*08_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
BASE	[23:0]	rw	<p>Time base value for channel x (x 1, 2)</p> <p>Note: The value of BASE can only be written if the corresponding TBU channel y is disabled</p> <p>Note: If the corresponding channel y is enabled, a read access to this register provides the current value of the underlying counter.</p>
Reserved	[31:24]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>

26.10 Timer Input Module (TIM)

26.10.1 Overview

The Timer Input Module (TIM) is responsible for filtering and capturing input signals of the GTM. Several characteristics of the input signals can be measured inside the TIM channels. For advanced data processing the detected input characteristics of the TIM module can be routed through the ARU to subsequent processing units of the GTM.

Input characteristics mean either time stamp values of detected input rising or falling edges together with the new signal level or the number of edges received since channel enable together with the actual time stamp or PWM signal durations for a whole PWM period.

The architecture of TIM is shown in [Figure 26-20](#).

26.10.1.1 TIM Block Diagram

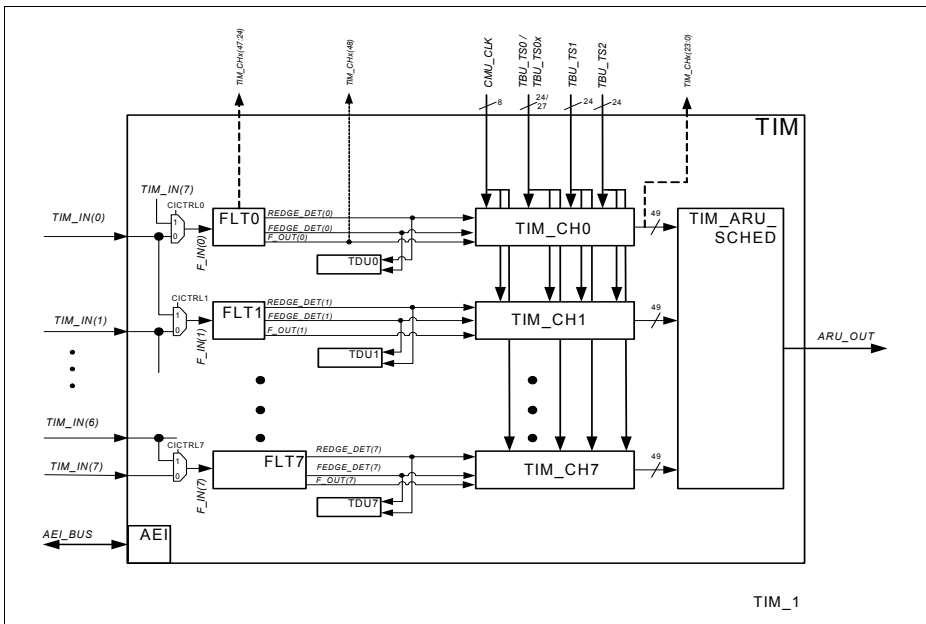


Figure 26-20 TIM Block Diagram

Generic Timer Module (GTM)

Each of the eight (8) dedicated input signals is filtered inside the FLTx subunit of the TIM Module. It should be noted that the incoming input signals are synchronized to the clock SYS_CLK, resulting in a delay of two SYS_CLK periods for the incoming signals.

The submodule TIM provides different filter mechanisms described in more detail in [Section 26.10.2](#). After filtering, the signal is routed to the corresponding TIM channel.

The measurement values can be read by the CPU directly via the AEI-Bus or they can be routed through the ARU to other submodules of the GTM.

For timeout detection of an incoming signal (no subsequent edge detected during a specified duration) each individual channel has a Timeout Detection Unit (TDU).

For the GTM TIM0 submodule only, the dashed signal outputs TIM[i]_CH[x](23:0), TIM[i]_CH[x](47:24) and TIM[i]_CH[x](48) come from the TIM0 submodule channels zero (0) to five (5) and are connected to MAP submodule. There, they are used for further processing and for routing to the DPLL.

The two (three) time bases coming from the TBU are connected to the TIM channels to annotate time stamps to incoming signals. For TIM0 the extended 27 bit width time base TBU_TS0 is connected to the TIM channels, and the user has to select if the lower 24 bits (TBU_TS0(23...0)) or the higher 24 bits (TBU_TS0(26...3)) are stored inside the GPR0 and GPR1 registers.

26.10.1.2 Input source selection INPUTSRCx

It can be configured which source shall be used for processing in the FLT, TDU, TIM_CH units. It can be selected by the bit fields CICTRLx and MODE_x, VAL_x in the register TIMi_CHx_IN_SRC which source is in use.

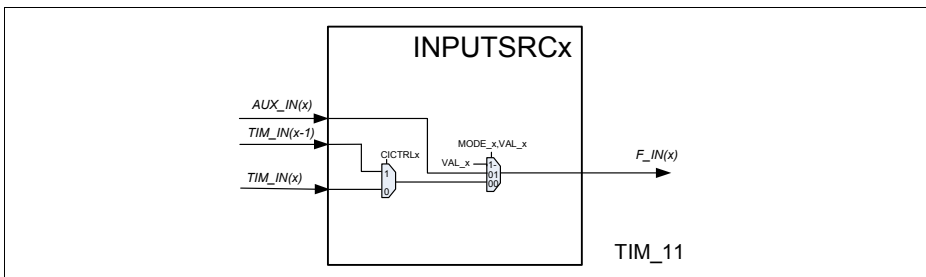


Figure 26-21 TIM Source Selection

In a certain MODE_x, VAL_x combination the input signal F_IN(x) can be driven by VAL_x(1) with 0 or 1 directly.

Due to the fact that all 8 channels are bundled in the register TIMi_CHx_IN_SRC a synchronous control of all 8 input channels is possible.

Generic Timer Module (GTM)

Two adjacent channels can be combined by setting the CICTRL bit field in the corresponding TIMi_CHx_CTRL register. This allows for a combination of complex measurements on one input signal with two TIM channels.

The additional input signal AUX_IN[x] can be selected as an input signal. The source of this signal is defined in the [Section 26.2.1.3](#).

26.10.1.3 External capture source selection EXTCAPSRCx

Each channel can operate on an external capture signal EXT_CAPTURE. The source to use for this signal can be configured by the bit field EXT_CAP_SRCx in the register TIMi_CHx_ECTRL.

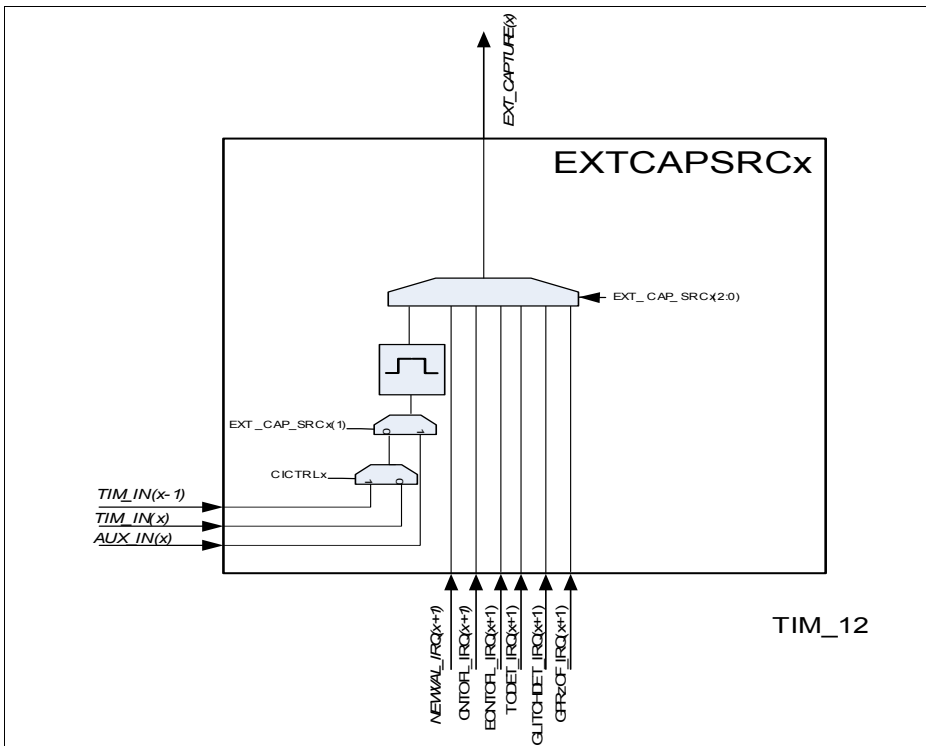


Figure 26-22 TIM external Capture Interrupt

The external capture functionality can be enabled for the TIM channel x with the bit EXT_CAP_EN in the register TIMi_CHx_CTRL, it will trigger on each rising edge. A pulse generation for each rising edge of the selected input signal TIM_IN[x] and AUX_IN[x] is applied.

Generic Timer Module (GTM)

The six TIM channel interrupt sources can be triggered by the operation in the certain TIM channel modes. Alternatively they can be issued by a soft trigger using the corresponding bits in the register `TIMi_CHx+1_FORCINT`.

26.10.2 TIM Filter Functionality (FLT)

26.10.2.1 Overview

The TIM submodule provides a configurable filter mechanism for each input signal. These filter mechanism is provided inside the FLT subunit.

FLT architecture is shown in [Figure 26-23](#)

The filter includes a clock synchronisation unit (CSU), an edge detection unit (EDU), and a filter counter associated to the filter unit (FLTU).

The CSU is synchronizing the incoming signal `F_IN` to the selected filter clock frequency, which is controlled with the bit field `FLT_CNT_FRQ` of register `TIMi_CHx_CTRL`.

The synchronized input signal `F_IN_SYNC` is used for further processing within the filter. It should be noted that glitches with a duration less than the selected CMU clock period are lost.

The filter modes can be applied individually to the falling and rising edges of an input signal. The following filter modes are available:

- immediate edge propagation mode,
- individual de-glitch time mode (up/down counter), and
- individual de-glitch time mode (hold counter).

FLT Architecture

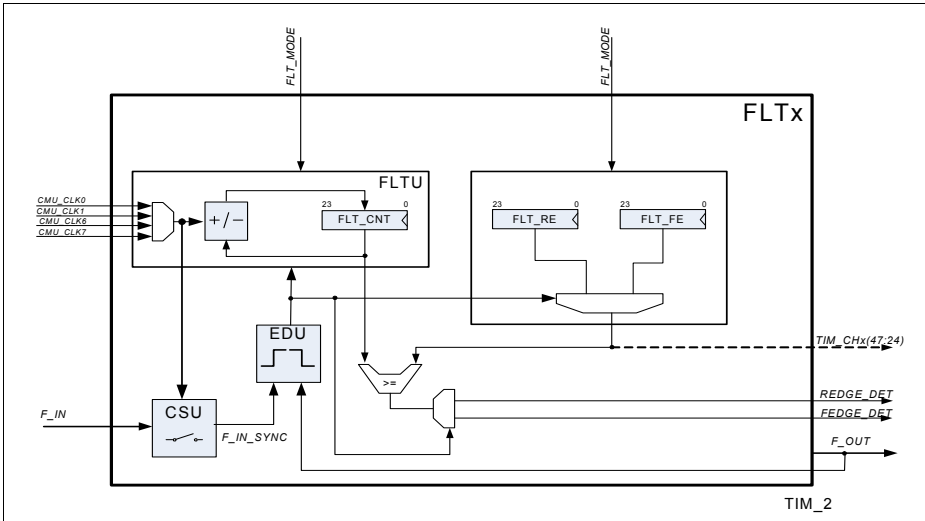


Figure 26-23 FLT Architecture

The filter parameters (deglitch and acceptance time) for the rising and falling edge can be configured inside the two filter parameter registers FLT_RE (rising edge) and FLT_FE (falling edge). The exact meaning of the parameter depends on the filter mode.

However the delay time T of both filter parameters FLT_xE can always be determined by:

$$T = (FLT_xE + 1) * TFLT_CLK,$$

whereas TFLT_CLK is the clock period of the selected CМУ clock signal in bit field FLT_CNT_FRQ of register TIMi_CHx_CTRL.

When a glitch is detected on an input signal a status flag GLITCHDET is set inside the TIMi_CHx_IRQ_NOTIFY register.

Table 26-19 gives an overview about the meanings for the registers FLT_RE and FLT_FE. In the individual deglitch time modes, the actual filter threshold for a detected regular edge is provided on the TIM[i]_CH[x](47:24) output line. In the case of immediate edge propagation mode, a value of zero is provided on the TIM[i]_CH[x](47:24) output line.

The TIM[i]_CH[x](47:24) output line is used by the MAP submodule for further processing (please see [Section 26.15](#)).

Filter Parameter summary for the different Filter Modes

Table 26-19 Filter Parameter summary for the different Filter Modes

Filter mode	Meaning of FLT_RE	Meaning of FLT_FE
Immediate edge propagation	Acceptance time for rising edge	Acceptance time for falling edge
Individual de-glitch time (up/down counter)	De-glitch time for rising edge	De-glitch time for falling edge
Individual de-glitch time (hold counter)	De-glitch time for rising edge	De-glitch time for falling edge

A counter FLT_CNT is used to measure the glitch and acceptance times.

The frequency of the FLT_CNT counter is configurable in bit field FLT_CNT_FRQ of register TIMi_CHx_CTRL.

The counter FLT_CNT can either be clock with the CMU_CLK0, CMU_CLK1, CMU_CLK6 or the CMU_CLK7 signal. These signals are coming from the CMU submodule.

The FLT_CNT, FLT_FE and FLT_RE registers are 24-bit width. For example, when the resolution of the CMU_CLK0 signal is 50ns this allows maximal de-glitch and acceptance times of about 838ms for the filter.

26.10.2.2 TIM Filter Modes

Immediate Edge Propagation Mode

In immediate edge propagation mode after detection of an edge the new signal level on F_IN_SYNC is propagated to F_OUT with a delay of one TFLT_CLK period and the new signal level remains unchanged until the configured acceptance time expires.

For each edge type the acceptance time can be specified separately in the FLT_RE and FLT_FE registers.

Each signal change on the input F_IN_SYNC during the duration of the acceptance time has no effect on the output signal level F_OUT of the filter but it sets the glitch GLITCHDET bit in the TIMi_CHx_IRQ_NOTIFY register.

After it expires an acceptance time the input signal F_IN_SYNC is observed and on signal level change the filter raises a new detected edge and the new signal level is propagated to F_OUT.

Independent of a signal level change the value of F_OUT is always set to F_IN_SYNC, when the acceptance time expires (see also [Figure 26-25](#)).

Generic Timer Module (GTM)

Figure 26-24 shows an example for the immediate edge propagation mode, in the case of rising edge detection. Both, the signal before filtering (F_IN) and after filtering (F_OUT) are shown. The acceptance time at1 is specified in the register FLT_RE.

Immediate Edge Propagation Mode in the case of a rising edge

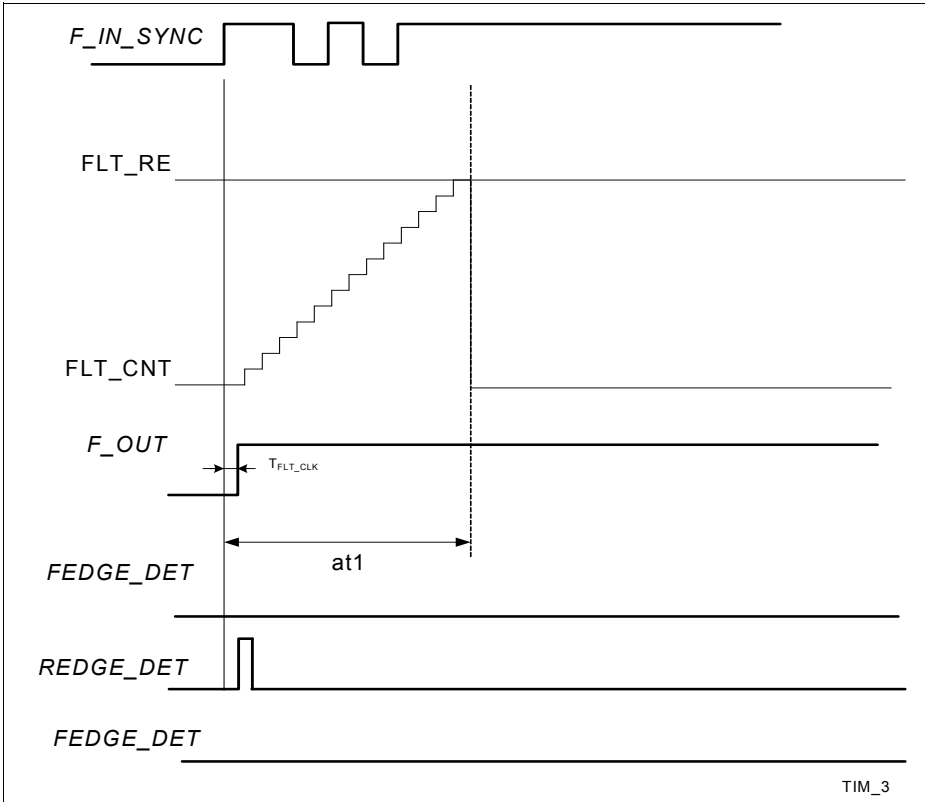


Figure 26-24 Immediate Edge Propagation Mode in the case of a rising edge

In immediate edge propagation mode the glitch measurement mechanism is not applied to the edge detection. Detected edges on F_IN_SYNC are transferred directly to F_OUT. The counter FLT_CNT is increment until acceptance time threshold is reached.

Figure 26-25 shows a more complex example of the TIM filter, in which both, rising and falling edges are configured in immediate edge propagation mode.

Generic Timer Module (GTM)

Immediate Edge Propagation Mode in the case of a rising and falling edge

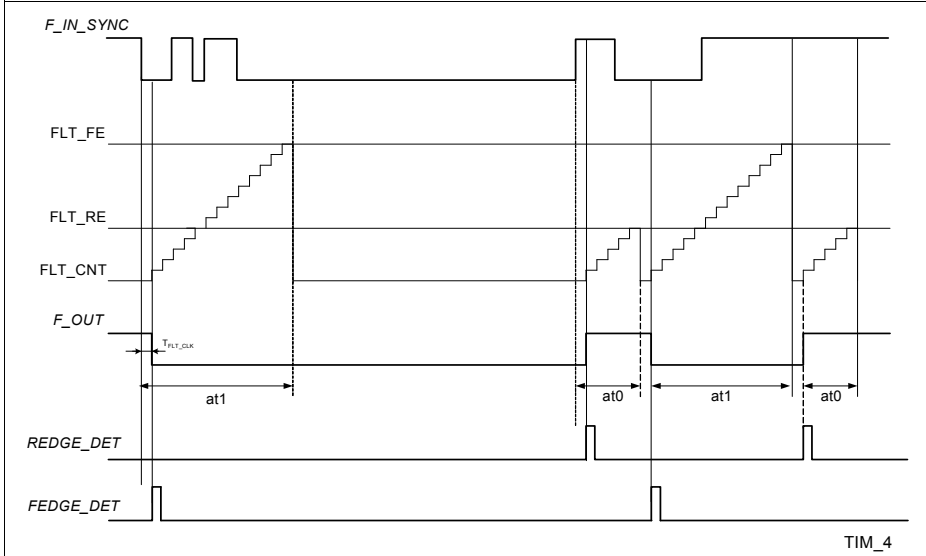


Figure 26-25 Immediate Edge Propagation Mode in the case of a rising and falling edge

If the FLT_CNT has reached the acceptance time for a specific signal edge and the signal F_IN_SYNC has already changed to the opposite level of F_OUT, the opposite signal level is set to F_OUT and the acceptance time measurement is started immediately. **Figure 26-25** shows this scenario at the detection of the first rising edge and the second falling edge.

Individual De-Glitch Time Mode (up/down counter)

In individual de-glitch time mode (up/down counter) each edge of an input signal can be filtered with an individual de-glitch threshold filter value mentioned in the registers FLT_RE and FLT_FE, respectively.

The filter counter register FLT_CNT is increment when the signal level on F_IN_SYNC is unequal to the signal level on F_OUT and decremented if F_IN_SYNC equals F_OUT. If After FLT_CNT has reached a value of zero during decrease the counter is stopped immediately.

If a glitch is detected a glitch detection bit GLITCHDET is set in the TIMi_CHx_IRQ_NOTIFY register.

Generic Timer Module (GTM)

The detected edge signal together with the new signal level is propagated to F_OUT after the individual de-glitch threshold is reached. **Figure 26-26** shows the behaviour of the filter in individual de-glitch time (up/down counter) mode in the case of the rising edge detection.

Individual Deglitch Time Mode (up/down counter) in the case of a rising edge

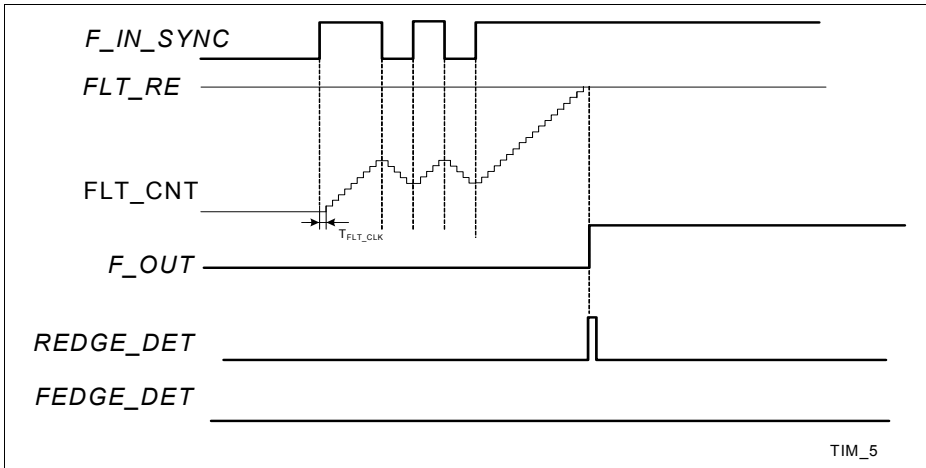


Figure 26-26 Individual Deglitch Time Mode (up/down counter) in the case of a rising edge

Individual De-Glitch Time Mode (hold counter)

In individual de-glitch time mode (hold counter) each edge of an input signal can be filtered with an individual de-glitch threshold filter value mentioned in the registers FLT_RE and FLT_FE, respectively.

The filter counter register FLT_CNT is increment when the signal level on F_IN_SYNC is unequal to the signal level on F_OUT and the counter value of FLT_CNT is hold if F_IN equals F_OUT.

If a glitch is detected the glitch detection bit GLITCHDET is set in the TIMi_CHx_IRQ_NOTIFY register.

The detected edge signal together with the new signal level is propagated to F_OUT after the individual de-glitch threshold is reached. **Figure 26-27** shows the behaviour of the filter in individual de-glitch time (hold counter) mode in the case of the rising edge detection.

Individual Deglitch Time Mode (hold counter) in the case of a rising edge

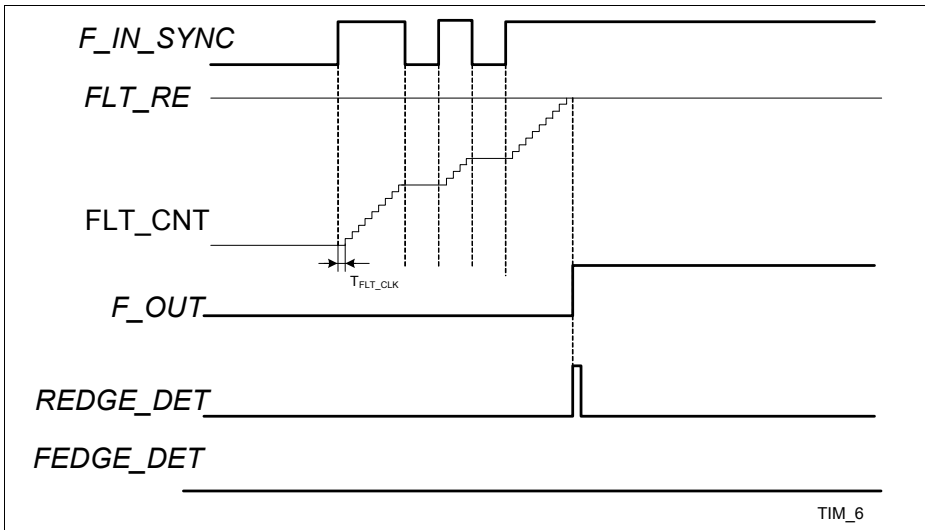


Figure 26-27 Individual Deglitch Time Mode (hold counter) in the case of a rising edge

Immediate Edge Propagation and Individual De-Glitch Mode

As already mentioned, the three different filter modes can be applied individually to each edge of the measured signal.

However, if one edge is configured with immediate edge propagation and the other edge with an individual deglitch mode (whether up/down counter or hold counter) a special consideration has to be applied.

Assume that the rising edge is configured for immediate edge propagation and the falling edge with individual deglitch mode (up/down counter) as shown in [Figure 26-28](#).

If the falling edge of the incoming signal already occurs during the measuring of the acceptance time of the rising edge, the measurement of the deglitch time on the falling edge is started delayed, but immediately after the acceptance time measurement phase of the rising edge has finished.

Consequently, the deglitch counter can not measure the time **TERROR**, as shown in [Figure 26-28](#).

Mixed mode measurement

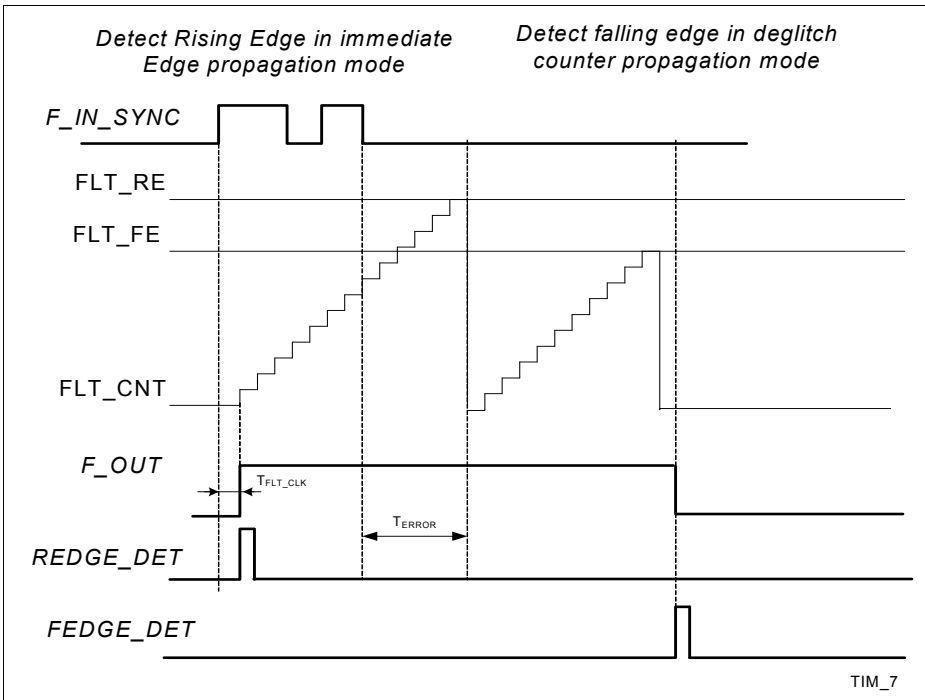


Figure 26-28 Mixed mode measurement

26.10.2.3 TIM Filter reconfiguration

If FLT_EN=1 a change of FLT_RE or FLT_FE will take place immediately. If FLT_EN=1 a change of FLT_MODE_RE or FLT_FE will be used with the next occurring corresponding edge. If mode is changed while the filter unit is processing a certain mode, it will end this edge filtering in the mode as started.

If FLT_EN=1 a change of FLT_DTR_RE or FLT_CTR_FE will take place immediately.

26.10.3 Timeout Detection Unit (TDU)

The Timeout Detection Unit (TDU) is responsible for timeout detection of the TIM input signals.

Each channel of the TIM submodule has its own Timeout Detection Unit (TDU) where a timeout event can be set up on the filtered input signal of the corresponding channel.

The TDU architecture is shown in [Figure 26-29](#).

26.10.3.1 Architecture of the TDU Subunit

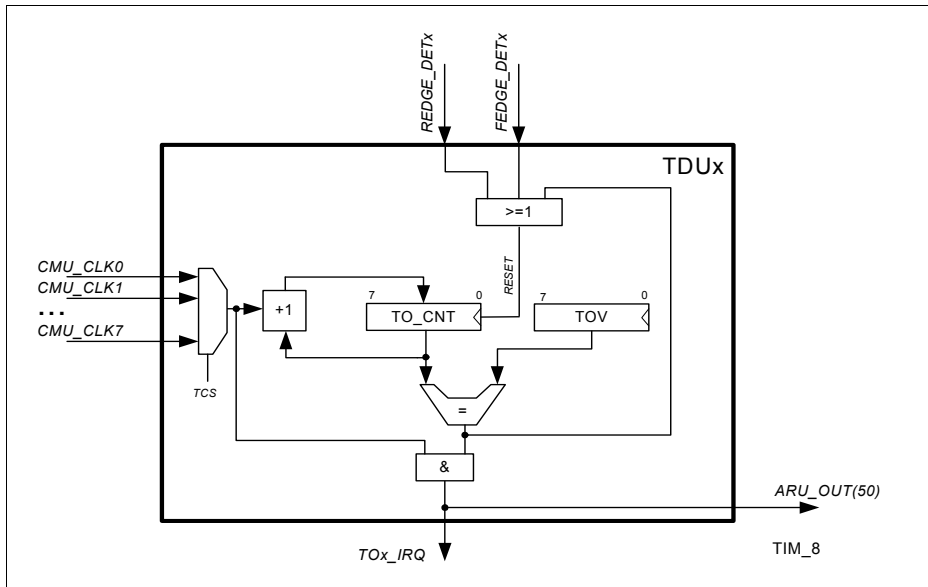


Figure 26-29 Architecture of the TDU Subunit

It is possible to detect timeouts with the resolution of the specified CMU_CLKx input signal selected with the bit field TCS of the register $TIMi_CHx_TDU$. The individual timeout values have to be specified in number of ticks of the selected input clock signal and have to be specified in the field TOV of timeout value register $TIMi_CHx_TDUV$ of the TIM channel x ($x:0...7$).

The exact time out value $TTDU$ can be calculated with:

$$T_{TDU} = (TOV + 1) * T_{CMU_GCLKx}$$

whereas T_{CMU_GCLKx} is the clock period of the selected CMU clock signal.

Timeout detection can be enabled or disabled individually inside the $TIMi_CHx_CTRL$ register by setting/resetting the TO_CTRL bit.

Timeout detection can be enabled to be sensitive to falling, rising or both edges of the input signal by writing the corresponding values to the bit field $TOCTRL$.

The counter TO_CNT is reset by each detected input edge coming either from the filtered input signal or when the timeout value TOV is reached by the counter TO_CNT .

After such a reset or by enabling the channel inside the $TIMi_CHx_CTRL$ register the counter TO_CNT starts counting again with the specified clock input signal.

Generic Timer Module (GTM)

Otherwise, timeout measurements starts immediately after the TO_CTRL bit inside the TIMi_CHx_CTRL register is written (enabled).

The TDU generates an interrupt signal TIM_TODETx_IRQ whenever a timeout is detected for an individual input signal, and the TODET bit is set inside the TIMi_CHx_IRQ_NOTIFY register.

In addition, when the ARU access is enabled with the ARU_EN bit inside the TIMi_CHx_CTRL register, the actual values stored inside the registers TIM[i]_CHx_GPR0 and TIM[i]_CHx_GPR1 are sent together with the last stored signal level to the ARU if a timeout event occurs.

To signal that a timeout occurred, the ARU_OUT(50) bit (ACB(2)) is set. The bit ACB(0) will be updated with the timeout event to the signal level on which the timeout was detected.

Thus, a destination could determine if a timeout occurred at the TIM input by evaluating ACB bit 2.

Since the TIM channel still monitors its input pin although the timeout happened, a valid edge could occur at the input pin while the timeout information is still valid at the ARU. In that case, the new edge associated data is stored inside the registers TIM[i]_CHx_GPR0 and TIM[i]_CHx_GPR1, the GPR overflow detected bit is set together in the ACB field (ACB(1)) with the timeout bit (ACB(2)) and the values are marked as valid to the ARU.

The ACB bit 2 is cleared, when a successful ARU write access by the TIM channel took place.

The ACB bit 1 is cleared, when a successful ARU write access by the TIM channel took place.

When a valid edge initiates an ARU write access which has not ended while a new timeout occurs the GPR overflow detected bit (ACB(1)) is set. The bit ACB(0) will be updated to the level on which the timeout occurred.

When a timeout occurred and initiates an ARU write access which has not ended while a new timeout occurs the GPR overflow detected bit (ACB(1)) is not set.

The following table clarifies the meaning of the ACB Bits for valid data provided by a TIM channel:

Table 26-20 TIM ARU Control Bits

ACB4/3	ACB2	ACB1	ACB0	Description
dc	0	0	SL	Valid edge detected
dc	0	1	SL	Input edge overwritten by subsequent edge

Generic Timer Module (GTM)

Table 26-20 TIM ARU Control Bits (cont'd)

ACB4/3	ACB2	ACB1	ACB0	Description
dc	1	0	SL	Timeout detected without valid edge
dc	1	1	SL	Timeout detected with subsequent valid edge detected

26.10.4 TIM Channel Architecture

26.10.4.1 Overview

Each TIM channel consist of an input edge counter ECNT, a Signal Measurement Unit (SMU) with a counter CNT, a counter shadow register CNTS for SMU counter and two general purpose registers GPR0 and GPR1 for value storage.

The value TOV of the timeout register TIMi_CHx_TDU is provided to TDU subunit of each individual channel for timeout measurement. The architecture of the TIM channel is depicted in [Figure 26-30](#).

TIM Channel Architecture

Generic Timer Module (GTM)

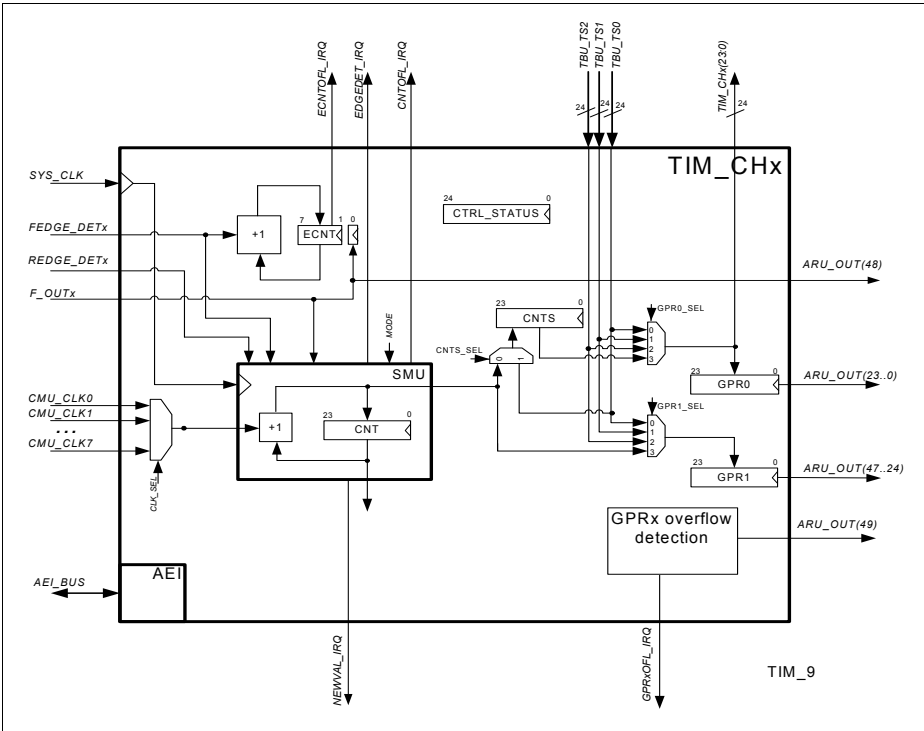


Figure 26-30 TIM Channel Architecture

Each TIM channel receives both input trigger signals REDGE_DET_x and FEDGE_DET_x, generated by the corresponding filter module in order to signalize a detected echo of the input signal F_IN_x. The signal F_OUT_x shows the filtered signal of the channel's input signal F_IN_x.

The edge counter ECNT counts every incoming filtered edge (rising and falling). The counter value is uneven in case of detected rising, and even in case of detected falling edge. Thus, the input signal level is part of the counter and can be obtained by bit 0 of ECNT. (However, the actual counter implementation counts only falling edges on ECNT[n:1] bits. It generates ECNT by composing the ECNT[n:1] bits with F_OUT_x as bit 0).

Thus, the whole counter value is always odd, when a positive edge was received and always even, when a negative edge was received.

The current ECNT[7:0] register content is made visible on the bits 31 down to 24 of the registers GPR0, GPR1 and CNTS. This allows the software to detect inconsistent read accesses to registers GPR0, GPR1, and CNTS. However, the update strategy of these

Generic Timer Module (GTM)

registers depends on the selected TIM modes, and thus the consistency check has to be adapted carefully.

It can be chosen with the bit field FR_ECNT_OFL when an ECNT overflow is signaled on ECNTOFL. An ECNT overflow can be signaled on 8 bit or full range resolution.

While reading the register TIMi_CHx_ECNT the bit ECNT[0] shows the input signal value F_OUTx independent of the state (enabled / disabled) of the channel. If a channel gets disabled (OSM mode or resetting TIM_EN) the content of TIMi_CHx_ECNT will be frozen until a read of the register takes place. This read will reset the ECNT counter. Continuing reads will show the input signal in bit ECNT[0] again.

When new data is written into GPR0 and GPR1 the NEWVAL bit is set in TIMi_CHx_IRQ_NOTIFY register and depending on corresponding enable bit value the NEWVALx_IRQ interrupt is raised.

Each TIM input channel has an ARU connection for providing data via the ARU to the other GTM submodules. The data provided to the ARU depends on the TIM channel mode and its corresponding adjustments (e.g. multiplexer configuration).

The bit ARU_EN of register TIMi_CHx_CTRL decides, whether the measurement results of registers GPR0 and GPR1 are consumed by another submodule via ARU (ARU_EN = 1) or the CPU via AEI (ARU_EN = 0).

To guarantee a consistent delivery of data from the GPR0 and GPR1 registers to the ARU or the CPU each TIM channel has to ensure that the data is consumed before it is overwritten with new values.

If new data was produced by the TIM channel (bit NEWVAL is set inside TIMi_CHx_IRQ_NOTIFY register) while the old data is not consumed by the ARU (ARU_EN = 1) or CPU (ARU_EN = 0), the TIM channel sets the GPROFL bit inside the status register TIMi_CHx_IRQ_NOTIFY and it overwrites the data inside the GPR registers GPR0 and GPR1. In addition when ARU_EN=1 the bit ACB(1) is set to 1 to indicate the overflow in the ARU data.

If the CPU is selected as consumer for the registers GPR0 and GPR1 (ARU_EN = 0), the acknowledge for reading out data is performed by a read access to the register GPR0. Thus, register GPR1 should be read always before GPR0.

If the ARU is selected as consumer for the registers GPR0 and GPR1 (ARU_EN = 1), the acknowledge for reading out data is performed by the ARU itself. However, the registers GPR0 and GPR1 could be read by CPU without giving an acknowledge.

26.10.4.2 TIM Channel Modes

The TIM provides six different measurement modes that can be configured with the bit field TIM_MODE of register TIMi_CHx_CTRL. The measurement modes are described in the following subsections. Besides these different basic measurement modes, there exist distinct configuration bits in the register TIMi_CHx_CTRL for a more detailed controlling of each mode. The meanings of these bits are as follows:

Generic Timer Module (GTM)

- DSL: control the signal level for the measurement modes (e.g. if a measurement is started with rising edge or falling edge, or if high level pulses or low level pulses are measured).
- EGPR0_SEL, GPR0_SEL and EGPR1_SEL, GPR1_SEL: control the actual content of the registers GPR0 and GPR1 after a measurement has finished.
- CNTS_SEL: control the content of the registers CNTS. The actual time for updating the CNTS register is mode dependent.
- OSM: activate measurement in one-shot mode or continuous mode. In one-shot mode only one measurement cycle is performed and after that the channel is disabled.
- NEWVAL: The NEWVAL IRQ interrupt is triggered at the end of a measurement cycle, signalling that the registers GPR0 and GPR1 are updated.
- ARU_EN: enables sending of the registers GPR0 and GPR1 together with the actual signal level (in bit 48) and the overflow signal GPROFL (in bit 49), and the timeout status information (bit 50) to the ARU.
- EXT_CAP_EN: forces an update of the registers GPR0 and GPR1 and CNTS (TIM channel mode dependant) only on each rising edge of the EXT_CAPTURE signal and triggers a NEWVAL IRQ interrupt. If this mode is disabled the NEWVAL IRQ interrupt is triggered at the end of each measurement cycle.

For each channel the source of the EXT_CAPTURE signal can be configured with the bit fields EXT_CAP_SRC in the register TIMi_CHx_ECTRL.

TIM PWM Measurement Mode (TPWM)

In TIM PWM Measurement Mode the TIM channel measures duty cycle and period of an incoming PWM signal. The DSL bit defines the polarity of the PWM signal to be measured.

When measurement of pulse high time and period is requested (PWM with a high level duty cycle, DSL=1), the channel starts measuring after the first rising edge is detected by the filter.

Measurement is done with the CNT register counting with the configured clock coming from CMU_CLKx until a falling edge is detected.

Then the counter value is stored inside the shadow register CNTS (if CNTS_SEL = 0) and the counter CNT counts continuously until the next rising edge is reached.

On this following rising edge the content of the CNTS register is transferred to GPR0 and the content of CNT register is transferred to GPR1, assuming settings for the selectors GPR0_SEL=11 and GPR1_SEL=11. By this, GPR0 contains the duty cycle length and GPR1 contains the period. It should be noted, that the bits 1 to 7 of the ECNT may be used to check data consistency of the registers GPR0 and GPR1.

In addition the CNT register is cleared NEWVAL status bit inside of TIMi_CHx_IRQ_NOTIFY status register and depending on corresponding interrupt enable condition TIM_NEWVALx_IRQ interrupt is raised.

Generic Timer Module (GTM)

The CNTS register update is not performed until measurement is started (first edge defined by DSL is detected). Afterwards each edge leaving the level defined by DSL is performing a CNTS register update.

If a PWM with a low level duty cycle should be measured (DSL = 0), the channel waits for a falling edge until measurement is started. On this edge the low level duty cycle time is stored first in CNTS and then finally in GPR0 and the period is stored in GPR1.

When a PWM period was successfully measured, the data in registers GPR0 and GPR1 is marked as valid for reading by the ARU when the ARU_EN bit is set inside TIMi_CHx_CTRL register, the NEWVAL bit is set inside the TIMi_CHx_IRQ_NOTIFY register, and a new measurement is started.

If the preceding PWM values were not consumed by a reader attached to the ARU (ARU_EN bit enabled) or by the CPU the TIM channel set GPROFL status bit in TIMi_CHx_IRQ_NOTIFY and depending on corresponding interrupt enable bit value raises a GPROFL_IRQ and overwrites the old values in GPR0 and GPR1. A new measurement is started afterwards.

If the register CNT produces an overflow during the measurement, the bit CNTOFL is set inside the register TIMi_CHx_IRQ_NOTIFY and interrupt TIM_CNTOFL[x]_IRQ is raised depending on corresponding interrupt enable condition.

If the register ECNT produces an overflow during the measurement, the bit ECNTOFL is set inside the register TIMi_CHx_IRQ_NOTIFY and interrupt TIM_ECNTOFL[x]_IRQ is raised depending on corresponding interrupt enable condition.

External capture TIM PWM Measurement Mode (TPWM)

If external capture is enabled, the pwm measurement is done continuously. The actual measurement values are captured to GPRx if an external capture event occurs.

Operation is done depending on cmu clock, ISL, DSL bit and the input signal value defined in the next table:

Table 26-21 TIM PWM Measurement Mode

Input signal F_OUTx	selected CMU Clock	External capture	ISL	DSL	Action description
0	1	0	-	0	CNT++
1	1	0	-	0	no
rising edge	-	0	0	0	capture CNT value in CNTS
falling edge	-	0	0	0	CNT = 0
rising edge	-	0	1	0	no
falling edge	-	0	1	0	capture CNT value in CNTS; CNT = 0

Generic Timer Module (GTM)
Table 26-21 TIM PWM Measurement Mode (cont'd)

Input signal F_OUTx	selected CMU Clock	External capture	ISL	DSL	Action description
1	1	0	-	1	CNT++
0	1	0	-	1	no
falling edge	-	0	0	1	capture CNT value in CNTS
rising edge	-	0	0	1	CNT = 0
falling edge	-	0	1	1	no
rising edge	-	0	1	1	capture CNT value in CNTS; CNT = 0
-	-	rising edge	-	-	do GPRx capture; issue NEWVAL_IRQ
-	0	0	-	-	no

The CNTS register update is not performed until measurement is started (first edge defined by DSL is detected). Afterwards the update of the CNTS register is defined by ISL, DSL combinations in the table above.

TIM Pulse Integration Mode (TPIM)

In TIM Pulse Integration Mode each TIM channel is able to measure a sum of pulse high or low times on an input signal, depending on the selected signal level bit DSL of register TIMi_CHx_CTRL register.

The pulse times are measured by incrementing the TIM channel counter CNT whenever the pulse has the specified signal level DSL. The counter is stopped whenever the input signal has the opposite signal level.

The counter CNT counts with the CMU_CLKx clock specified by the CLK_SEL bit field of the TIMi_CHx_CTRL register.

The CNT register is reset at the time the channel is activated (enabling via AEI write access) and it accumulates pulses while the channel is staying enabled.

Whenever the counter is stopped, the registers CNTS, GPR0 and GPR1 are updated according to settings of its corresponding input multiplexers, using the bits GPR0_SEL, EGPR0_SEL, EGPR1_SEL, GPR1_SEL, and CNTS_SEL. It should be noted, that the bits 1 to 7 of the ECNT may be used to check data consistency of the registers GPR0 and GPR1.

When the ARU_EN bit is set inside the TIMi_CHx_CTRL register the measurement results of the registers GPR0 and GPR1 can be send to subsequent submodules attached to the ARU.

Generic Timer Module (GTM)
External capture TIM Pulse Integration Mode (TPIM)

If external capture is enabled, the pulse integration is done until next external capture event occurs.

Operation is done depending on CMU clock, DSL bit and the input signal value defined in the next table:

Table 26-22 TIM integration Mode

Input signal F_OUTx	selected CMU Clock	External capture	ISL	DSL	Action description
0	1	0	-	0	CNT++
1	1	0	-	0	no
1	1	0	-	1	CNT++
0	1	0	-	1	no
-	-	rising edge	-	-	do GPRx capture; issue NEWVAL_IRQ
-	0	0	-	-	no

TIM Input Event Mode (TIEM)

In TIM Input Event Mode the TIM channel is able to count edges.

It is configurable if rising, falling or both edges should be counted. This can be done with the bit fields DSL and ISL in TIMi_CHx_CTRL register.

In addition, a TIM[i]_NEWVAL[x]_IRQ interrupt is raised when the configured edge was received and this interrupt was enabled.

The counter register CNT is used to count the number of edges, and the bit fields EGPR0_SEL, EGPR1_SEL, GPR0_SEL, GPR1_SEL, and CNTS_SEL can be used to configure the desired update values for the registers GPR0, GPR1 and CNTS. These register are updated whenever the edge counter CNT is increment due to the arrival of a desired edge.

If the preceding data was not consumed by a reader attached to the ARU or by the CPU the TIM channel sets GPROFL status bit and raises a GPROFL[x]_IRQ if it was enabled in TIMi_CHx_IRQ_EN register and overwrites the old values in GPR0 and GPR1 with the new ones.

If the register CNT produces an overflow during the measurement, the bit CNTOFL is set inside the register TIMi_CHx_IRQ_NOTIFY and interrupt TIM_CNTOFL[x]_IRQ is raised depending on corresponding interrupt enable condition.

If the register ECNT produces an overflow during the measurement, the bit ECNTOFL is set inside the register TIMi_CHx_IRQ_NOTIFY and interrupt TIM_ECNTOFL[x]_IRQ is raised depending on corresponding interrupt enable condition.

Generic Timer Module (GTM)

The TIM Input Event Mode does not depend on the bit field CLK_SEL of register TIMi_CHx_CTRL.

External capture TIM Input Event Mode (TIEM)

If external capture is enabled, capturing is done depending on ISL, DSL bit and the input signal value defined in the next table:

Table 26-23 TIM Input Event Mode

Input signal F_OUTx	External capture	ISL	DSL	Action description
-	rising edge	1	-	do capture; issue NEWVAL_IRQ; CNT++
-	0	-	-	no
1	rising edge	0	1	do capture; issue NEWVAL_IRQ; CNT++
0	-	0	1	no
0	rising edge	0	0	do capture; issue NEWVAL_IRQ; CNT++
1	-	0	0	no

TIM Input Prescaler Mode (TIPM)

In the TIM Input Prescaler Mode the number of edges which should be detected before a TIM[i]_NEWVAL[x]_IRQ is raised is programmable. In this mode it must be specified in the CNTS register after how many edges the interrupt has to be raised.

A value of 0 in CNTS means that after one edge an interrupt is raised, and a value of 1 means that after two edges an interrupt is raised, and so on.

The edges to be counted can be selected by the bit fields DSL and ISL of register TIMi_CHx_CTRL.

With each triggered interrupt, the registers GPR0 and GPR1 are updated according to bits EGPR0_SEL, EGPR1_SEL, GPR0_SEL and GPR1_SEL.

If the register ECNT produces an overflow during the measurement, the bit ECNTOFL is set inside the register TIMi_CHx_IRQ_NOTIFY and interrupt TIM_ECNTOFL[x]_IRQ is raised depending on corresponding interrupt enable condition.

The TIM Input Prescaler Mode does not depend on the bit field CLK_SEL of register TIMi_CHx_CTRL.

Generic Timer Module (GTM)
External capture TIM Input Prescaler Mode (TIPM)

If external capture is enabled, the external capture events are counted instead of the input signal edges.

Operation is done depending on DSL, ISL bit and the input signal value defined in the next table:

Table 26-24 TIM Input Event Mode

Input signal F_OUTx	External capture	ISL	DSL	Action description
-	rising edge	1	-	if CNT == CNTS then do capture; issue NEWVAL_IRQ; CNT = 0; else CNT++ endif
-	0	1	-	no
1	rising edge	0	1	if CNT == CNTS then do capture; issue NEWVAL_IRQ; CNT = 0; else CNT++ endif
0	-	0	1	no
0	rising edge	0	0	if CNT == CNTS then do capture; issue NEWVAL_IRQ; CNT = 0; else CNT++ endif
1	-	0	0	no

TIM Bit Compression Mode (TBCM)

The TIM Bit Compression Mode can be used to combine all filtered input signals of a TIM submodule to a parallel 8 bit data word, which can be routed to the ARU.

Since this mode uses all eight input signals with its input filters, it is only available within TIM channel 0 of each TIM submodule. [Figure 26-31](#) gives an overview of the TIM bit compression mode.

TIM Bit Compression Mode

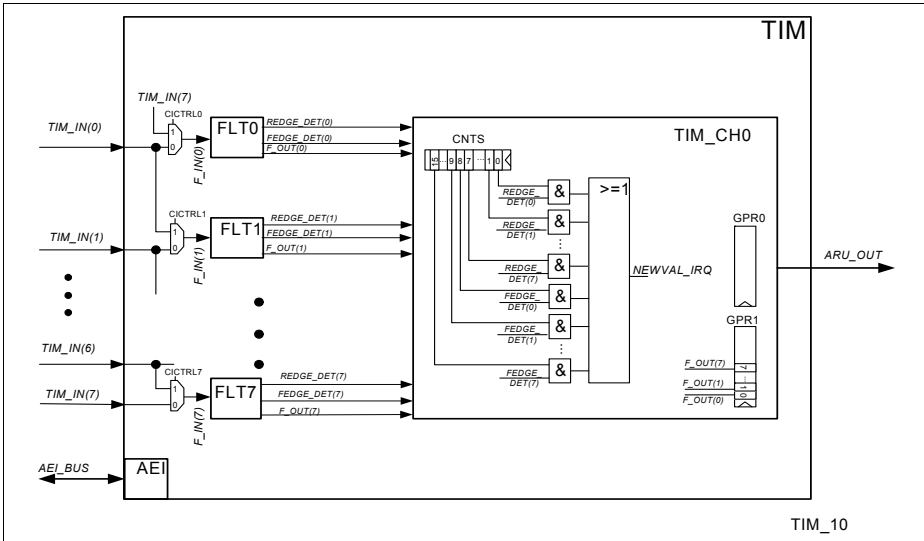


Figure 26-31 TIM Bit Compression Mode

The register CNTS of TIM channel 0 is used to configure the event that releases the NEWVAL_IRQ and samples the input signals F_IN(0) to F_IN(7) in ascending order as a parallel data word in GPR1.

The bits 0 to 7 of the CNTS register are used to select the REDGE_DET signals of the TIM filters 0 to 7 as a sampling event, and the bits 8 to 15 are used to select the FEDGE_DET signals of the TIM filters 0 to 7, respectively. If multiple events are selected, the events are OR-combined (see also [Figure 26-31](#)).

EGPR0_SEL, GRP0_SEL selects the timestamp value, which is routed through the ARU. GRP1_SEL is not applicable in TBCM mode.

If the bit ARU_EN of register TIMi_CH0_CTRL is set, the sampled data of register GPR1 is routed together with a time stamp of register GPR0 to the ARU, whenever the NEWVAL_IRQ is released.

In TIM Bit compression mode, the register ECNT increments on with each NEWVAL_IRQ, which means that the value of ECNT may depend on all eight input signals. Consequently, the LSB of ECNT does not reflect the actual level of the input signal TIM_IN0.

If the register ECNT produces an overflow during the measurement, the bit ECNTOFL is set inside the register TIMi_CHx_IRQ_NOTIFY and interrupt TIM_ECNTOFL[x]_IRQ is raised depending on corresponding interrupt enable condition.

The TIM Bit Compression Mode does not depend on the bit field CLK_SEL of register TIMi_CHx_CTRL.

Generic Timer Module (GTM)
External capture Bit Compression Mode (TBCM)

If external capture is enabled, capturing is done depending on DSL, ISL bit and the input signal value defined in the next table:

Table 26-25 TIM Input Event Mode

Input signal F_OUTx	External capture	ISL	DSL	Action description
-	rising edge	1	-	do capture; issue NEWVAL_IRQ; CNT++
-	0	1	-	no
1	rising edge	0	1	do capture; issue NEWVAL_IRQ; CNT++
0	-	0	1	no
0	rising edge	0	0	do capture; issue NEWVAL_IRQ; CNT++
1	-	0	0	no

TIM Gated Periodic Sampling Mode (TGPS)

In the TIM Gated Periodic Sampling Mode the number of CMU clock cycles which should elapse before capturing and rising TIMi_NEWVALx_IRQ is programmable. In this mode it must be specified in the CNTS register after how many CMU clock cycles the interrupt has to be raised.

A value of 0 in TIMi_CHx_CNTS means that after one CLK_SEL edge a trigger/interrupt is raised, and a value of 1 means that after two edges a trigger/interrupt is raised, and so on.

In the TIMi_CHx_CNT register the elapsed cycles were incremented and compared against TIMi_CHx_CNTS. If TIMi_CHx_CNT is greater or equal to TIMi_CHx_CNTS a trigger will be raised. This allows by writing a value to TIMi_CHx_CNTS that the actual period time can be changed on the fly.

Operation is done depending on CMU clock, DSL, ISL bit and the input signal value defined in the next table:

Table 26-26 TIM Gated Periodic Sampling Mode

Input signal F_OUTx	selected CMU Clock	External capture	ISL	DSL	Action description
-	1	0	1	-	if CNT == CNTS then do capture; issue NEWVAL_IRQ; CNT = 0; else CNT++ endif
0	0	0	0	1	no
1	1	0	0	1	if CNT == CNTS then do capture; issue NEWVAL_IRQ; CNT = 0; else CNT++ endif
0	0	-	0	1	no
0	1	0	0	0	if CNT == CNTS then do capture; issue NEWVAL_IRQ; CNT = 0; else CNT++ endif
1	0	0	0	0	no
-	0	0	-	-	no

In this mode the TIMi_CHx_GPR1 operates as a shadow register for TIMi_CHx_CNTS. This would allow that the period for the next sampling period could be specified. The update of TIMi_CHx_CNTS will only take place once on a trigger if the TIMi_CHx_GPR1 was written by the CPU. This means that the capture value from the previous trigger can be read by the CPU from TIMi_CHx_GPR1 and afterwards the new sampling period (the one after the actual sampling period) could be written.

With each triggered interrupt, the registers GPR0 and GPR1 are updated according to bits GPR0_SEL, GPR1_SEL, EGPR0_SEL and EGPR1_SEL.

When selecting ECNT as a source for the capture registers, GPRx will show the edge count and the input signal value at point of capture. Selecting GPR0_SEL = 11_B and EGPR0_SEL = 0_B for TIM channel 0 all 8 TIM input signals will captured to GPR[7:0].

In the TGPS Mode the bit field CLK_SEL of register TIMi_CHx_CTRL will define the selected CMU clock which will be used.

The behaviour of the ECNT counter is configurable by ECNT_RESET. If set to 1 on each interrupt (period expired) the ECNT will be reset. Otherwise it operates in wrap around mode.

Generic Timer Module (GTM)

If the register ECNT produces an overflow during the measurement, the bit ECNTOFL is set inside the register TIMi_CHx_IRQ_NOTIFY and interrupt TIM_ECNTOFLx_IRQ is raised depending on corresponding interrupt enable condition.

External capture Bit Gated Periodic Sampling Mode (TGPS)

If external capture is enabled, the external capture events will capture the GPRx, reset the counter CNT and issue a NEWVAL_IRQ.

Operation is done depending on CMU clock, DSL, ISL bit and the input signal value defined in the next table:

Table 26-27 TIM Gated Periodic Sampling Mode

Input signal F_OUTx	selected CMU Clock	External capture	ISL	DSL	Action description
-	1	0	1	-	if CNT == CNTS then do capture; issue NEWVAL_IRQ; CNT = 0; else CNT++ endif
0	0	0	0	1	no
1	1	0	0	1	if CNT == CNTS then do capture; issue NEWVAL_IRQ; CNT = 0; else CNT++ endif
0	0	-	0	1	no
0	1	0	0	0	if CNT == CNTS then do capture; issue NEWVAL_IRQ; CNT = 0; else CNT++ endif
1	0	0	0	0	no
-	0	0	-	-	no
-	-	rising edge	-	-	do capture; issue NEWVAL_IRQ; CNT = 0

26.10.5 MAP Submodule Interface

The GTM provides one dedicated TIM submodule TIM0 where channels zero (0) to five (5) are connected to the MAP submodule described in [Section 26.15](#). There, the TIM0 submodule channels provide the input signal level together with the actual filter value and the annotated time stamp for the edge together in a 49 bit wide signal to the MAP

Generic Timer Module (GTM)

submodule. This 49 bit wide data signal is marked as valid with a separate valid signal `tim0_map_dval[x]` (`x:0...5`).

Table 26-28 TIM0 to MAP Interface

tim0_map_data[x](48)	signal level bit from tim0_ch[x]
<code>tim0_map_data[x](47:24)</code>	actual filter value <code>TIM0_CHx_FLT_RE/TIM0_CHx_FLT_FE</code> if corresponding channel <code>x</code> bit field <code>FLT_MODE_RE/FLT_MODE_FE</code> is 1 else 0 is assigned.
<code>tim0_map_data[x](23:0)</code>	time stamp value selected by <code>TBU0_SEL</code> , <code>GRP0_SEL</code> , <code>EGPR0_SEL</code> , <code>CNTS_SEL</code> if channel <code>x</code> if bit field <code>TIM_EN=1</code>
<code>tim0_map_dval[x]</code>	mark <code>tim0_map_data[x]</code> valid for one clock cycle

Note: With `TIM_EN=1` the MAP interface starts operation, it is not dependent on the setting of the bit fields `TIM_MODE`, `ISL`, and `DSL`.

Note: While the MAP interface is in the use the following guidelines have to be fulfilled, otherwise inconsistent filter values can be transferred.

Change `TIM0_CHx_FLT_RE` only between occurrence of rising and falling edge.

Change `TIM0_CHx_FLT_FE` only between occurrence of falling and rising edge.

26.10.6 TIM Interrupt Signals

TIM provides 6 interrupt lines per channel. These interrupts are shown below:

Table 26-29 TIM Interrupt Signals

Signal	Description
<code>TIM[i]_NEWVAL[x]_IRQ</code>	New measurement value detected by SMU of channel <code>x</code> (<code>x:0...7</code>)
<code>TIM[i]_ECNTOFL[x]_IRQ</code>	ECNT counter overflow of channel <code>x</code> (<code>x:0...7</code>)
<code>TIM[i]_CNTOFL[x]_IRQ</code>	SMU CNT counter overflow of channel <code>x</code> (<code>x:0...7</code>)
<code>TIM[i]_GPROF[x]_IRQ</code>	GPR0 and GPR1 data overflow, old data was not read out before new data has arrived at input pin (<code>x:0...7</code>)
<code>TIM[i]_TODET[x]_IRQ</code>	Time out reached for input signal of channel <code>x</code> (<code>x:0...7</code>)
<code>TIM[i]_GLITCHDET_IRQ</code>	A glitch was detected by the TIM filter of channel (<code>x: 0...7</code>).

26.10.7 TIM Configuration Registers Overview

TIM contains following configuration registers:

Table 26-30 TIM Configuration Registers Overview

Register Name	Description	Detail in Section
TIMi_CHx_CTRL	TIM channel x control register (x:0...7)	Section 26.10.8.1 Section 26.10.8.2
TIMi_CHx_ECTRL	TIM channel x (x:0...7) extended control	Section 26.10.8.2 0
TIMi_CHx_FLT_RE	TIM channel x filter parameter 0 register (x:0...7)	Section 26.10.8.3
TIMi_CHx_FLT_FE	TIM channel x filter parameter 1 register (x:0...7)	Section 26.10.8.4
TIMi_CHx_TDUV	TIM channel x TDU control register (x:0...7)	Section 26.10.8.1 7
TIMi_CHx_TDUC	TIM channel x TDU counter register (x:0...7)	Section 26.10.8.1 8
TIMi_CHx_GPR0	TIM channel x general purpose 0 register (x:0...7)	Section 26.10.8.6
TIMi_CHx_GPR1	TIM channel x general purpose 1 register (x:0...7)	Section 26.10.8.7
TIMi_CHx_CNT	TIM channel x SMU counter register (x:0...7)	Section 26.10.8.8
TIMi_CHx_ECNT	TIM channel x (x:0...7) SMU edge counter	Section 26.10.8.1 9
TIMi_CHx_CNTS	TIM channel x SMU shadow counter register (x:0...7)	Section 26.10.8.9
TIMi_CHx_IRQ_NOTIFY	TIM channel x interrupt notification register (x:0...7)	Section 26.10.8.1 0
TIMi_CHx_IRQ_EN	TIM channel x interrupt enable register (x:0...7)	Section 26.10.8.1 1
TIMi_CHx_EIRQ_EN	TIM channel x (x:0...7) error interrupt enable	Section 26.10.8.1 6
TIMi_CHx_IRQ_FORCINT	TIM channel x software interrupt force register (x:0...7)	Section 26.10.8.1 2
TIMi_CHx_IRQ_MODE	TIM IRQ mode configuration register (x=0...7)	Section 26.10.8.1 3

Generic Timer Module (GTM)**Table 26-30 TIM Configuration Registers Overview (cont'd)**

Register Name	Description	Detail in Section
TIMi_RST	TIM global software reset register	Section 26.10.8.1 4
TIMi_IN_SRC	TIM AUX IN source selection	Section 26.10.8.1 5

26.10.8 TIM Configuration Registers Description

All of the following registers are 32-bit only accessible.

26.10.8.1 Register TIM_i_CH_x_CTRL (x:0...7) (i: 12)

GTM_TIM1_CH_x_CTRL (x=0-7)
TIM1_Channel_x_CTRL Register

(01824_H+x*80_H)

Reset Value: 00000000_H

GTM_TIM2_CH_x_CTRL (x=0-7)
TIM2_Channel_x_CTRL Register

(02024_H+x*80_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOCTRL		EGP R1_ SEL	EGP R0_ SEL	FR_ ECN T_ O FL	CLK_SEL		FLT_ CTR _FE	FLT_ MOD E_ F E	FLT_ CTR _RE	FLT_ MOD E_ R E	EXT_ _CA P_ E N	FLT_CNT_ FRQ	FLT_ EN		
rw		rw	rw	rw	rw		rw	rw	rw	rw	r	rw	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECN T_ R ESE T	ISL	DSL	CNT S_ S EL	GPR1_ SE L	GPR0_ SE L	Rese rved	CICT RL	ARU _EN	OSM	TIM_MODE			TIM_ EN		
rw	rw	rw	rw	rw	rw	r	rw	rw	rw	rw			rwh		

Field	Bits	Type	Description
TIM_EN	0	rwh	TIM channel x (x:0...7) enable 0 _B Channel disabled 1 _B Channel enabled Note: Enabling of the channel resets the registers ECNT, TIM _i _CH _x _CNT, TIM _i _CH _x _GPR0, and TIM _i _CH _x _GPR1 to their reset values. Note: After finishing the action in one-shot mode the TIM_EN bit is cleared automatically. Otherwise, the bit must be cleared manually.

Generic Timer Module (GTM)

Field	Bits	Type	Description
TIM_MODE	[3:1]	rw	<p>TIM channel x (x:0...7) mode</p> <p>000_B PWM Measurement Mode (TPWM) 001_B Pulse Integration Mode (TPIM) 010_B Input Event Mode (TIEM) 011_B Input Prescaler Mode (TIPM) 100_B Bit Compression Mode (TBCM) 101_B Gated Periodic Sampling Mode (TGPS)</p> <p>Note: The Bit Compression Mode is only available in TIM channel 0. If this mode is selected in any other channels, the TIM_MODE = 000 (TPWM mode) is used instead. However, the register TIM_MODE is read with value 100.</p> <p>Note: If an undefined value is written to the TIM_MODE register, the hardware switches automatically to TIM_MODE = 000 (TPWM mode).</p> <p>Note: The TIM_MODE register should not be changed while the TIM channel is enabled.</p> <p>Note: If TIM channel is enabled and operating in TPWM or TPIM mode after the first valid edge defined by DSL has occurred, a reconfiguration of DSL, ISL, and TIM_MODE will not change the channel behavior. Reading these bit fields after reconfiguration will show the newly configured settings but the initial channel behavior will not change. Only a disabling of the TIM_EN=1 will change the channel operation mode.</p>
OSM	4	rw	<p>One-shot mode</p> <p>0_B Continuous operation mode 1_B One-shot mode</p> <p>Note: After finishing the action in one-shot mode the TIM_EN bit is cleared automatically.</p>
ARU_EN	5	rw	<p>GPR0 and GPR1 register values routed to ARU</p> <p>0_B Registers content not routed 1_B Registers content routed</p>
CICTRL	6	rw	<p>Channel Input Control</p> <p>0_B use signal TIM_IN(x) as input for channel x 1_B use signal TIM_IN(x-1) as input for channel x (or TIM_IN(7) if x is 0)</p>
Reserved	7	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
GPR0_SEL L	[9:8]	rw	Selection for GPR0 register If EGPR0_SEL = 0: 00 _B use TBU_TS0 as input 01 _B use TBU_TS1 as input 10 _B use TBU_TS2 as input 11 _B use CNTS as input; if TGPS mode in channel = 0 is selected use TIM Filter F_OUT as input If EGPR0_SEL = 1: 00 _B use ECNT as input 01 _B reserved 10 _B reserved 11 _B reserved <i>Note: If a reserved value is written to the EGPR0_SEL, GPR0_SEL bit fields, the hardware will use TBU_TS0 input.</i>
GPR1_SEL L	[11:10]	rw	Selection for GPR1 register If EGPR0_SEL = 0: 00 _B use TBU_TS0 as input 01 _B use TBU_TS1 as input 10 _B use TBU_TS2 as input 11 _B use CNT as input If EGPR0_SEL = 1: 00 _B use ECNT as input 01 _B reserved 10 _B reserved 11 _B reserved <i>Note: In TBCM mode EGPR1_SEL, GPR1_SEL are ignored TIM Filter F_OUT is used as input.</i> <i>Note: If a reserved value is written to the EGPR1_SEL, GPR1_SEL bit fields, the hardware will use TBU_TS0 input.</i>
CNTS_SEL L	12	rw	Selection for CNTS register 0 _B use CNT register as input 1 _B use TBU_TS0 as input Note: The functionality of the CNTS_SEL is disabled in the modes TIPM and TBCM.

Generic Timer Module (GTM)

Field	Bits	Type	Description
DSL	13	rw	Signal level control 0 _B Measurement starts with falling edge (low level measurement) 1 _B Measurement starts with rising edge (high level measurement)
ISL	14	rw	Ignore signal level 0 _B use DSL bit for selecting active signal level 1 _B ignore DSL and treat both edges as active edge Note: This bit is only applicable in Input Event mode (TIEM)
ECNT_RE SET	15	rw	Enables resetting the ECNT counter in periodic sampling mode 0 _B ECNT counter operating in wrap around mode 1 _B ECNT counter is reset with periodic sampling
FLT_EN	16	rw	Filter enable for channel x (x:0...7) 0 _B Filter disabled and internal states are reset 1 _B Filter enabled Note: If the filter is disabled all filter related units (including CSU) are bypassed, which means that the signal F_IN is directly routed to signal F_OUT.
FLT_CNT_FRQ	[18:17]	rw	Filter counter frequency select 00 _B FLT_CNT counts with CMU_CLK0 01 _B FLT_CNT counts with CMU_CLK1 10 _B FLT_CNT counts with CMU_CLK6 11 _B FLT_CNT counts with CMU_CLK7
EXT_CAP_EN	19	rw	Enables external capture mode The selected TIM mode is only sensitive to external capture pulses the input event changes are ignored 0 _B External capture disabled 1 _B External capture enabled
FLT_MOD E_RE	20	rw	Filter mode for rising edge 0 _B Immediate edge propagation mode 1 _B individual de-glitch mode
FLT_CTR_RE	21	rw	Filter counter mode for rising edge 0 _B Up/Down Counter 1 _B Hold Counter Note: This bit is only applicable in Individual Deglitch Time Mode

Generic Timer Module (GTM)

Field	Bits	Type	Description
FLT_MOD_E_FE	22	rw	Filter mode for falling edge 0 _B Immediate edge propagation mode 1 _B individual de-glitch mode
FLT_CTR_FE	23	rw	Filter counter mode for falling edge 0 _B Up/Down Counter 1 _B Hold Counter Note: This bit is only applicable in Individual Deglitch Time Mode
CLK_SEL	[26:24]	rw	CMU clock source select for channel 000 _B CMU_CLK0 selected 001 _B CMU_CLK1 selected 010 _B CMU_CLK2 selected 011 _B CMU_CLK3 selected 100 _B CMU_CLK4 selected 101 _B CMU_CLK5 selected 110 _B CMU_CLK6 selected 111 _B CMU_CLK7 selected
FR_ECNT_OFL	27	rw	Extended Edge counter overflow behaviour 0 _B Overflow will be signalled on ECNT bit width = 8 1 _B Overflow will be signalled on EECNT bit width (full range)
EGPR0_SEL	28	rw	Extension of GPR0_SEL bit field Details described in GPR0_SEL bit field.
EGPR1_SEL	29	rw	Extension of GPR1_SEL bit field Details described in GPR1_SEL bit field.
TOCTRL	[31:30]	rw	Timeout control 00 _B Timeout feature disabled 01 _B Timeout feature enabled for rising edge only 10 _B Timeout feature enabled for falling edge only 11 _B Timeout feature enabled for both edges

Generic Timer Module (GTM)

26.10.8.2 Register TIM0_CHx_CTRL (x:0...7)

GTM_TIM0_CHx_CTRL (x=0-7)

TIM Channel x Control Register (01024_H+x*80_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOCTRL		EGP R1_ SEL	EGP R0_ SEL	FR_ ECN T_O FL	CLK_SEL			FLT_ CTR _FE	FLT_ MOD E_F E	FLT_ CTR _RE	FLT_ MOD E_R E	Rese rved	FLT_CNT_ FRQ	FLT_ EN	
rw		rw	rw	rw	rw			rw	rw	rw	rw	r	rw	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECN T_R ESE T	ISL	DSL	CNT S_S EL	GPR1_ SE L	GPR0_ SE L	TBU 0_ SE L	CICT RL	ARU _EN	OSM	TIM_MODE				TIM_ EN	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw				rwh

Field	Bits	Type	Description
TIM_EN	0	rwh	<p>TIM channel x (x:0...7)enable</p> <p>0_B Channel disabled 1_B Channel enabled</p> <p>Note: Enabling of the channel resets the registers ECNT, TIMi_CHx_CNT, TIMi_CHx_GPR0, and TIMi_CHx_GPR1 to their reset values.</p> <p>Note: After finishing the action in one-shot mode the TIM_EN bit is cleared automatically. Otherwise, the bit must be cleared manually.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
TIM_MODE	[3:1]	rw	<p>TIM channel x (x:0...7) mode</p> <p>000_B PWM Measurement Mode (TPWM) 001_B Pulse Integration Mode (TPIM) 010_B Input Event Mode (TIEM) 011_B Input Prescaler Mode (TIPM) 100_B Bit Compression Mode (TBCM) 101_B Gated Periodic Sampling Mode (TGPS)</p> <p>Note: The Bit Compression Mode is only available in TIM channel 0. If this mode is selected in any other channels, the TIM_MODE = 000 (TPWM mode) is used instead. However, the register TIM_MODE is read with value 100.</p> <p>Note: If an undefined value is written to the TIM_MODE register, the hardware switches automatically to TIM_MODE = 000 (TPWM mode).</p> <p>Note: The TIM_MODE register should not be changed while the TIM channel is enabled.</p> <p>Note: If TIM channel is enabled and operating in TPWM or TPIM mode after the first valid edge defined by DSL has occurred, a reconfiguration of DSL, ISL, and TIM_MODE will not change the channel behavior. Reading these bit fields after reconfiguration will show the newly configured settings but the initial channel behavior will not change. Only a disabling of the TIM_EN=0 will change the channel operation mode.</p>
OSM	4	rw	<p>One-shot mode</p> <p>0_B Continuous operation mode 1_B One-shot mode</p> <p>Note: After finishing the action in one-shot mode the TIM_EN bit is cleared automatically.</p>
ARU_EN	5	rw	<p>GPR0 and GPR1 register values routed to ARU</p> <p>0_B Registers content not routed 1_B Registers content routed</p>
CICTRL	6	rw	<p>Channel Input Control</p> <p>0_B use signal TIM_IN(x) as input for channel x 1_B use signal TIM_IN(x-1) as input for channel x (or TIM_IN(7) if x is 0)</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
TBU0_SE L	7	rw	TBU_TS0 bits input select for TIM_CH[x]_GPRz (z: 0, 1) 0 _B Use TBU_TS0(23...0) to store in TIM0_CH[x]_GPR0 / TIM0_CH[x]_GPR1 1 _B Use TBU_TS0(26...3) to store in TIM0_CH[x]_GPR0 / TIM0_CH[x]_GPR1 Note: This bit is only applicable for TIM0
GPR0_SE L	[9:8]	rw	Selection for GPR0 register If EGPR0_SEL =0: 00 _B use TBU_TS0 as input 01 _B use TBU_TS1 as input 10 _B use TBU_TS2 as input 11 _B use CNTS as input; if TGPS mode in channel = 0 is selected use TIM Filter F_OUT as input If EGPR0_SEL =1: 00 _B use ECNT as input 01 _B reserved 10 _B reserved 11 _B reserved Note: If a reserved value is written to the EGPR0_SEL, GPR0_SEL bit fields, the hardware will use TBU_TS0 input.
GPR1_SE L	[11:10]	rw	Selection for GPR1 register If EGPR0_SEL =0: 00 _B use TBU_TS0 as input 01 _B use TBU_TS1 as input 10 _B use TBU_TS2 as input 11 _B use CNT as input If EGPR0_SEL =1: 00 _B use ECNT as input 01 _B reserved 10 _B reserved 11 _B reserved Note: In TBCM mode EGPR1_SEL, GPR1_SEL are ignored TIM Filter F_OUT is used as input. Note: If a reserved value is written to the EGPR1_SEL, GPR1_SEL bit fields, the hardware will use TBU_TS0 input.

Generic Timer Module (GTM)

Field	Bits	Type	Description
CNTS_SEL	12	rw	Selection for CNTS register 0 _B use CNT register as input 1 _B use TBU_TS0 as input Note: The functionality of the CNTS_SEL is disabled in the modes TIPM and TBCM.
DSL	13	rw	Signal level control 0 _B Measurement starts with falling edge (low level measurement) 1 _B Measurement starts with rising edge (high level measurement)
ISL	14	rw	Ignore signal level 0 _B use DSL bit for selecting active signal level 1 _B ignore DSL and treat both edges as active edge Note: This bit is only applicable in Input Event mode (TIEM and TIPM)
ECNT_RESET	15	rw	Enables resetting the ECNT counter in periodic sampling mode 0 _B ECNT counter operating in wrap around mode 1 _B ECNT counter is reset with periodic sampling
FLT_EN	16	rw	Filter enable for channel x (x:0...7) 0 _B Filter disabled and internal states are reset 1 _B Filter enabled Note: If the filter is disabled all filter related units (including CSU) are bypassed, which means that the signal F_IN is directly routed to signal F_OUT.
FLT_CNT_FRQ	[18:17]	rw	Filter counter frequency select 00 _B FLT_CNT counts with CMU_CLK0 01 _B FLT_CNT counts with CMU_CLK1 10 _B FLT_CNT counts with CMU_CLK6 11 _B FLT_CNT counts with CMU_CLK7
EXT_CAP_EN	19	rw	Enables external capture mode The selected TIM mode is only sensitive to external capture pulses the input event changes are ignored 0 _B External capture disabled 1 _B External capture enabled
FLT_MOD_E_RE	20	rw	Filter mode for rising edge 0 _B Immediate edge propagation mode 1 _B individual de-glitch mode

Generic Timer Module (GTM)

Field	Bits	Type	Description
FLT_CTR_RE	21	rw	Filter counter mode for rising edge 0 _B Up/Down Counter 1 _B Hold Counter Note: This bit is only applicable in Individual Deglitch Time Mode
FLT_MOD_E_FE	22	rw	Filter mode for falling edge 0 _B Immediate edge propagation mode 1 _B individual de-glitch mode
FLT_CTR_FE	23	rw	Filter counter mode for falling edge 0 _B Up/Down Counter 1 _B Hold Counter Note: This bit is only applicable in Individual Deglitch Time Mode
CLK_SEL	[26:24]	rw	CMU clock source select for channel 000 _B CMU_CLK0 selected 001 _B CMU_CLK1 selected 010 _B CMU_CLK2 selected 011 _B CMU_CLK3 selected 100 _B CMU_CLK4 selected 101 _B CMU_CLK5 selected 110 _B CMU_CLK6 selected 111 _B CMU_CLK7 selected
FR_ECNT_OFL	27	rw	Extended Edge counter overflow behaviour 0 _B Overflow will be signalled on ECNT bit width = 8 1 _B Overflow will be signalled on EECNT bit width (full range)
EGPR0_SEL	28	rw	Extension of GPR0_SEL bit field Details described in GPR0_SEL bit field.
EGPR1_SEL	29	rw	Extension of GPR1_SEL bit field Details described in GPR1_SEL bit field.
TOCTRL	[31:30]	rw	Timeout control 00 _B Timeout feature disabled 01 _B Timeout feature enabled for both edges 10 _B Timeout feature enabled for rising edge only 11 _B Timeout feature enabled for falling edge only

Generic Timer Module (GTM)

26.10.8.3 Register TIMi_CHx_FLT_RE (i:0...2)(x:0...7)

GTM_TIM0_CHx_FLT_RE (x=0-7)

GTM_TIM0 Channel x Filter Parameter 0 Register
(0101C_H+x*80_H)

Reset Value: 00000000_H

GTM_TIM1_CHx_FLT_RE (x=0-7)

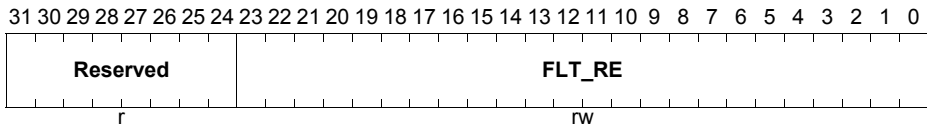
GTM_TIM1 Channel x Filter Parameter 0 Register
(0181C_H+x*80_H)

Reset Value: 00000000_H

GTM_TIM2_CHx_FLT_RE (x=0-7)

GTM_TIM2 Channel x Filter Parameter 0 Register
(0201C_H+x*80_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
FLT_RE	[23:0]	rw	Filter parameter for rising edge Note: This register has different meanings in the various filter modes. Immediate edge propagation mode = acceptance time for rising edge Individual deglitch time mode = deglitch time for rising edge
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

26.10.8.4 Register TIMi_CHx_FLT_FE (i:0...2)(x:0...7)

GTM_TIM0_CHx_FLT_FE (x=0-7)

TIM0 Channel x Filter Parameter 1 Register

(01020_H+x*80_H)

Reset Value: 00000000_H

GTM_TIM1_CHx_FLT_FE (x=0-7)

TIM1 Channel x Filter Parameter 1 Register

(01820_H+x*80_H)

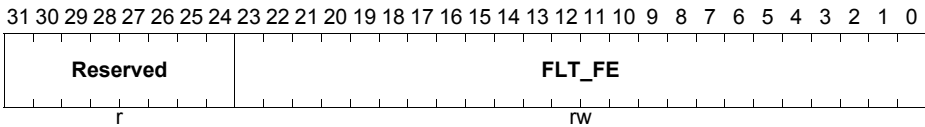
Reset Value: 00000000_H

GTM_TIM2_CHx_FLT_FE (x=0-7)

TIM2 Channel x Filter Parameter 1 Register

(02020_H+x*80_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
FLT_FE	[23:0]	rw	Filter parameter for falling edge Note: This register has different meanings in the various filter modes. Immediate edge propagation mode = acceptance time for falling edge Individual deglitch time mode = deglitch time for falling edge
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

26.10.8.5 Register TIMi_CHx_TDU (i:0...2)(x:0...7)

Functionality is replaced by register TIMi_CHx_TDUV, TIMi_CHx_TDUC and bit field TOCTRL in Register TIMi_CHx_CTRL

Generic Timer Module (GTM)

26.10.8.6 Register TIMi_CHx_GPR0 (i:0...2)(x:0...7)

GTM_TIM0_CHx_GPR0 (x=0-7)

TIM0 Channel x General Purpose 0 Register

(01000_H+x*80_H)

Reset Value: 0X000000_H

GTM_TIM1_CHx_GPR0 (x=0-7)

TIM1 Channel x General Purpose 0 Register

(01800_H+x*80_H)

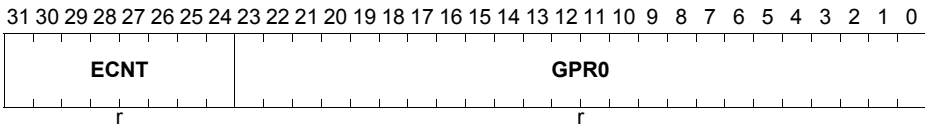
Reset Value: 0X000000_H

GTM_TIM2_CHx_GPR0 (x=0-7)

TIM2 Channel x General Purpose 0 Register

(02000_H+x*80_H)

Reset Value: 0X000000_H



Field	Bits	Type	Description
GPR0	[23:0]	r	<p>Input signal characteristic parameter 0</p> <p>Note: The content of this register has different meaning for the TIM channels modes. The content directly depends on the bit fields EGPR0_SEL, GPR0_SEL of register TIMi_CHx_CTRL.</p>
ECNT	[31:24]	r	<p>Edge counter</p> <p>Note: The ECNT counts every incoming filtered edge (rising and falling). The counter value is uneven in case of detected rising, and even in case of detected falling edge. Thus, the input signal level is part of the counter and can be obtained by bit 0 of ECNT.</p> <p>Note: The ECNT register is reset to its initial value when the channel is enabled. Please note, that bit 0 depends on the input level coming from the filter unit and defines the reset value immediately.</p>

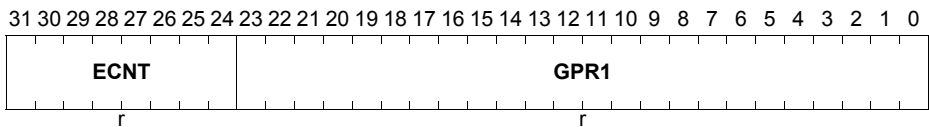
Generic Timer Module (GTM)

26.10.8.7 Register TIMi_CHx_GPR1 (i:0...2)(x:0...7)

GTM_TIM0_CHx_GPR1 (x=0-7)
 TIM0 Channel x General Purpose 1 Register
 (01004_H+x*80_H) Reset Value: 0X000000_H

GTM_TIM1_CHx_GPR1 (x=0-7)
 TIM1 Channel x General Purpose 1 Register
 (01804_H+x*80_H) Reset Value: 0X000000_H

GTM_TIM2_CHx_GPR1 (x=0-7)
 TIM2 Channel x General Purpose 1 Register
 (02004_H+x*80_H) Reset Value: 0X000000_H



Field	Bits	Type	Description
GPR1	[23:0]	r	<p>Input signal characteristic parameter 1</p> <p>Note: The content of this register has different meaning for the TIM channels modes. The content directly depends on the bit fields EGPR1_SEL, GPR1_SEL of register TIMi_CHx_CTRL.</p> <p>Note: In TBCM mode if EGPR1_SEL=1, GPR1_SEL=01 then TIM_INP_VAL is used as input in all other cases TIM filter F_OUT is used as input and bits GPR1(23:8)=0.</p> <p>Note: The content of this register can only be written in TIM channel mode TGPS.</p>
ECNT	[31:24]	r	<p>Edge counter</p> <p>Note: The ECNT counts every incoming filtered edge (rising and falling). The counter value is uneven in case of detected rising, and even in case of detected falling edge. Thus, the input signal level is part of the counter and can be obtained by bit 0 of ECNT.</p> <p>Note: The ECNT register is reset to its initial value when the channel is enabled. Please note, that bit 0 depends on the input level coming from the filter unit and defines the reset value immediately.</p>

Generic Timer Module (GTM)

26.10.8.8 Register TIMi_CHx_CNT (i:0...2)(x:0...7)

GTM_TIM0_CHx_CNT (x=0-7)

TIM0 Channel x SMU Counter Register

(01008_H+x*80_H)

Reset Value: 00000000_H

GTM_TIM1_CHx_CNT (x=0-7)

TIM1 Channel x SMU Counter Register

(01808_H+x*80_H)

Reset Value: 00000000_H

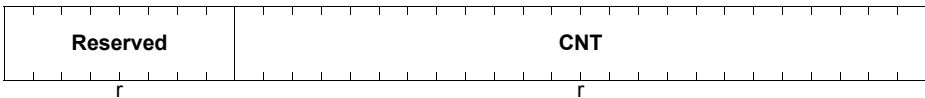
GTM_TIM2_CHx_CNT (x=0-7)

TIM2 Channel x SMU Counter Register

(02008_H+x*80_H)

Reset Value: 00000000_H

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0



Field	Bits	Type	Description
CNT	[23:0]	r	Actual SMU counter value Note: The meaning of this value depends on the configured mode: TPWM = actual duration of PWM signal. TPIM = actual duration of all pulses (sum of pulses). TIEM = actual number of received edges. TIPM = actual number of received edges.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.10.8.9 Register TIMi_CHx_CNTS (i:0...2)(x:0...7)

GTM_TIM0_CHx_CNTS (x=0-7)

TIM0 Channel x SMU Shadow Counter Register

(01010_H+x*80_H)

Reset Value: 0X000000_H

GTM_TIM1_CHx_CNTS (x=0-7)

TIM1 Channel x SMU Shadow Counter Register

(01810_H+x*80_H)

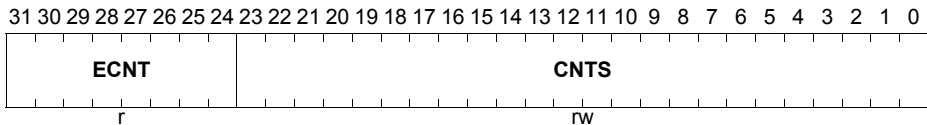
Reset Value: 0X000000_H

GTM_TIM2_CHx_CNTS (x=0-7)

TIM2 Channel x SMU Shadow Counter Register

(02010_H+x*80_H)

Reset Value: 0X000000_H



Field	Bits	Type	Description
CNTS	[23:0]	rw	<p>Counter shadow register</p> <p>Note: The content of this register has different meaning for the TIM channels modes. The content depends directly on the bit field CNTS_SEL of register TIMi_CHx_CTRL.</p> <p>Note: The register TIMi_CHx_CNTS is only writable in TIPM, TBCM, and TGPS mode.</p>
ECNT	[31:24]	r	<p>Edge counter</p> <p>Note: The ECNT counts every incoming filtered edge (rising and falling). The counter value is uneven in case of detected rising, and even in case of detected falling edge. Thus, the input signal level is part of the counter and can be obtained by bit 0 of ECNT.</p> <p>Note: The ECNT register is reset to its initial value when the channel is enabled. Please note, that bit 0 depends on the input level coming from the filter unit and defines the reset value immediately.</p>

Generic Timer Module (GTM)

26.10.8.10 Register TIM_i_CH_x_IRQ_NOTIFY (i:0...2)(x:0...7)
GTM_TIM0_CH_x_IRQ_NOTIFY (x=0-7)
TIM0 Channel x Interrupt Notification Register

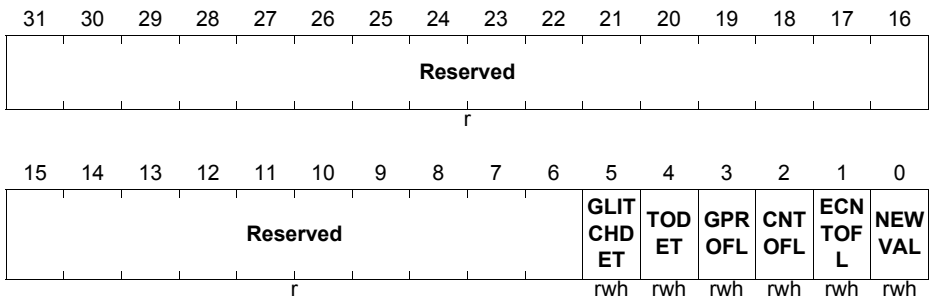
 (0102C_H+x*80_H)

Reset Value: 00000000_H
GTM_TIM1_CH_x_IRQ_NOTIFY (x=0-7)
TIM1 Channel x Interrupt Notification Register

 (0182C_H+x*80_H)

Reset Value: 00000000_H
GTM_TIM2_CH_x_IRQ_NOTIFY (x=0-7)
TIM2 Channel x Interrupt Notification Register

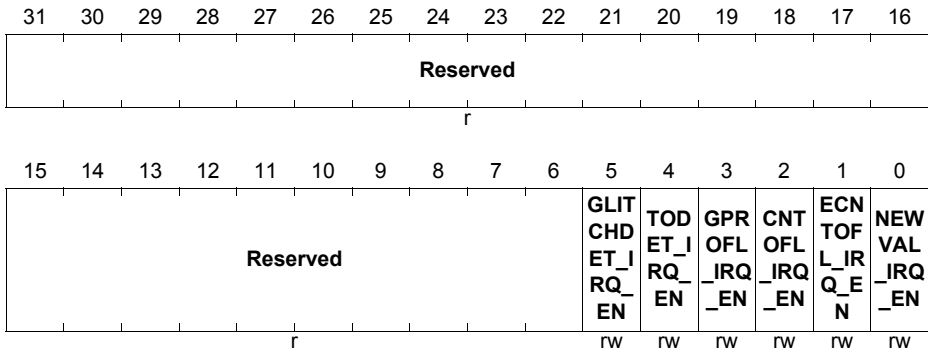
 (0202C_H+x*80_H)

Reset Value: 00000000_H


Field	Bits	Type	Description
NEWVAL	0	rwh	New measurement value detected by in channel x (x:0...7) 0 _B No event was occurred 1 _B NEWVAL was occurred on the TIM channel Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ECNTOFL	1	rwh	counter overflow of channel x, (x:0...7) See bit 0
CNTOFL	2	rwh	SMU CNT counter overflow of channel x, (x:0...7) See bit 0
GPROFL	3	rwh	GPR0 and GPR1 data overflow, old data not read out before new data has arrived at input pin, (x:0...7) See bit 0
TODET	4	rwh	Timeout reached for input signal of channel x, (x:0...7) see bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
GLITCHDET	5	rwh	Glitch detected on channel x, (x:0...7) 0_B no glitch detected for last edge 1_B glitch detected for last edge Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
Reserved	[31:6]	r	Reserved Read as zero, should be written as zero

26.10.8.11 Register TIMi_CHx_IRQ_EN (i:0...2)(x:0...7)
GTM_TIM0_CHx_IRQ_EN (x=0-7)
TIM0 Channel x Interrupt Enable Register
 $(01030_H + x * 80_H)$
Reset Value: 00000000_H
GTM_TIM1_CHx_IRQ_EN (x=0-7)
TIM1 Channel x Interrupt Enable Register
 $(01830_H + x * 80_H)$
Reset Value: 00000000_H
GTM_TIM2_CHx_IRQ_EN (x=0-7)
TIM2 Channel x Interrupt Enable Register
 $(02030_H + x * 80_H)$
Reset Value: 00000000_H


Field	Bits	Type	Description
NEWVAL_IRQ_EN	0	rw	TIM_NEWVALx_IRQ interrupt enable 0_B Disable interrupt, interrupt is not visible outside GTM 1_B Enable interrupt, interrupt is visible outside GTM

Generic Timer Module (GTM)

Field	Bits	Type	Description
ECNTOFL_IRQ_EN	1	rw	TIM_ECNTOFLx_IRQ interrupt enable see bit 0
CNTOFL_IRQ_EN	2	rw	TIM_CNTOFLx_IRQ interrupt enable see bit 0
GPROFL_IRQ_EN	3	rw	TIM_GPROFLx_IRQ interrupt enable see bit 0
TODET_IRQ_EN	4	rw	TIM_TODETx_IRQ interrupt enable see bit 0
GLITCHDET_IRQ_EN	5	rw	TIM_GLITCHDETx_IRQ interrupt enable see bit 0
Reserved	[31:6]	r	Reserved Read as zero, should be written as zero

26.10.8.12 Register TIMi_CHx_IRQ_FORCINT (i:0...2)(x:0...7)

GTM_TIM0_CHx_IRQ_FORCINT (x=0-7)

TIM0 Channel x Software Interrupt Force Register

(01034_H+x*80_H)

Reset Value: 00000000_H

GTM_TIM1_CHx_IRQ_FORCINT (x=0-7)

TIM1 Channel x Software Interrupt Force Register

(01834_H+x*80_H)

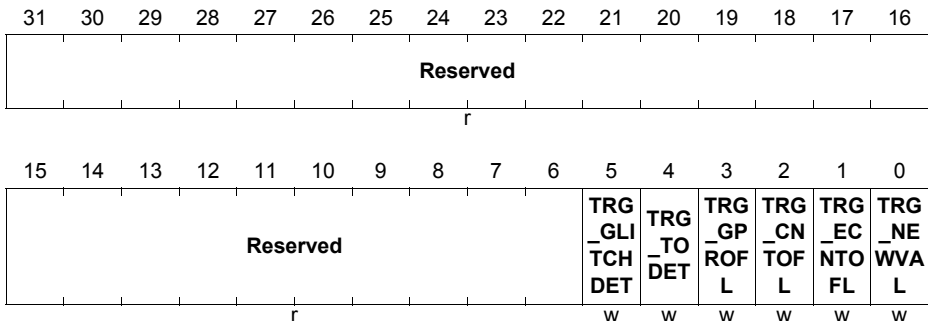
Reset Value: 00000000_H

GTM_TIM2_CHx_IRQ_FORCINT (x=0-7)

TIM2 Channel x Software Interrupt Force Register

(02034_H+x*80_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG_NEWVAL	0	w	Trigger NEWVAL bit in TIM_CHx_IRQ_NOTIFY register by software 0 _B No interrupt triggering 1 _B Assert corresponding field in TIMi_CHx_IRQ_NOTIFY register Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
TRG_ECNTOFL	1	w	Trigger ECNTOFL bit in TIM_CHx_IRQ_NOTIFY register by software see bit 0
TRG_CNTOFL	2	w	Trigger CNTOFL bit in TIM_CHx_IRQ_NOTIFY register by software see bit 0
TRG_GPROFL	3	w	Trigger GPROFL bit in TIM_CHx_IRQ_NOTIFY register by software see bit 0
TRG_TODET	4	w	Trigger TODET bit in TIM_CHx_IRQ_NOTIFY register by software see bit 0
TRG_GLITCHDET	5	w	Trigger GLITCHDET bit in TIM_CHx_IRQ_NOTIFY register by software see bit 0
Reserved	[31:6]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.10.8.13 Register TIM_i_CH_x_IRQ_MODE (i:0...2)(x:0...7)

GTM_TIM0_CH_x_IRQ_MODE (x=0-7)

TIM0 IRQ Mode Configuration Register

(01038_H+x*80_H)

Reset Value: 00000000_H

GTM_TIM1_CH_x_IRQ_MODE (x=0-7)

TIM1 IRQ Mode Configuration Register

(01838_H+x*80_H)

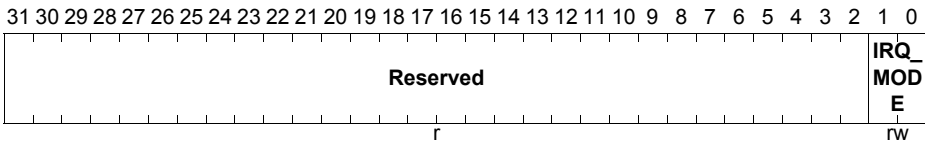
Reset Value: 00000000_H

GTM_TIM2_CH_x_IRQ_MODE (x=0-7)

TIM2 IRQ Mode Configuration Register

(02038_H+x*80_H)

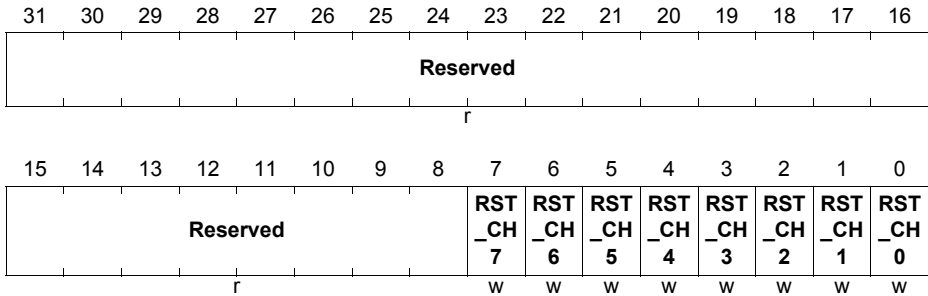
Reset Value: 00000000_H



Field	Bits	Type	Description
IRQ_MODE	[1:0]	rw	IRQ mode selection 00 _B Level mode 01 _B Pulse mode 10 _B Pulse-Notify mode 11 _B Single-Pulse mode Note: The interrupt modes are described in Section 26.2.5 .
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

26.10.8.14 Register TIMi_RST (i:0..2)

GTM_TIMi_RST (i=0-2)

 TIMi Global Software Reset Register(0107C_H+i*800_H) Reset Value: 00000000_H


Field	Bits	Type	Description
RST_CH0	0	w	Software reset of channel 0 0 _B No action 1 _B Reset channel 0 Note: This bit is cleared automatically after write by CPU. The channel registers are set to their reset values and channel operation is stopped immediately.
RST_CH1	1	w	Software reset of channel 1 see bit 0
RST_CH2	2	w	Software reset of channel 2 see bit 0
RST_CH3	3	w	Software reset of channel 3 see bit 0
RST_CH4	4	w	Software reset of channel 4 see bit 0
RST_CH5	5	w	Software reset of channel 5 see bit 0
RST_CH6	6	w	Software reset of channel 6 see bit 0
RST_CH7	7	w	Software reset of channel 7 see bit 0
Reserved	[31:8]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.10.8.15 Register TIMi_IN_SRC (i:0...2)

GTM_TIMi_IN_SRC (i=0-2)

TIMi_IN_SRC Long Name (01078_H+i*800_H) Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODE_7		VAL_7		MODE_6		VAL_6		MODE_5		VAL_5		MODE_4		VAL_4	
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE_3		VAL_3		MODE_2		VAL_2		MODE_1		VAL_1		MODE_0		VAL_0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
VAL_0	[1:0]	rw	<p>Value to be fed to Channel 0</p> <p>multicore encoding in use(VAL_x(1) defines the state of the signal)</p> <p>00_B State is 0 (ignore write access)</p> <p>01_B Change state to 0</p> <p>01_B Change state to 1</p> <p>01_B State is 1 (ignore write access)</p> <p>function depends on combination of VAL_x(1) and MODE_x(1), see MODE_0 description.</p> <p><i>Note: Any read access to a VAL_x, bit field will always result in a value 00 or 11 indicating current state. A modification of the state is only performed with the values 01 and 10. Writing the values 00 and 11 is always ignored.</i></p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
MODE_0	[3:2]	rw	<p>Input source to Channel 0 multicore encoding in use(MODE_x(1) defines the state of the signal) 00_B State is 0 (ignore write access) 01_B Change state to 0 10_B Change state to 1 (VAL_0 is connected to channel) 11_B State is 1 (ignore write access) Function table: MODE_x(1)=0, VAL_x(1)=0: The input signal defined by bit field CICTRL of the TIM channel is used as input source. MODE_x(1)=0, VAL_x(1)=1: The signal TIM_AUX_IN of the TIM channel is used as input source. MODE_x(1)=1: The state VAL_x(1) defines the input level for the TIM channel.</p> <p><i>Note: Note: Any read access to a MODE_x, bit field will always result in a value 00 or 11 indicating current state. A modification of the state is only performed with the values 01 and 10. Writing the values 00 and 11 is always ignored.</i></p>
VAL_1	[5:4]	rw	<p>Value to be fed to Channel 1 see bits 1:0</p>
MODE_1	[7:6]	rw	<p>Input source to Channel 1 see bits 3:2</p>
VAL_2	[9:8]	rw	<p>Value to be fed to Channel 2 see bits 1:0</p>
MODE_2	[11:10]	rw	<p>Input source to Channel 2 see bits 3:2</p>
VAL_3	[13:12]	rw	<p>Value to be fed to Channel 3 see bits 1:0</p>
MODE_3	[15:14]	rw	<p>Input source to Channel 3 see bits 3:2</p>
VAL_4	[17:16]	rw	<p>Value to be fed to Channel 4 see bits 1:0</p>
MODE_4	[19:18]	rw	<p>Input source to Channel 4 see bits 3:2</p>

Generic Timer Module (GTM)

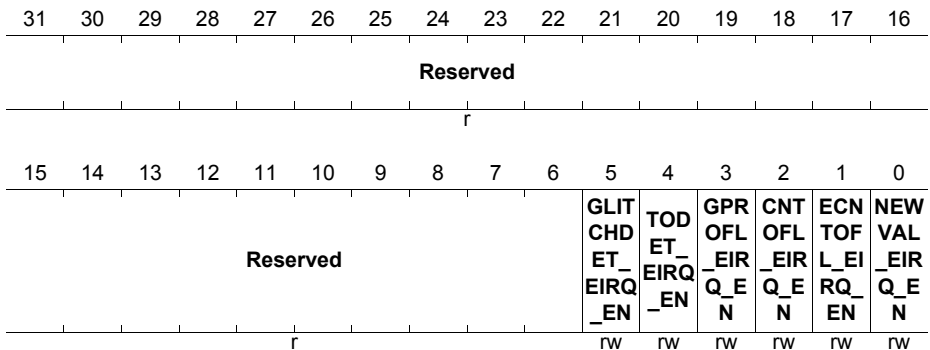
Field	Bits	Type	Description
VAL_5	[21:20]	rw	Value to be fed to Channel 5 see bits 1:0
MODE_5	[23:22]	rw	Input source to Channel 5 see bits 3:2
VAL_6	[25:24]	rw	Value to be fed to Channel 6 see bits 1:0
MODE_6	[27:26]	rw	Input source to Channel 6 see bits 3:2
VAL_7	[29:28]	rw	Value to be fed to Channel 7 see bits 1:0
MODE_7	[31:30]	rw	Input source to Channel 7 see bits 3:2

Generic Timer Module (GTM)

26.10.8.16 Register TIM_i_CH_x_EIRQ_EN (i:0...2)(x:0...7)
GTM_TIM0_CH_x_EIRQ_EN (x=0-7)
TIM0 Channel x Error Interrupt Enable Register
 (0103C_H+x*80_H)

Reset Value: 00000000_H
GTM_TIM1_CH_x_EIRQ_EN (x=0-7)
TIM1 Channel x Error Interrupt Enable Register
 (0183C_H+x*80_H)

Reset Value: 00000000_H
GTM_TIM2_CH_x_EIRQ_EN (x=0-7)
TIM2 Channel x Error Interrupt Enable Register
 (0203C_H+x*80_H)

Reset Value: 00000000_H


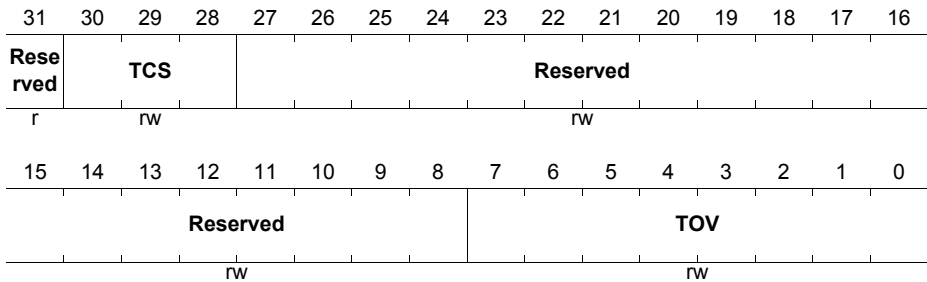
Field	Bits	Type	Description
NEWVAL_ EIRQ_EN	0	rw	TIM_NEWVAL_x_EIRQ error interrupt enable 0 _B Disable error interrupt, error interrupt is not visible outside GTM 1 _B Enable error interrupt, error interrupt is visible outside GTM
ECNTOFL_ _EIRQ_EN	1	rw	TIM_ECNTOFL_x_IRQ interrupt enable see bit 0
CNTOFL_ EIRQ_EN	2	rw	TIM_CNTOFL_x_IRQ interrupt enable see bit 0
GPROFL_ EIRQ_EN	3	rw	TIM_GPROFL_IRQ interrupt enable see bit 0
TODET_EI RQ_EN	4	rw	TIM_TODET_x_IRQ interrupt enable see bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
GLITCHDET_EIRQ_EN	5	rw	TIM_GLITCHDET_x_IRQ interrupt enable see bit 0
Reserved	[31:6]	r	Reserved Read as zero, should be written as zero

26.10.8.17 Register $TIM_i_CH_x_TDUV$ ($i:0\dots2$)($x:0\dots7$)

GTM_TIM0_CHx_TDUV ($x=0-7$)
 TIM0 Channel x TDUV Register (01018_H+x*80_H) Reset Value: 00000000_H
 GTM_TIM1_CHx_TDUV ($x=0-7$)
 TIM1 Channel x TDUV Register (01818_H+x*80_H) Reset Value: 00000000_H
 GTM_TIM2_CHx_TDUV ($x=0-7$)
 TIM2 Channel x TDUV Register (02018_H+x*80_H) Reset Value: 00000000_H

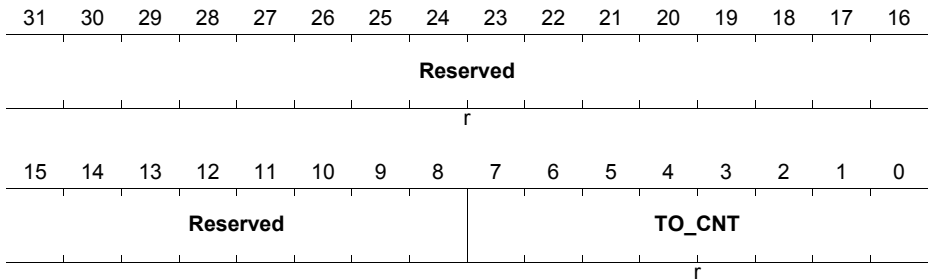


Field	Bits	Type	Description
TOV	[7:0]	rw	Time out duration for channel x
Reserved	[27:8]	rw	Reserved Read as zero, should be written as zero
TCS	[30:28]	rw	Timeout Clock selection 000 _B CMU_CLK0 selected 001 _B CMU_CLK1 selected 010 _B CMU_CLK2 selected 011 _B CMU_CLK3 selected 100 _B CMU_CLK4 selected 101 _B CMU_CLK5 selected 110 _B CMU_CLK6 selected 111 _B CMU_CLK7 selected
Reserved	31	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.10.8.18 Register TIM_i_CH_x_TDUC (i:0...2)(x:0...7)

GTM_TIM0_CH_x_TDUC (x=0-7)
 TIM0 Channel x TDUC Register (01014_H+x*80_H) Reset Value: 00000000_H
 GTM_TIM1_CH_x_TDUC (x=0-7)
 TIM1 Channel x TDUC Register (01814_H+x*80_H) Reset Value: 00000000_H
 GTM_TIM2_CH_x_TDUC (x=0-7)
 TIM2 Channel x TDUC Register (02014_H+x*80_H) Reset Value: 00000000_H



Field	Bits	Type	Description
TO_CNT	[7:0]	rh	Current Timeout value for channel x
Reserved	[31:8]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.10.8.19 Register TIM_i_CH_x_ECNT (i:0...2)(x:0...7)

GTM_TIM0_CH_x_ECNT (x=0-7)

TIM0 Channel x Edge Counter Register

(0100C_H+x*80_H)

Reset Value: 00000000_H

GTM_TIM1_CH_x_ECNT (x=0-7)

TIM1 Channel x Edge Counter Register

(0180C_H+x*80_H)

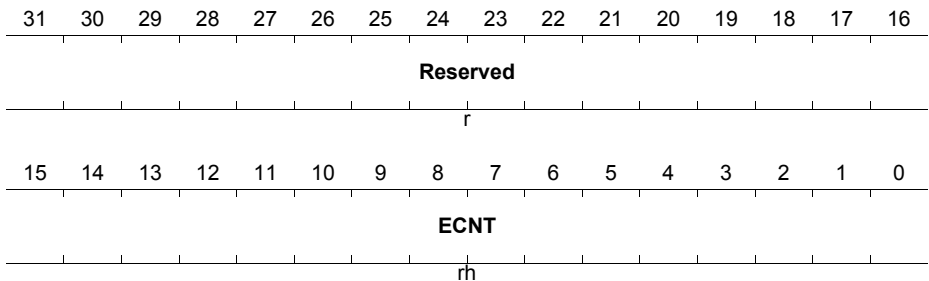
Reset Value: 00000000_H

GTM_TIM2_CH_x_ECNT (x=0-7)

TIM2 Channel x Edge Counter Register

(0200C_H+x*80_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
ECNT	[15:0]	rh	Edge counter Note: If TIM channel is disabled the content of ECNT gets frozen. A read will auto clear the bits [15:1]. Further read access to ECNT will show on Bit 0 the actual input signal value of the channel.
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.10.8.20 Register TIM_i_CH_x_ECTRL (i:0...2)(x:0...7)
GTM_TIM0_CH_x_ECTRL (x=0-7)

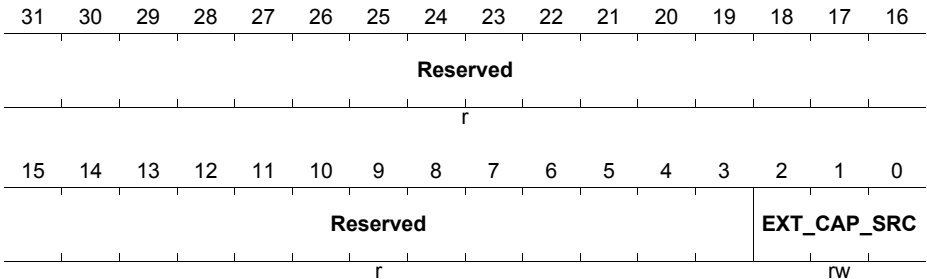
TIM0 Channel x External Capture

Control Register (01028_H+x*80_H)
Reset Value: 00000000_H
GTM_TIM1_CH_x_ECTRL (x=0-7)

TIM1 Channel x External Capture

Control Register (01828_H+x*80_H)
Reset Value: 00000000_H
GTM_TIM2_CH_x_ECTRL (x=0-7)

TIM2 Channel x External Capture

Control Register (02028_H+x*80_H)
Reset Value: 00000000_H


Field	Bits	Type	Description
EXT_CAP_SRC	[2:0]	rw	Defines selected source for triggering the EXT_CAPTURE functionality 000 _B NEW_VAL_IRQ of following channel selected 001 _B AUX_IN selected 010 _B CNTOFL_IRQ of following channel selected 011 _B and CICTRL=1: use signal TIM_IN(x) as input for channel x and CICTRL=0: use signal TIM_IN(x-1) as input for channel x (or TIM_IN(m-1) if x is 0) 100 _B ECNTOFL_IRQ of following channel selected 101 _B TODET_IRQ of following channel selected 110 _B GLITCHDET_IRQ of following channel selected 111 _B GPROFL_IRQ of following channel selected
Reserved	[31:3]	r	Reserved Read as zero, should be written as zero

26.11 Timer Output Module (TOM)

26.11.1 Overview

The Timer Output Module (TOM) offers 16 independent channels to generate simple PWM signals at each output pin TOM[i]_CH[x]_OUT (i=0...1; x=0...15).

Additionally, at TOM output TOM[i]_CH15_OUT a pulse count modulated signal can be generated.

The architecture of the TOM submodule is depicted in [Figure 26-32](#).

Some useful indices are defined as:

y=0,1

z=0...7

26.11.1.1 TOM Block diagram

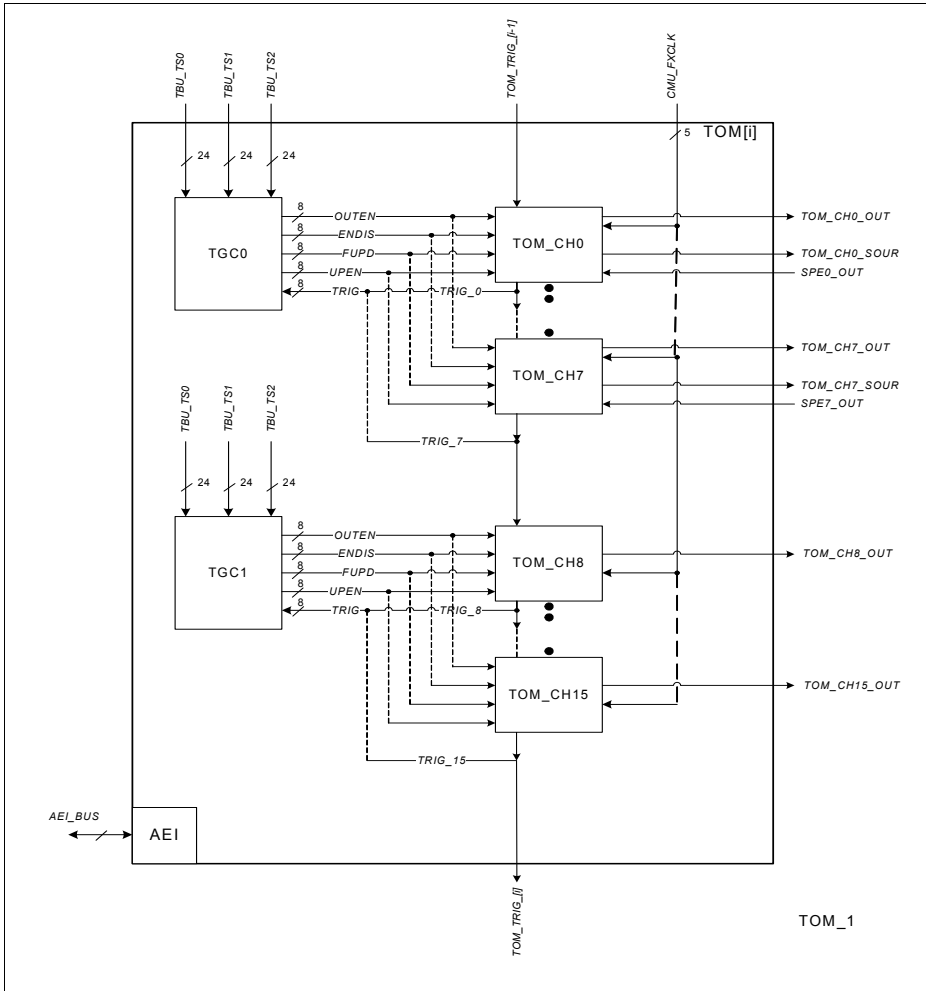


Figure 26-32 TOM Block diagram

The two submodules TGC0 and TGC1 are global channel control units that control the enabling/disabling of the channels and their outputs as well as the update of their period and duty cycle register.

Generic Timer Module (GTM)

The module TOM receives two (three) timestamp values TBU_TS0, TBU_TS1 (and TBU_TS2) in order to realize synchronized output behaviour on behalf of a common time base.

The 5 dedicated clock line inputs CMU_FXCLK are providing divided clocks that can be selected to clock the output pins.

26.11.2 TOM Global Channel Control (TGC0, TGC1)

26.11.2.1 Overview

There exist two global channel control units (TGC0 and TGC1) to drive a number of individual TOM channels synchronously by external or internal events.

Each TGC[y] can drive up to eight TOM channels where TGC0 controls TOM channels 0 to 7 and TGC1 controls TOM channels 8 to 15.

The TOM submodule supports four different kinds of signalling mechanisms:

- Global enable/disable mechanism for each TOM channel with control register TOMi_TGCy_ENDIS_CTRL and status register TOMi_TGCy_ENDIS_STAT
- Global output enable mechanism for each TOM channel with control register TOMi_TGCy_OUTEN_CTRL and status register TOMi_TGCy_OUTEN_STAT
- Global force update mechanism for each TOM channel with control register TOMi_TGCy_FUPD_CTRL
- Update enable of the register CM0, CM1 and CLK_SRC_STAT for each TOM channel with the control bit field UPEN_CTRL[z] of TOMi_TGCy_GLB_CTRL

26.11.2.2 TGC Subunit

Each of the first three individual mechanisms (enable/disable of the channel, output enable and force update) can be driven by three different trigger sources.

The three trigger sources are:

- the host CPU (bit HOST_TRIG of register TOMi_TGCy_GLB_CTRL)
- the TBU time stamp (signal TBU_TS0., TBU_TS1, TBU_TS2)
- the internal trigger signal TRIG (bunch of trigger signals TRIG_[x])

Note: The trigger signal is only active for one configured CMU clock period.

As a consequence, if the configured CMU clock period of the channel generating the trigger TRIG_[X] is smaller than the CMU clock period of triggered channel, the triggered channel may not recognize the trigger signal TRIG_[X].

Thus, it is recommended to configure all channels connected via the trigger TRIG_[x] to the same CMU clock period.

Generic Timer Module (GTM)

The first way is to trigger the control mechanism by a direct register write access via host CPU (bit HOST_TRIG of register TOMi_TGCy_GLB_CTRL).

The second way is provided by a compare match trigger on behalf of a specified time base coming from the module TBU (selected by bits TBU_SEL) and the time stamp compare value defined in the bit field ACT_TB of register TOMi_TGCy_ACT_TB.

Note, a signed compare of ACT_TB and selected TBU_TS[x] with x=0,1,2 is performed.

The third possibility is the input TRIG (bunch of trigger signals TRIG_[x]) coming from the TOM channels 0 to 7 / 8 to 15.

The corresponding trigger signal TRIG_[x] coming from channel [x] can be masked by the register TOMi_TGCy_INT_TRIG.

To enable or disable each individual TOM channel, the registers TOMi_TGCy_ENDIS_CTRL and/or TOMi_TGCy_ENDIS_STAT have to be used.

The register TOMi_TGCy_ENDIS_STAT controls directly the signal ENDIS. A write access to this register is possible.

The register TOMi_TGCy_ENDIS_CTRL is a shadow register that overwrites the value of register TOMi_TGCy_ENDIS_STAT if one of the three trigger conditions matches.

TOM Global channel control mechanism

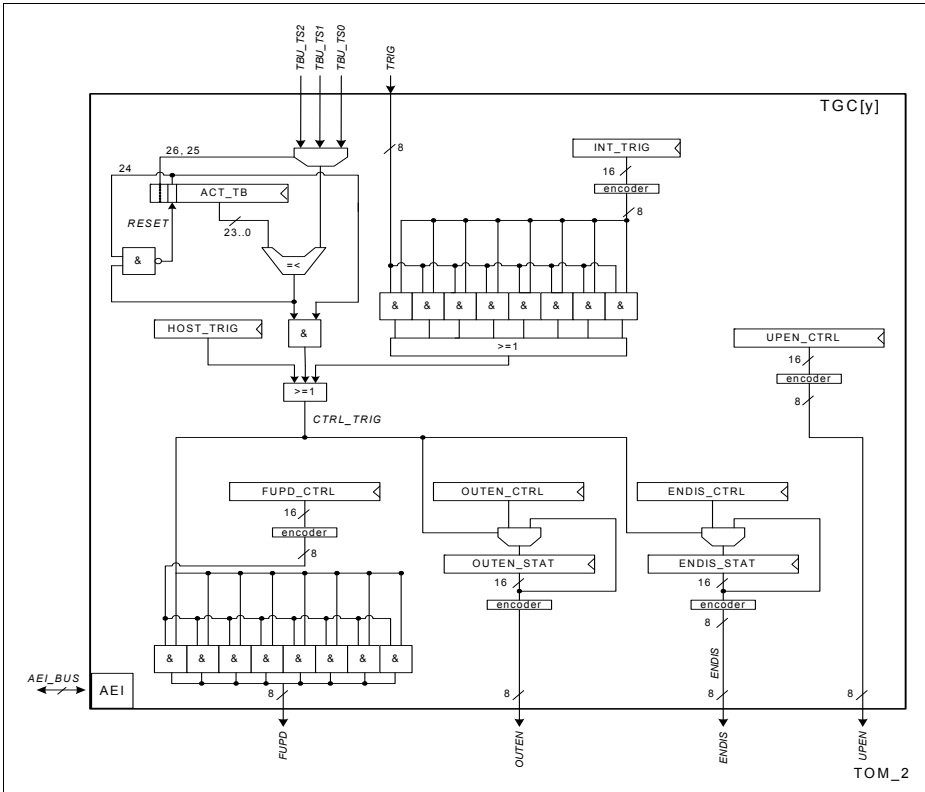


Figure 26-33 TOM Global channel control mechanism

The output of the individual TOM channels can be controlled using the register TOMi_TGCy_OUTEN_CTRL and TOMi_TGCy_OUTEN_STAT.

The register TOMi_TGCy_OUTEN_STAT controls directly the signal OUTEN. A write access to this register is possible.

The register TOMi_TGCy_OUTEN_CTRL is a shadow register that overwrites the value of register TOMi_TGCy_OUTEN_STAT if one of the three trigger conditions matches.

If a TOM channel is disabled by the register TOMi_TGCy_OUTEN_STAT, the actual value of the channel output at TOM_CH[x]_OUT is defined by the signal level bit (SL) defined in the channel control register TOMi_CHx_CTRL.

If the output is enabled, the output at TOM_CH[x]_OUT depends on value of FlipFlop SOUR.

Generic Timer Module (GTM)

The register TOMi_TGCy_FUPD_CTRL defines which of the TOM channels receive a FORCE_UPDATE event if the trigger signal CTRL_TRIG is raised.

The register bits UPEN_CTRL[z] defines for which TOM channel the update of the working register CM0, CM1 and CLK_SRC by the corresponding shadow register SR0, SR1 and CLK_SRC_SR is enabled. If update is enabled, the register CM0, CM1 and CLK_SRC will be updated on reset of counter register CN0 (see [Figure 26-34](#) and [Figure 26-35](#)).

26.11.3 TOM Channel (TOM_CH[x])

Each individual TOM channel comprises a Counter Compare Unit 0 (CCU0), a Counter Compare Unit 1 (CCU1) and the Signal Output Generation Unit (SOU). The architecture is depicted in [Figure 26-34](#) for channels 0 to 7 and in [Figure 26-35](#) for channels 8 to 15.

26.11.3.1 TOM Channel 0...7 architecture

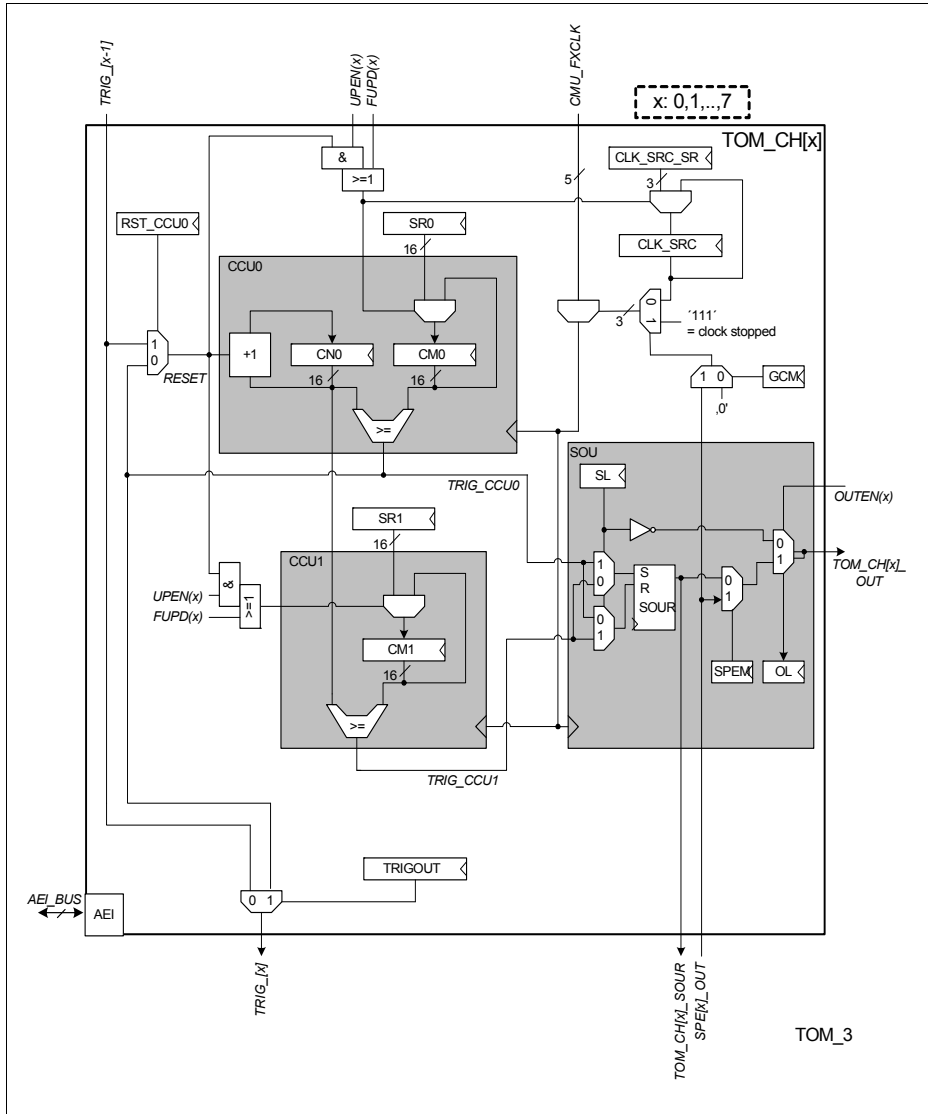


Figure 26-34 TOM Channel 0...7 architecture

26.11.3.2 TOM Channel 8...14 architecture

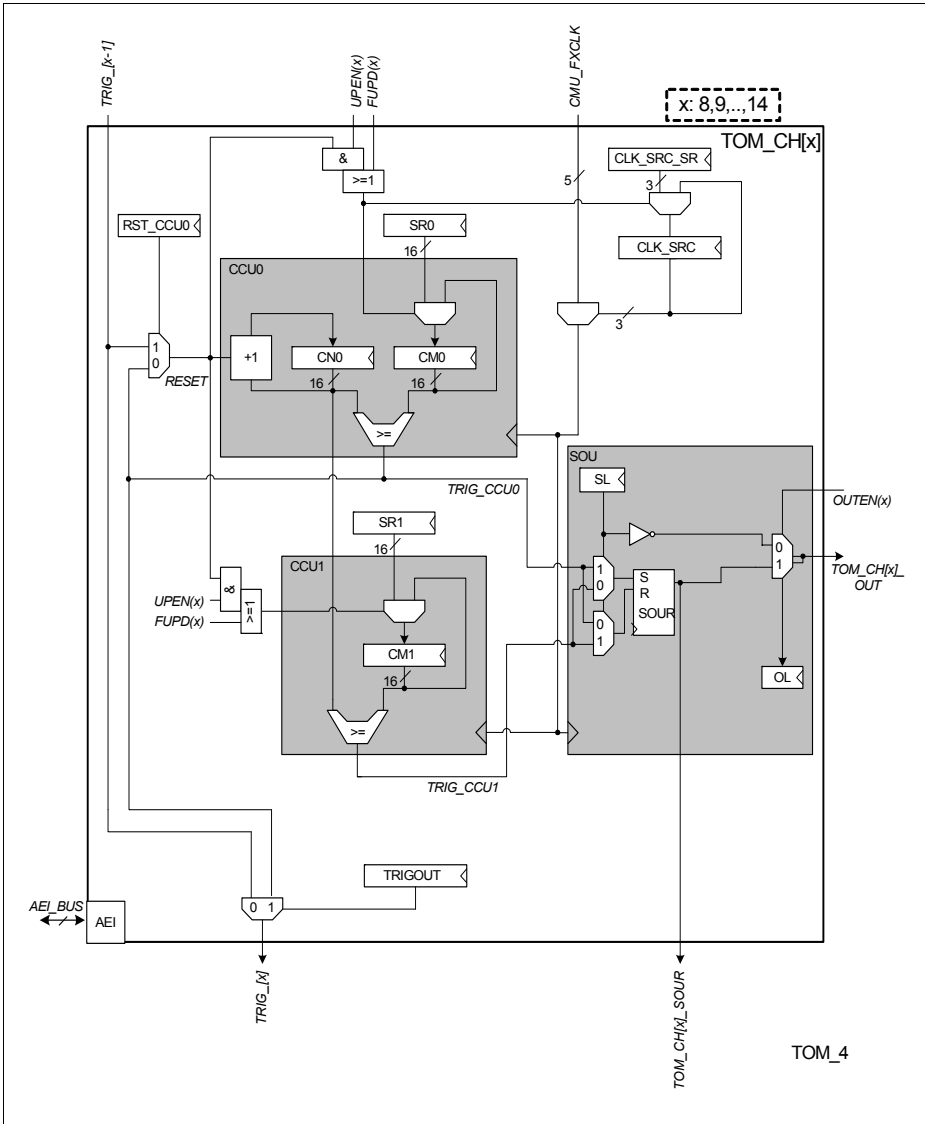


Figure 26-35 TOM Channel 8...14 architecture

26.11.3.3 TOM Channel 15 architecture for PCM generation

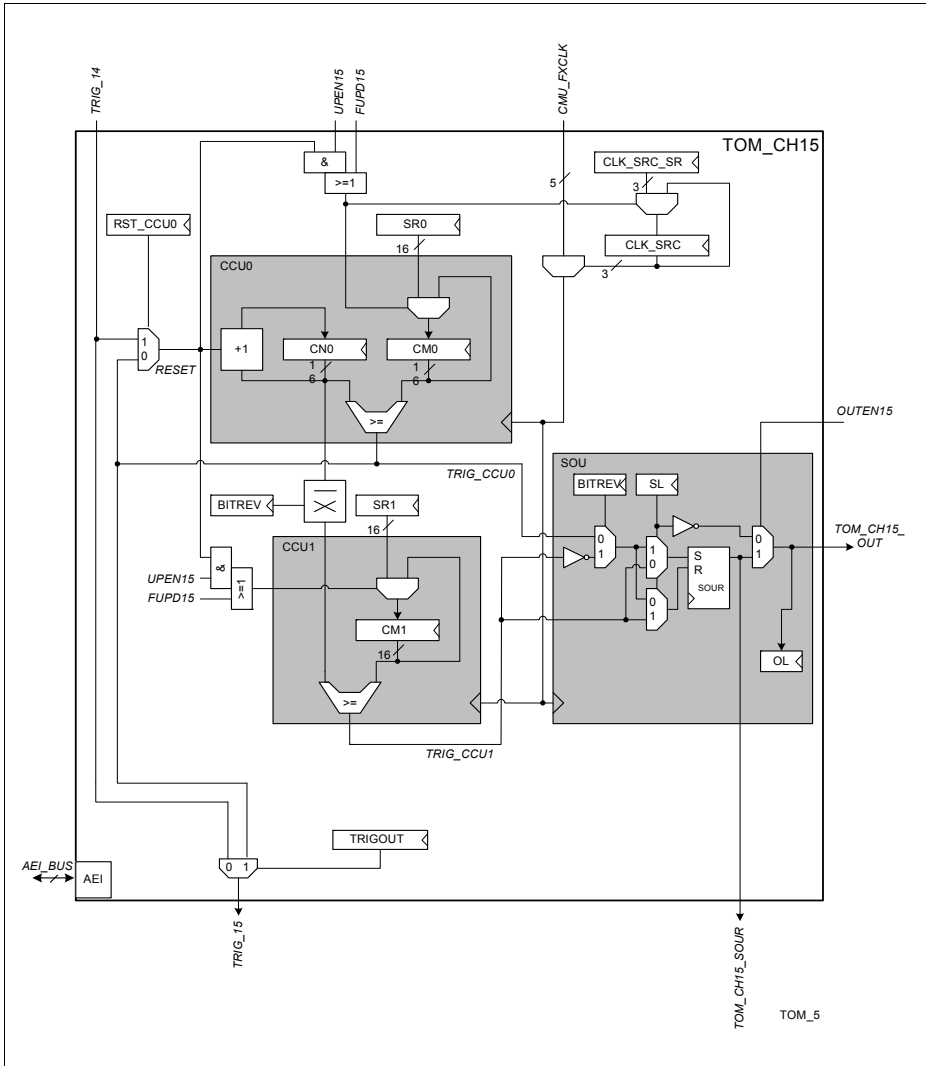


Figure 26-36 TOM Channel 15 architecture for PCM generation

Generic Timer Module (GTM)

The CCU0 contains a counter CN0 which is clocked with one of the selected input frequencies (CMU_FXCLK) provided from outside of the submodule.

Depending on configuration bits RST_CCU0 of register TOMi_CHx_CTRL the counter can be reset either when the counter value is equal to the compare value CM0 or when signalled by the TOM[i] trigger signal TRIG_[x-1] of the preceding channel [x-1] (which can also be the last channel of preceding instance TOM[i-1]).

When the counter register CN0 is greater or equal than the register CM0, the subunit CCU0 triggers the SOU subunit and the succeeding TOM submodule channel (signal TRIG_CCU0).

In the subunit CCU1 the counter register CN0 is compared with the value of register CM1. If CN0 is greater or equal than CM1 the subunit CCU1 triggers the SOU subunit (signal TRIG_CCU1).

If counter register CN0 of channel x is reset by its own CCU0 unit (i.e. the compare match of $CN0 \geq CM0$ configured by RST_CCU0=0), following statements are valid:

- if $CM0=0$ or $CM0=1$, 0% duty cycle ($=\overline{SL}$) is generated independent of CM1. The counter CN0 is not counting.
- the configuration of $CM1=0$ represents 0% duty cycle ($=\overline{SL}$) at the output
- the configuration of $CM1 \geq CM0$ represents 100% duty cycle ($=SL$)

If counter register CN0 of x channel is reset by the trigger signal coming from another channel or the assigned TIM module (configured by RST_CCU0=1), following statements are valid:

- CM0 defines the edge to SL value, CM1 defines the edge to \overline{SL} value
- if $CM0=CM1$, the output is 100% SL (CM0 has higher priority)
- if $CM0=0$, the output stays at its last value (CN0 stops counting)

The hardware ensures that for both 0% and 100% duty cycle no glitch occurs at the output of the TOM channel.

The SOU subunit is responsible for output signal generation. On a trigger TRIG_CCU0 from subunit CCU0 or TRIG_CCU1 from subunit CCU1 a SR-FlipFlop of subunit SOU is either set or reset. If it is set or reset depends on the configuration bit SL of the control register TOMi_CHx_CTRL. The initial signal output level for the channel is the erse value of the bit SL.

Figure 26-39 clarifies the PWM output behaviour with respect to the SL bit definition.

The output level on the TOM channel output pin TOM[i]_CH[x]_OUT is captured in bit OL of register TOMi_CHx_STAT.

26.11.3.4 Duty cycle, period and selected counter clock frequency update mechanisms

The two action registers CM0 and CM1 can be reloaded with the content of the shadow registers SR0 and SR1. The register CLK_SRC that determines the clock frequency of

Generic Timer Module (GTM)

the counter register CN0 can be reloaded with its shadow register CLK_SRC_SR (bit field in register TOMi_CHx_CTRL)

The update of the register CM0, CM1 and CLK_SRC with the content of its shadow register is done when the reset of the counter register CN0 is requested (via signal RESET). This reset of CN0 is done if the comparison of CN0 greater or equal than CM0 is true or when the reset is triggered by another TOM channel [x-1] via the signal TRIG_[x-1].

For TOM0 channel 0 TRIG_[1] is '0' hard coded.

With the update of the register CLK_SRC at the end of a period a new counter CN0 clock frequency can easily be adjusted.

An update of duty cycle, period and counter CN0 clock frequency becoming effective synchronously with start of a new period can easily be reached by performing following steps:

1. disable the update of the action register with the content of the corresponding shadow register by setting the channel specific configuration bit UPEN_CTRL[z] of register TOMi_TGCy_GLB_CTRL to '0'.
2. write new desired values to SR0, SR1, CLK_SRC_SR
3. enable update of the action register by setting the channel specific configuration bit UPEN_CTRL[z] of register TOMi_TGCy_GLB_CTRL to '1'.

synchronous update of duty cycle only

A synchronous update of only the duty cycle can be done by simply writing the desired new value to register SR1 without preceding disable of the update mechanism (as described in the section above). The new duty cycle is then applied in the period following the period where the update of register SR1 was done.

synchronous update of duty cycle

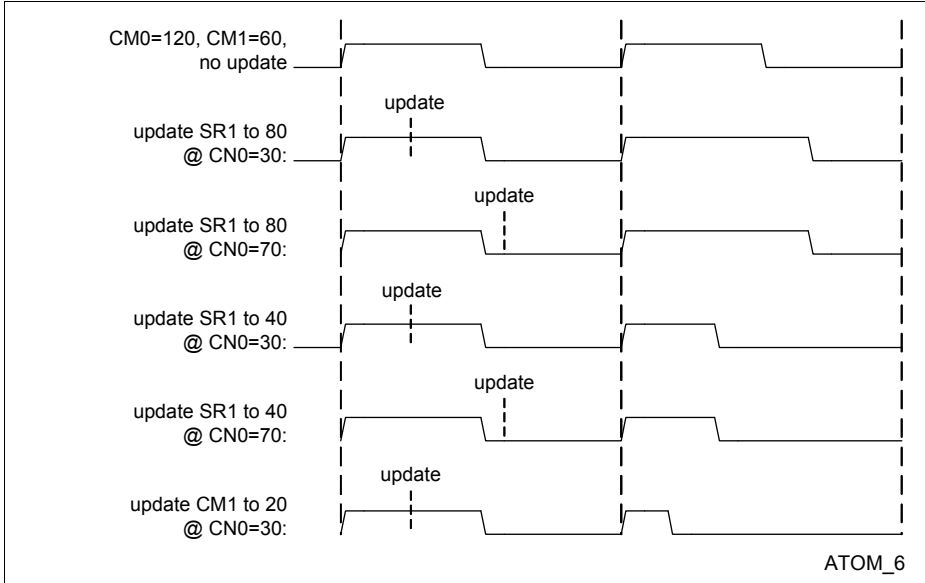


Figure 26-37 synchronous update of duty cycle

asynchronous update of duty cycle only

If the update of the duty cycle should be performed independent of the start of a new period (asynchronous), the desired new value can be written directly to register CM1. In this case it is recommended to additionally either disable the synchronous update mechanism as a whole (i.e. clearing bits UPEN_CTRL[z] of corresponding channel [x] in register TOMi_TGXy_GLB_CTRL) or updating SR1 with the same value as CM1 before writing to CM1.

Depending on the point of time of the update of CM1 in relation to the actual value of CN0 and CM1, the new duty cycle is applied in the current period or the following period (see [Figure 26-38](#)). In any case the creation of glitches are avoided. The new duty cycle may jitter from update to update by a maximum of one period (given by CM0). However, the period remains unchanged.

asynchronous update of duty cycle

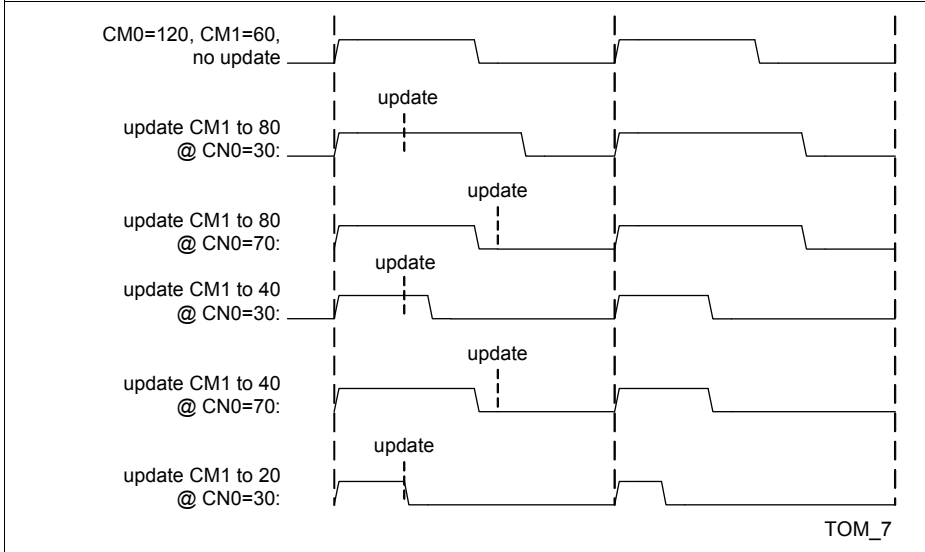


Figure 26-38 asynchronous update of duty cycle

26.11.3.5 TOM continuous mode

In continuous mode the TOM channel starts incrementing the counter register CN0 once it is enabled by setting the corresponding bits in register TOMi_TGCy_ENDIS_STAT (refer to [Section 26.11.2.2](#) for details of enabling a TOM channel).

The signal level of the generated output signal can be configured with the configuration bit SL of the channel configuration register TOMi_CHx_CTRL.

If the counter CN0 is reset from CM0 back to zero, the first edge of a period is generated at TOM[i]_CH[x]_OUT.

The second edge of the period is generated if CN0 has reached CM1.

Every time the counter CN0 has reached the value of CM0 it is reset back to zero and proceeds with incrementing.

PWM Output with respect to configuration bit SL in continuous mode

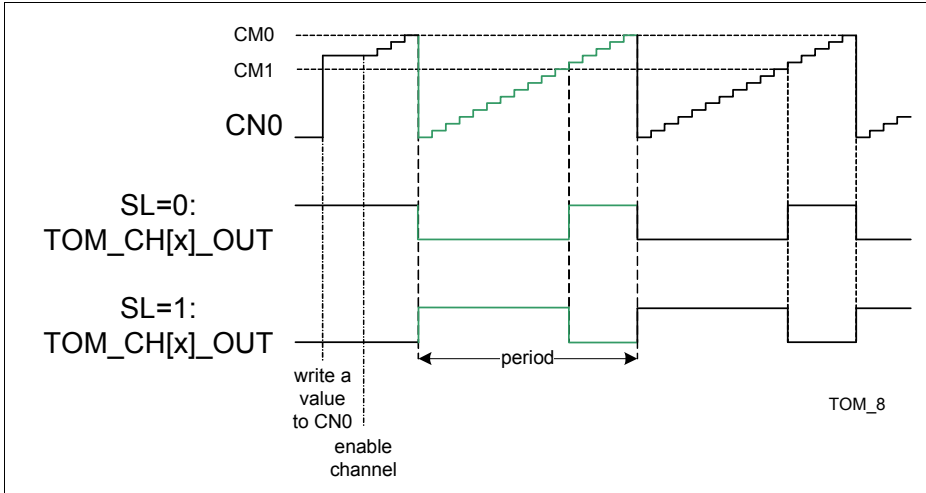


Figure 26-39 PWM Output with respect to configuration bit SL in continuous mode

26.11.3.6 TOM One shot mode

In One-shot mode, the TOM channel generates one pulse with a signal level specified by the configuration bit SL in the channel [x] configuration register TOMi_CHx_CTRL.

First the channel has to be enabled by setting the corresponding TOMi_TGCy_ENDIS_STAT value and the one-shot mode has to be enabled by setting bit OSM in register TOMi_CHx_CTRL.

In one-shot mode the counter CN0 will not be increment once the channel is enabled.

A write access to the register CN0 triggers the start of pulse generation (i.e. the increment of the counter register CN0).

If SPE mode of TOM[i] channel 2 is enabled (set bit SPEM of register TOMi_CH2_CTRL), also the trigger signal SPE[i]_NIPD can trigger the reset of register CN0 to zero and a start of the pulse generation.

The new value of CN0 determines the start delay of the first edge. The delay time of the first edge is given by $(CM0 - CN0)$ multiplied with period defined by current value of CLK_SRC.

If the counter CN0 is reset from CM0 back to zero, the first edge at TOM[i]_CH[x]_OUT is generated.

Generic Timer Module (GTM)

To avoid an update of CMx register with content of SRx register at this point in time, the automatic update should be disabled by setting UPEN_CTRLx = 00 (in register TOMi_CHx_CTRL).

The second edge is generated if CN0 is greater or equal than CM1 (i.e. CN0 was increment until it has reached CM1 or CN0 is greater than CM1 after an update of CM1). If the counter CN0 has reached the value of CM0 a second time, the counter stops.

PWM Output with respect to configuration bit SL in one-shot mode: trigger by writing to CN0

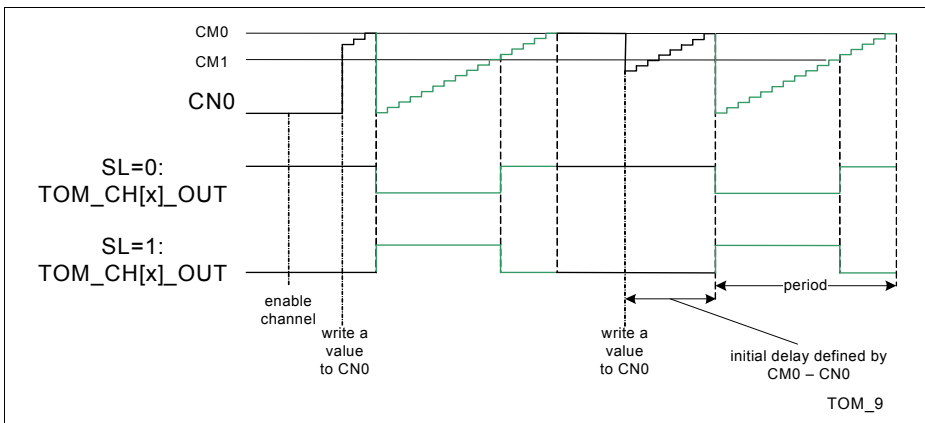


Figure 26-40 PWM Output with respect to configuration bit SL in one-shot mode: trigger by writing to CN0

Further output of single periods can be started by a write access to register CN0.

If CN0 is already incrementing (i.e. started by writing to CN0 a value $CN0_{start} < CM0$), the affect of a second write access to CN0 depends on the phase of CN0:

phase 1: update of CN0 before CN0 reaches first time CM0

phase 2: update of CN0 after CN0 has reached first time CM0 but less than CM1

phase 3: update of CN0 after CN0 has reached first time CM0 and CN0 is greater than or equal CM1

In phase 1: writing to counter CN0 a value $CN0_{new} < CM0$ leads to shift of first edge (generated if CN0 reaches CM0 first time) by the time $CM0 - CN0_{new}$.

In phase 2: writing to incrementing counter CN0 a value $CN0_{new} < CM1$ while $CN0_{old}$ is below CM1 leads to a lengthening of the pulse. The counter CN0 stops if it reaches CM0.

Generic Timer Module (GTM)

In phase 3: writing to incrementing counter CN0 a value CN0new while CN0old is already greater than or equal CM1 leads to an immediate restart of a single pulse generation inclusive the initial delay defined by CM0 - CN0new.

26.11.3.7 Pulse count modulation

At the output TOM[i]_CH15_OUT a pulse count modulated signal can be generated instead of the simple PWM output signal.

Figure 26-36 outlines the circuit for Pulse Count Modulation.

The PCM mode is enabled by setting bit BITREV to 1.

With the configuration bit BITREV=1 a bit-reversing of the counter output CN0 is configured. In this case the bits LSB and MSB are swapped, the bits LSB+1 and MSB-1 are swapped, the bits LSB+2 and MSB-2 are swapped and so on.

The effect of bit-reversing of the CN0 register value is shown in the following **Figure 26-41**.

Bit reversing of counter

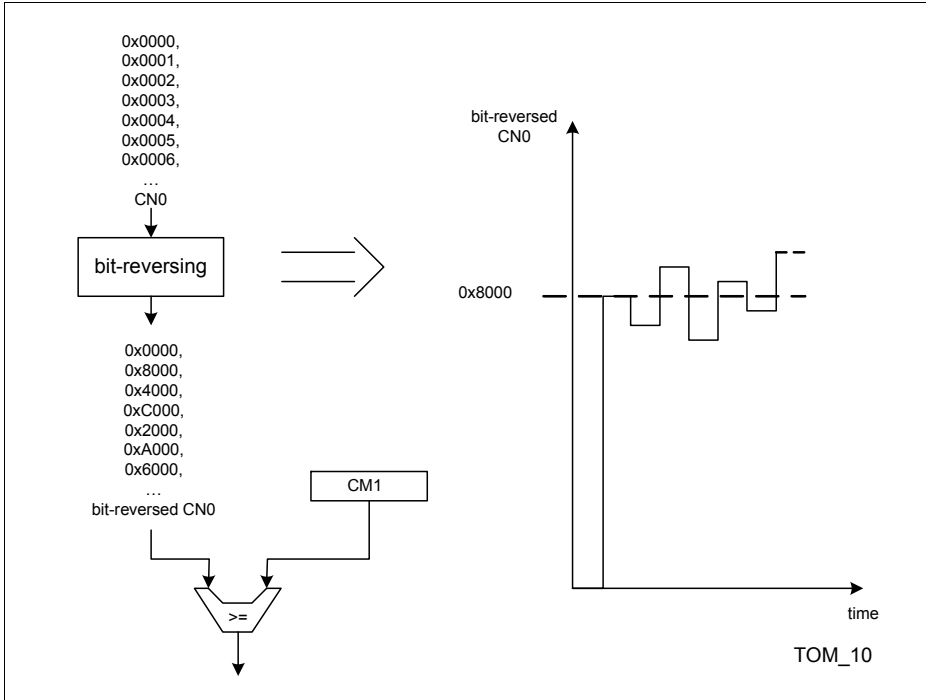


Figure 26-41 Bit reversing of counter

In the PCM mode the counter register CN0 is increment by every clock tick depending on configured CMU clock (CMU_FXCLK).

The output of counter register CN0 is first bit-reversed and than compared with the configured register value CM1.

If the bit-reversed value of register CN0 is greater than CM1, the SR-FlipFlop of submodule SOU is set (depending on configuration register SL) otherwise the SR-FlipFlop is reset. This generates at the output TOM[i]_CH15_OUT a pulse count modulated signal.

In PCM mode the CM0 register - in which the period id defined - normally has to be set to its maximum value 0xFFFF.

Note: In this mode the interrupt CCU1TC (see register ATOMi_CHx_IRQ_NOTIFY) is set every time if bitreserve values of CN0 is greater than or equal CM1 which may

Generic Timer Module (GTM)

be multiple times during one period. Therefore, from application point of view it is not useful to enable this interrupt.

26.11.4 TOM BLDC Support

The TOM submodule offers in combination with the SPE submodule a BLDC support. To drive a BLDC engine TOM channels 0 to 7 can be used.

The BLDC support can be configured by setting the SPEM bit inside the TOM_i_CH_[z]_CTRL register. When this bit is set the TOM channel output is controlled through the SPE_OUT_(z) signal coming from the SPE submodule (see [Figure 26-34](#)). Please refer to [Section 26.17](#) for a detailed description of the SPE submodule.

The TOM_[i]_CH₂ can be used together with the SPE module to trigger a delayed update of the SPE_OUT_CTRL register after new input pattern detected by SPE (signalled by SPE_[i]_NIPD). This feature is configured on TOM_[i]_CH₂ by setting SPEM=1 and OSM=1. For details please refer to section of SPE submodule description.

26.11.5 TOM Gated Counter Mode

Each TOM - SPE module combination provides also the feature of a gated counter mode. This is reached by using the FSOI input of a TIM module to gate the clock of a CCU0 submodule.

To configure this mode, registers of module SPE should be set as following:

- the SPE should be enabled (bit SPE_EN = 1),
- all three TIM inputs should be disabled (SIE0 = SIE1 = SIE2 = 0),
- SPE_i_OUT_CTRL should be set to 00005555h (set SPE_OUT() to '0'),
- mode FSOM should be enabled (FSOM=1),
- set in bit field FSOL bit c if channel c of module TOM is chosen for gated counter mode

Additionally in module TOM

- the SPE mode should be disabled (SPEM=0) and
- the gated counter mode should be enabled (GCM=1)

As a result of this configuration, the counter CN0 in submodule CCU0 of TOM channel c counts as long as input FSOI is '0'.

26.11.6 TOM Interrupt signals

The following table describes TOM interrupt signals:

Table 26-31 TOM Interrupt signals

Signal	Description
TOM_CCU0TCx_IRQ	CCU0 Trigger condition interrupt for channel x
TOM_CCU1TCx_IRQ	CCU1 Trigger condition interrupt for channel x

26.11.7 TOM Configuration Register overview

The following table gives an overview of the TOM configuration registers.

Table 26-32 TOM Configuration Register overview

Register name	Description	Details in Section
TOMi_TGC0_GLB_CTRL	TGC0 global control register	Section 26.11.8.1
TOMi_TGC1_GLB_CTRL	TGC1 global control register	
TOMi_TGC0_ENDIS_CTRL	TGC0 enable/disable control register	Section 26.11.8.2
TOMi_TGC1_ENDIS_CTRL	TGC1 enable/disable control register	
TOMi_TGC0_ENDIS_STAT	TGC0 enable/disable status register	Section 26.11.8.3
TOMi_TGC1_ENDIS_STAT	TGC1 enable/disable status register	
TOMi_TGC0_ACT_TB	TGC0 action time base register	Section 26.11.8.4
TOMi_TGC1_ACT_TB	TGC1 action time base register	
TOMi_TGC0_OUTEN_CTRL	TGC0 output enable control register	Section 26.11.8.5
TOMi_TGC1_OUTEN_CTRL	TGC1 output enable control register	
TOMi_TGC0_OUTEN_STAT	TGC0 output enable status register	Section 26.11.8.6
TOMi_TGC1_OUTEN_STAT	TGC1 output enable status register	
TOMi_TGC0_FUPD_CTRL	TGC0 force update control register	Section 26.11.8.7
TOMi_TGC1_FUPD_CTRL	TGC1 force update control register	

Generic Timer Module (GTM)
Table 26-32 TOM Configuration Register overview (cont'd)

Register name	Description	Details in Section
TOMi_TGC0_INT_TRIG	TGC0 internal trigger control register	Section 26.11.8.8
TOMi_TGC1_INT_TRIG	TGC1 internal trigger control register	
TOMi_CHx_CTRL	TOM Channel x control register (x=0...14)	Section 26.11.8.9
TOMi_CH15_CTRL	TOM Channel 15 control register	Section 26.11.8.10
TOMi_CHx_CN0	TOM Channel x CCU0 counter register (x=0...15)	Section 26.11.8.11
TOMi_CHx_CM0	TOM Channel x CCU0 compare register (x=0...15)	Section 26.11.8.12
TOMi_CHx_SR0	TOM Channel x CCU0 compare shadow register (x=0...15)	Section 26.11.8.13
TOMi_CHx_CM1	TOM Channel x CCU1 compare register (x=0...15)	Section 26.11.8.14
TOMi_CHx_SR1	TOM Channel x CCU1 compare shadow register (x=0...15)	Section 26.11.8.15
TOMi_CHx_STAT	TOM channel status register (x=0...15)	Section 26.11.8.16
TOMi_CHx_IRQ_NOTIFY	TOM channel x interrupt notification register (x=0...15)	Section 26.11.8.17
TOMi_CHx_IRQ_EN	TOM channel x interrupt enable register (x=0...15)	Section 26.11.8.18
TOMi_CHx_IRQ_FORCINT	TOM channel x software interrupt generation register (x=0...15)	Section 26.11.8.19
TOMi_CHx_IRQ_MODE	IRQ mode configuration register (x=0...15)	Section 26.11.8.20

26.11.8 TOM Configuration Registers Description

All of the following registers are 32-bit only accessible.

26.11.8.1 Register TOMi_TGC0_GLB_CTRL

GTM_TOMi_TGC0_GLB_CTRL (i=0-1)

TOMi TGC0 Global Control Register(08030_H+i*800_H) Reset Value: 00000000_H

GTM_TOMi_TGC1_GLB_CTRL (i=0-1)

TOMi TGC1 Global Control Register(08230_H+i*800_H) Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPEN_CT RL7		UPEN_CT RL6		UPEN_CT RL5		UPEN_CT RL4		UPEN_CT RL3		UPEN_CT RL2		UPEN_CT RL1		UPEN_CT RL0	
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RST _CH 7	RST _CH 6	RST _CH 5	RST _CH 4	RST _CH 3	RST _CH 2	RST _CH 1	RST _CH 0	Reserved							HOS T_T RIG
w	w	w	w	w	w	w	w	r							w

Field	Bits	Type	Description
HOST_TRIG	0	w	<p>Trigger request signal (see TGC0, TGC1) to update the register ENDIS_STAT and OUTEN_STAT</p> <p>0_B no trigger request 1_B set trigger request Read as 0.</p> <p><i>Note:</i> This flag is cleared automatically after triggering the update</p>
Reserved	[7:1]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>
RST_CH0	8	w	<p>Software reset of channel 0</p> <p>0_B No action 1_B Reset channel Read as 0.</p> <p><i>Note:</i> This bit is cleared automatically after write by CPU. The channel registers are set to their reset values and channel operation is stopped immediately. The S-r FlipFlop SOUR is set to '1'.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
RST_CH1	9	w	Software reset of channel 1 See bit 8
RST_CH2	10	w	Software reset of channel 2 See bit 8
RST_CH3	11	w	Software reset of channel 3 See bit 8
RST_CH4	12	w	Software reset of channel 4 See bit 8
RST_CH5	13	w	Software reset of channel 5 See bit 8
RST_CH6	14	w	Software reset of channel 6 See bit 8
RST_CH7	15	w	Software reset of channel 7 See bit 8
UPEN_CT RL0	[17:16]	rw	TOM channel 0 enable update of register CM0, CM1 and CLK_SRC from SR0, SR1 and CLK_SRC_SR Write of following double bit values is possible: 00 _B don't care, bits 1:0 will not be changed 01 _B update disabled: is read as 00 (see below) 10 _B update enabled: is read as 11 (see below) 11 _B don't care, bits 1:0 will not be changed Read of following double values means: 00 _B channel disabled 11 _B channel enabled
UPEN_CT RL1	[19:18]	rw	TOM channel 1 enable update of register CM0, CM1 and CLK_SRC See bits 17:16
UPEN_CT RL2	[21:20]	rw	TOM channel 2 enable update of register CM0, CM1 and CLK_SRC See bits 17:16
UPEN_CT RL3	[23:22]	rw	TOM channel 3 enable update of register CM0, CM1 and CLK_SRC See bits 17:16
UPEN_CT RL4	[25:24]	rw	TOM channel 4 enable update of register CM0, CM1 and CLK_SRC See bits 17:16

Generic Timer Module (GTM)

Field	Bits	Type	Description
UPEN_CT RL5	[27:26]	rw	TOM channel 5 enable update of register CM0, CM1 and CLK_SRC See bits 17:16
UPEN_CT RL6	[29:28]	rw	TOM channel 6 enable update of register CM0, CM1 and CLK_SRC See bits 17:16
UPEN_CT RL7	[31:30]	rw	TOM channel 7 enable update of register CM0, CM1 and CLK_SRC See bits 17:16

26.11.8.2 Register TOMi_TGC0_ENDIS_CTRL

GTM_TOMi_TGC0_ENDIS_CTRL (i=0-1)

TOMi TGC0 Enable/Disable Control Register

(08070_H+i*800_H)

Reset Value: 00000000_H

GTM_TOMi_TGC1_ENDIS_CTRL (i=0-1)

TOMi TGC1 Enable/Disable Control Register

(08270_H+i*800_H)

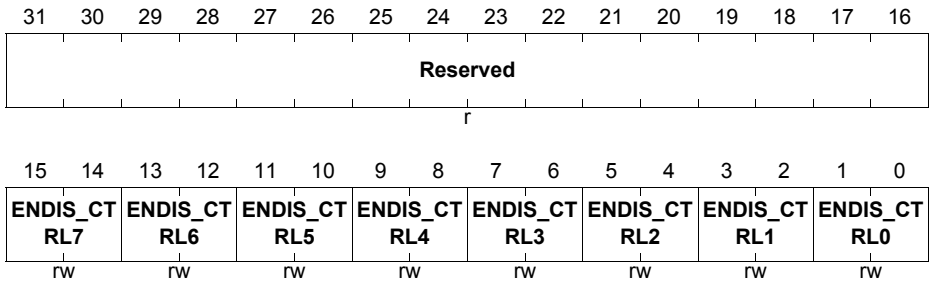
Reset Value: 00000000_H

GTM_ATOMi_AGC_ENDIS_CTRL (i=0-3)

ATOMi AGC Enable/Disable Control Register

(0D044_H+i*800_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
ENDIS_CT RL0	[1:0]	rw	<p>(A)TOM channel 0 enable/disable update value If a TOM channel is disabled, the counter CN0 is stopped and the FlipFlop SOUR is set to the inverse value of control bit SL. On an enable event, the counter CN0 starts counting from its current value. Write of following double bit values is possible: 00_B don't care, bits 1:0 of register ENDIS_STAT will not be changed on an update trigger 01_B disable channel on an update trigger 10_B enable channel on an update trigger 11_B don't change bits 1:0 of this register Note: if the channel is disabled (ENDIS[0]=0) or the output is disabled (OUTEN[0]=0), the TOM channel 0 output TOM_OUT[0] is the inverted value of bit SL.</p>
ENDIS_CT RL1	[3:2]	rw	<p>(A)TOM channel 1 enable/disable update value See bits 1:0</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
ENDIS_CT RL2	[5:4]	rw	(A)TOM channel 2 enable/disable update value See bits 1:0
ENDIS_CT RL3	[7:6]	rw	(A)TOM channel 3 enable/disable update value See bits 1:0
ENDIS_CT RL4	[9:8]	rw	(A)TOM channel 4 enable/disable update value See bits 1:0
ENDIS_CT RL5	[11:10]	rw	(A)TOM channel 5 enable/disable update value See bits 1:0
ENDIS_CT RL6	[13:12]	rw	(A)TOM channel 6 enable/disable update value See bits 1:0
ENDIS_CT RL7	[15:14]	rw	(A)TOM channel 7 enable/disable update value See bits 1:0
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.11.8.3 Register TOMi_TGC0_ENDIS_STAT

GTM_TOMi_TGC0_ENDIS_STAT (i=0-1)

TOMi TGC0 Enable/Disable Status Register

(08074_H+i*800_H)

Reset Value: 00000000_H

GTM_TOMi_TGC1_ENDIS_STAT (i=0-1)

TOMi TGC1 Enable/Disable Status Register

(08274_H+i*800_H)

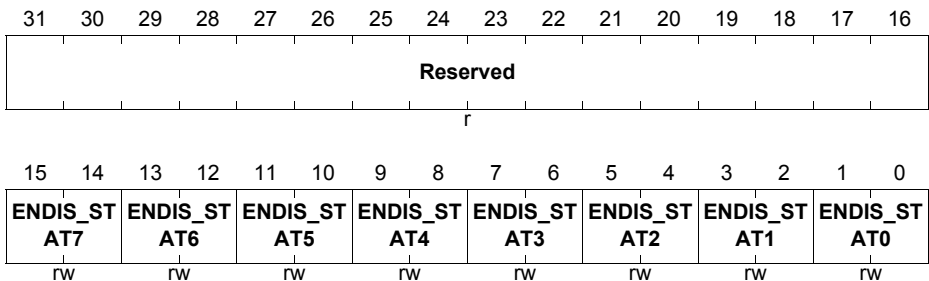
Reset Value: 00000000_H

GTM_ATOMi_AGC_ENDIS_STAT (i=0-3)

ATOMi AGC Enable/Disable Status Register

(0D048_H+i*800_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
ENDIS_ST AT0	[1:0]	rw	(A)TOM channel 0 enable/disable If a TOM channel is disabled, the counter CN0 is stopped and the FlipFlop SOUR is set to the inverse value of control bit SL. On an enable event, the counter CN0 starts counting from its current value. Write of following double bit values is possible: 00 _B don't care, bits 1:0 will not be changed 01 _B channel disabled: is read as 00 (see below) 10 _B channel enabled: is read as 11 (see below) 11 _B don't care, bits 1:0 will not be changed Read of following double values means: 00 _B channel disable 11 _B channel enable
ENDIS_ST AT1	[3:2]	rw	(A)TOM channel 1 enable/disable See bits 1:0
ENDIS_ST AT2	[5:4]	rw	(A)TOM channel 2 enable/disable See bits 1:0

Generic Timer Module (GTM)

Field	Bits	Type	Description
ENDIS_ST AT3	[7:6]	rw	(A)TOM channel 3 enable/disable See bits 1:0
ENDIS_ST AT4	[9:8]	rw	(A)TOM channel 4 enable/disable See bits 1:0
ENDIS_ST AT5	[11:10]	rw	(A)TOM channel 5 enable/disable See bits 1:0
ENDIS_ST AT6	[13:12]	rw	(A)TOM channel 6 enable/disable See bits 1:0
ENDIS_ST AT7	[15:14]	rw	(A)TOM channel 7 enable/disable See bits 1:0
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.11.8.4 Register TOMi_TGC0_ACT_TB

GTM_TOMi_TGC0_ACT_TB (i=0-1)

TOMi TGC0 Action Time Base Register

 $(08034_H + i * 800_H)$

 Reset Value: 00000000_H

GTM_TOMi_TGC1_ACT_TB (i=0-1)

TOMi TGC1 Action Time Base Register

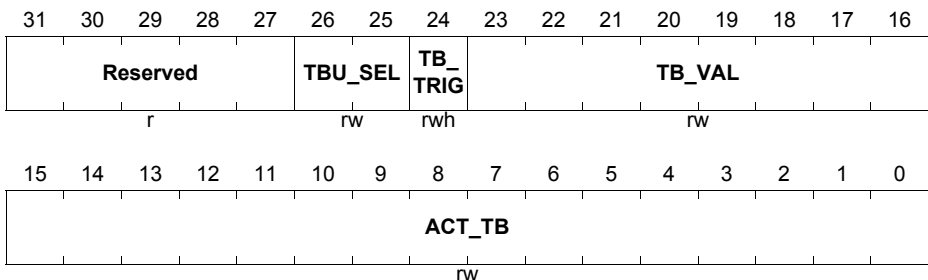
 $(08234_H + i * 800_H)$

 Reset Value: 00000000_H

GTM_ATOMi_AGC_ACT_TB (i=0-3)

TOMi TGC0 Action Time Base Register

 $(0D04C_H + i * 800_H)$

 Reset Value: 00000000_H


Field	Bits	Type	Description
ACT_TB	[23:0]	rw	Time base value specifies the signed compare value with selected signal TBU_TCx (x=0...2). If selected TBU_TSx value is in the interval [ACT_TB-007FFFFFFF _H ,ACT_TB] the event is in the past and the trigger is generated immediately. Otherwise the event is in the future and the trigger is generated if selected TBU_TSx is equal to ACT_TB.
TB_TRIG	24	rwh	Set trigger request 0 _B no trigger request 1 _B set trigger request Note: This flag is reset automatically if the selected time base unit (TBU_TS0 or TBU_TS1 or TBU_TS2 if present) has reached the value ACT_TB and the update of the register were triggered.

Generic Timer Module (GTM)

Field	Bits	Type	Description
TBU_SEL	[26:25]	rw	Selection of time base used for comparison 00 _B TBU_TS0 selected 01 _B TBU_TS1 selected 10 _B TBU_TS2 selected 11 _B Reserved; same as 00 _B Note: The bit combination “10” is only applicable if the TBU of the device contains three time base channels. Otherwise, this bit combination is also reserved. Please refer to GTM Architecture block diagram on page 3 to determine the number of channels for TBU of this device.
Reserved	[31:27]	r	Reserved Read as zero, should be written as zero

26.11.8.5 Register TOMi_TGC0_OUTEN_CTRL

GTM_TOMi_TGC0_OUTEN_CTRL (i=0-1)

TOMi TGC0 Output Enable Control Register

(08078_H+i*800_H)

Reset Value: 00000000_H

GTM_TOMi_TGC1_OUTEN_CTRL (i=0-1)

TOMi TGC1 Output Enable Control Register

(08278_H+i*800_H)

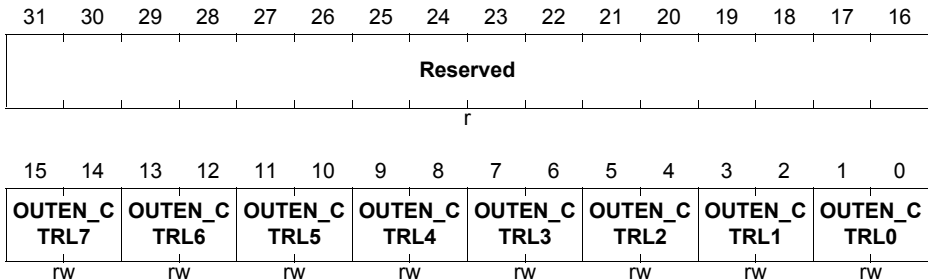
Reset Value: 00000000_H

GTM_ATOMi_AGC_OUTEN_CTRL (i=0-3)

ATOMi AGC Output Enable Control Register

(0D050_H+i*800_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
OUTEN_C TRL0	[1:0]	rw	Output (A)TOM_OUT(0) enable/disable update value Write of following double bit values is possible: 00 _B don't care, bits 1:0 of register OUTEN_STAT will not be changed on an update trigger 01 _B disable channel output on an update trigger 10 _B enable channel output on an update trigger 11 _B don't change bits 1:0 of this register Note: if the channel is disabled (ENDIS[0]=0) or the output is disabled (OUTEN[0]=0), the TOM channel 0 output TOM_OUT[0] is the inverted value of bit SL.
OUTEN_C TRL1	[3:2]	rw	Output (A)TOM_OUT(1)enable/disable update value See bits 1:0
OUTEN_C TRL2	[5:4]	rw	Output (A)TOM_OUT(2) enable/disable update value See bits 1:0
OUTEN_C TRL3	[7:6]	rw	Output (A)TOM_OUT(3) enable/disable update value See bits 1:0
OUTEN_C TRL4	[9:8]	rw	Output (A)TOM_OUT(4) enable/disable update value See bits 1:0
OUTEN_C TRL5	[11:10]	rw	Output (A)TOM_OUT(5) enable/disable update value See bits 1:0
OUTEN_C TRL6	[13:12]	rw	Output (A)TOM_OUT(6) enable/disable update value See bits 1:0
OUTEN_C TRL7	[15:14]	rw	Output (A)TOM_OUT(7) enable/disable update value See bits 1:0
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

26.11.8.6 Register TOMi_TGC0_OUTEN_STAT

GTM_TOMi_TGC0_OUTEN_STAT (i=0-1)

TOMi TGC0 Output Enable Status Register

(0807C_H+i*800_H)

Reset Value: 00000000_H

GTM_TOMi_TGC1_OUTEN_STAT (i=0-1)

TOMi TGC1 Output Enable Status Register

(0827C_H+i*800_H)

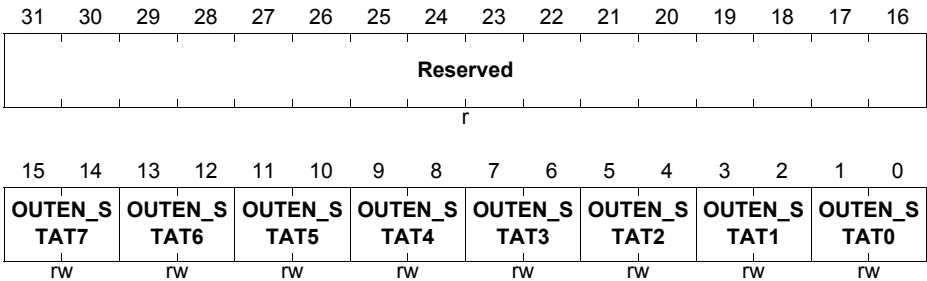
Reset Value: 00000000_H

GTM_ATOMi_AGC_OUTEN_STAT (i=0-3)

ATOMi AGC Output Enable Status Register

(0D054_H+i*800_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
OUTEN_S TAT0	[1:0]	rw	Control/status of output (A)TOM_OUT(0) Write of following double bit values is possible: 00 _B don't care, bits 1:0 will not be changed 01 _B channel disabled: is read as 00 (see below) 10 _B channel enabled: is read as 11 (see below) 11 _B don't care, bits 1:0 will not be changed Read of following double values means: 00 _B channel disable 11 _B channel enable
OUTEN_S TAT1	[3:2]	rw	Control/status of output (A)TOM_OUT(1) See bits 1:0
OUTEN_S TAT2	[5:4]	rw	Control/status of output (A)TOM_OUT(2) See bits 1:0
OUTEN_S TAT3	[7:6]	rw	Control/status of output (A)TOM_OUT(3) See bits 1:0
OUTEN_S TAT4	[9:8]	rw	Control/status of output (A)TOM_OUT(4) See bits 1:0

Generic Timer Module (GTM)

Field	Bits	Type	Description
OUTEN_S TAT5	[11:10]	rw	Control/status of output (A)TOM_OUT(5) See bits 1:0
OUTEN_S TAT6	[13:12]	rw	Control/status of output (A)TOM_OUT(6) See bits 1:0
OUTEN_S TAT7	[15:14]	rw	Control/status of output (A)TOM_OUT(7) See bits 1:0
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

26.11.8.7 Register TOMi_TGC0_FUPD_CTRL

GTM_TOMi_TGC0_FUPD_CTRL (i=0-1)

TOMi TGC0 Force Update Control Register

(08038_H+i*800_H)

Reset Value: 00000000_H

GTM_TOMi_TGC1_FUPD_CTRL (i=0-1)

TOMi TGC1 Force Update Control Register

(08238_H+i*800_H)

Reset Value: 00000000_H

GTM_ATOMi_AGC_FUPD_CTRL (i=0-3)

ATOMi AGC Force Update Control Register

(0D058_H+i*800_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSTCN0_ CH7	RSTCN0_ CH6	RSTCN0_ CH5	RSTCN0_ CH4	RSTCN0_ CH3	RSTCN0_ CH2	RSTCN0_ CH1	RSTCN0_ CH0								
rw	rw	rw	rw	rw	rw	rw	rw								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FUPD_CT RL7	FUPD_CT RL6	FUPD_CT RL5	FUPD_CT RL4	FUPD_CT RL3	FUPD_CT RL2	FUPD_CT RL1	FUPD_CT RL0								
rw	rw	rw	rw	rw	rw	rw	rw								

Field	Bits	Type	Description
FUPD_CT RL0	[1:0]	rw	Force update of (A)TOM channel 0 operation registers Write of following double bit values is possible: 00 _B don't care, bits 1:0 will not be changed 01 _B force update disabled: is read as 00 (see below) 10 _B force update enabled: is read as 11 (see below) 11 _B don't care, bits 1:0 will not be changed Read of following double values means: 00 _B force update disabled 11 _B force channel enabled
FUPD_CT RL1	[3:2]	rw	Force update of (A)TOM channel 1 operation registers See bits 1:0
FUPD_CT RL2	[5:4]	rw	Force update of (A)TOM channel 2 operation registers See bits 1:0

Generic Timer Module (GTM)

Field	Bits	Type	Description
FUPD_CT RL3	[7:6]	rw	Force update of (A)TOM channel 3 operation registers See bits 1:0
FUPD_CT RL4	[9:8]	rw	Force update of (A)TOM channel 4 operation registers See bits 1:0
FUPD_CT RL5	[11:10]	rw	Force update of (A)TOM channel 5 operation registers See bits 1:0
FUPD_CT RL6	[13:12]	rw	Force update of (A)TOM channel 6 operation registers See bits 1:0
FUPD_CT RL7	[15:14]	rw	Force update of (A)TOM channel 7 operation registers See bits 1:0
RSTCN0_ CH0	[17:16]	rw	Reset CN0 of channel 0 on force update event Write of following double bit values is possible: 00 _B don't care, bits 1:0 will not be changed 01 _B CN0 is not reset on forced update: is read as 00 (see below) 10 _B CN0 is reset on forced update: is read as 11 (see below) 11 _B don't care, bits 1:0 will not be changed Read of following double values means: 00 _B CN0 is not reset on forced update 11 _B CN0 is reset on forced update
RSTCN0_ CH1	[19:18]	rw	Reset CN0 of channel 1 on force update event See bits 17:16
RSTCN0_ CH2	[21:20]	rw	Reset CN0 of channel 2 on force update event See bits 17:16
RSTCN0_ CH3	[23:22]	rw	Reset CN0 of channel 3 on force update event See bits 17:16
RSTCN0_ CH4	[25:24]	rw	Reset CN0 of channel 4 on force update event See bits 17:16
RSTCN0_ CH5	[27:26]	rw	Reset CN0 of channel 5 on force update event See bits 17:16

Generic Timer Module (GTM)

Field	Bits	Type	Description
RSTCN0_ CH6	[29:28]	rw	Reset CN0 of channel 6 on force update event See bits 17:16
RSTCN0_ CH7	[31:30]	rw	Reset CN0 of channel 7 on force update event See bits 17:16

26.11.8.8 Register TOMi_TGC0_INT_TRIG

GTM_TOMi_TGC0_INT_TRIG (i=0-1)

TOMi TGC0 Internal Trigger Control Register

(0803C_H+i*800_H)

Reset Value: 00000000_H

GTM_TOMi_TGC1_INT_TRIG (i=0-1)

TOMi TGC1 Internal Trigger Control Register

(0823C_H+i*800_H)

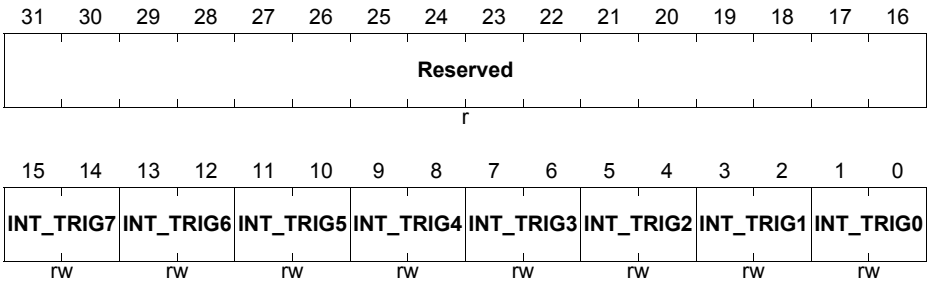
Reset Value: 00000000_H

GTM_ATOMi_AGC_INT_TRIG (i=0-3)

ATOMi AGC Internal Trigger Control Register

(0D05C_H+i*800_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
INT_TRIG 0	[1:0]	rw	Select input signal TRIG_0 as a trigger source Write of following double bit values is possible: 00 _B don't care, bits 1:0 will not be changed 01 _B internal trigger from channel 0 (TRIG_0) not used: is read as 00 (see below) 10 _B internal trigger from channel 0 (TRIG_0) used: is read as 11 (see below) 11 _B don't care, bits 1:0 will not be changed Read of following double values means: 00 _B internal trigger from channel 0 (TRIG_0) not used 11 _B internal trigger from channel 0 (TRIG_0) used
INT_TRIG 1	[3:2]	rw	Select input signal TRIG_1 as a trigger source See bits 1:0
INT_TRIG 2	[5:4]	rw	Select input signal TRIG_2 as a trigger source See bits 1:0
INT_TRIG 3	[7:6]	rw	Select input signal TRIG_3 as a trigger source See bits 1:0

Generic Timer Module (GTM)

Field	Bits	Type	Description
INT_TRIG 4	[9:8]	rw	Select input signal TRIG_4 as a trigger source See bits 1:0
INT_TRIG 5	[11:10]	rw	Select input signal TRIG_5 as a trigger source See bits 1:0
INT_TRIG 6	[13:12]	rw	Select input signal TRIG_6 as a trigger source See bits 1:0
INT_TRIG 7	[15:14]	rw	Select input signal TRIG_7 as a trigger source See bits 1:0
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

26.11.8.9 Register TOMi_CHx_CTRL (x:0...14)

GTM_TOM0_CHx_CTRL (x=0-14)

TOM0 Channel x Control Register'

 $(08000_H + x * 0040_H)$

 Reset Value: 0000800_H

GTM_TOM1_CHx_CTRL (x=0-14)

TOM1 Channel x Control Register'

 $(08800_H + x * 0040_H)$

 Reset Value: 0000800_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	GCM	SPE M	Rese rved	OSM	Rese rved	TRIG OUT	Reserved			RST _CC U0	Reserved				
r	rw	rw	r	rw	r	rw	r			rw	r				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rese rved	CLK_SRC_SR		SL	Reserved											
r	rw		rw	r											

Field	Bits	Type	Description
Reserved	[10:0]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

Field	Bits	Type	Description
SL	11	rw	Signal level for duty cycle 0 _B Low signal level 1 _B High signal level If the output is disabled, the output TOM_OUT[x] is set to inverse value of SL.
CLK_SRC_SR	[14:12]	rw	Clock source select for channel The register CLK_SRC is updated with the value of CLK_SRC_SR together with the update of register CM0 and CM1. The input of the FX clock divider depends on the value of FXCLK_SEL (see CMU). 000 _B CMU_FXCLK(0) selected: clock selected by FXCLKSEL 001 _B CMU_FXCLK(1) selected: clock selected by FXCLKSEL / 2 ⁴ 010 _B CMU_FXCLK(2) selected: clock selected by FXCLKSEL / 2 ⁸ 011 _B CMU_FXCLK(3) selected: clock selected by FXCLKSEL / 2 ¹² 100 _B CMU_FXCLK(4) selected: clock selected by FXCLKSEL / 2 ¹⁶ 101 _B no CMU_FXCLK selected, clock of channel stopped 110 _B no CMU_FXCLK selected, clock of channel stopped 111 _B no CMU_FXCLK selected, clock of channel stopped Note: if clock of channel is stopped (i.e. CLK_SRC = 101/110/111), the channel can only be restarted by resetting CLK_SRC_SR to a value of 000 to 100 and forcing an update via the force update mechanism.
Reserved	[19:15]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

Field	Bits	Type	Description
RST_CCU0	20	rw	Reset source of CCU0 0 _B Reset counter register CN0 to 0 on matching comparison CM0 1 _B Reset counter register CN0 to 0 on trigger TRIG_[x-1] <i>Note: On TOM channel 2 SPEM=1 has special meaning. If SPEM = 1, the signal SPE_NIPD triggers the reset of CN0 independent of RST_CN0.</i>
Reserved	[23:21]	r	Reserved Read as zero, should be written as zero
TRIGOUT	24	rw	Trigger output selection (output signal TRIG_[x]) of module TOM_CH[x] 0 _B TRIG_[x] is TRIG_[x-1] 1 _B TRIG_[x] is TRIG_CCU0
Reserved	25	r	Reserved Read as zero, should be written as zero
OSM	26	rw	One-shot mode In this mode the counter CN0 counts for only one period The length of period is defined by CM0 A write access to the register CN0 triggers the start of counting 0 _B One-shot mode disabled 1 _B One-shot mode enabled
Reserved	27	r	Reserved Read as zero, should be written as zero
SPEM	28	rw	SPE mode enable for channel 0 _B SPE mode disabled 1 _B SPE mode enabled <i>Note: The SPE mode is only implemented for TOM instances connected to a SPE module and only for channels 0 to 7.</i> <i>Note: On TOM channel 2 SPEM=1 has special meaning. If SPEM = 1, the signal SPE_NIPD triggers the reset of CN0.</i> <i>If SPEM = 1 and OSM=1, the signal SPE_NIPD triggers the start of single pulse generation</i>

Generic Timer Module (GTM)

Field	Bits	Type	Description
GCM	29	rw	Gated Counter Mode enable 0 _B Gated Counter mode disabled 1 _B Gated Counter mode enabled Note: The Gated Counter mode is only available for TOM instances connected to a SPE module and only for channels 0 to 7.
Reserved	[31:30]	r	Reserved Read as zero, should be written as zero

26.11.8.10 Register TOMi_CH15_CTRL

GTM_TOMi_CH15_CTRL (i=0-1)

TOMi Channel 15 Control Register(083C0_H+i*800_H) Reset Value: 0000800_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Reserved				BIT	OSM	Reserved	TRIG OUT	Reserved				RST _CC U0	Reserved			
r				nw	nw	r	nw	r				nw	r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reserved	CLK_SRC_SR			SL	Reserved											
r	nw			nw	r											

Field	Bits	Type	Description
Reserved	[10:0]	r	Reserved Read as zero, should be written as zero
SL	11	nw	Signal level for duty cycle 0 _B Low signal level 1 _B High signal level If the output is disabled, the output ATOM_OUT[x] is set to inverse value of SL.

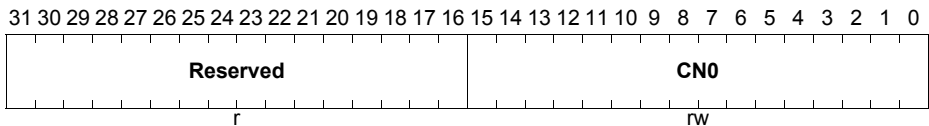
Generic Timer Module (GTM)

Field	Bits	Type	Description
CLK_SRC_SR	[14:12]	rw	<p>Clock source select for channel</p> <p>The register CLK_SRC is updated with the value of CLK_SRC_SR together with the update of register CM0 and CM1.</p> <p>The input of the FX clock divider depends on the value of FXCLK_SEL (see CMU).</p> <p>000_B CMU_FXCLK(0) selected: clock selected by FXCLKSEL</p> <p>001_B CMU_FXCLK(1) selected: clock selected by FXCLKSEL / 2⁴</p> <p>010_B CMU_FXCLK(2) selected: clock selected by FXCLKSEL / 2⁸</p> <p>011_B CMU_FXCLK(3) selected: clock selected by FXCLKSEL / 2¹²</p> <p>100_B CMU_FXCLK(4) selected: clock selected by FXCLKSEL / 2¹⁶</p> <p>101_B no CMU_FXCLK selected, clock of channel stopped</p> <p>110_B no CMU_FXCLK selected, clock of channel stopped</p> <p>111_B no CMU_FXCLK selected, clock of channel stopped</p> <p>Note: if clock of channel is stopped (i.e. CLK_SRC = 101/110/111), the channel can only be restarted by resetting CLK_SRC_SR to a value of 000 to 100 and forcing an update via the force update mechanism.</p>
Reserved	[19:15]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>
RST_CCU0	20	rw	<p>Reset source of CCU0</p> <p>0_B Reset counter register CN0 to 0 on matching comparison CM0</p> <p>1_B Reset counter register CN0 to 0 on trigger TRIG_14</p>
Reserved	[23:21]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>
TRIGOUT	24	rw	<p>Trigger output selection (output signal TRIG_15) of module TOM_CH15</p> <p>0_B TRIG_15 is TRIG_14</p> <p>1_B TRIG_15 is TRIG_CCU0</p>

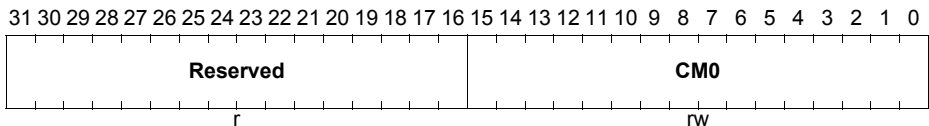
Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	25	r	Reserved Read as zero, should be written as zero
OSM	26	rw	One-shot mode In this mode the counter CN0 counts for only one period The length of period is defined by CM0 A write access to the register CN0 triggers the start of counting 0 _B One-shot mode disabled 1 _B One-shot mode enabled
BITREV	27	rw	Bit-reversing of output of counter register CN0 This bit enables the PCM mode of channel 15.
Reserved	[31:28]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.11.8.11 Register TOMi_CHx_CN0 (x:0...15)
GTM_TOM0_CHx_CN0 (x=0-15)
TOM0 Channel x CCU0 Counter Register
 $(08014_H + x * 0040_H)$
Reset Value: 00000000_H
GTM_TOM1_CHx_CN0 (x=0-15)
TOM1 Channel x CCU0 Counter Register
 $(08814_H + x * 0040_H)$
Reset Value: 00000000_H


Field	Bits	Type	Description
CN0	[15:0]	rw	TOM CCU0 counter register This counter is stopped if the TOM channel is disabled and not reset on an enable event of TOM channel.
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

26.11.8.12 Register TOMi_CHx_CM0 (x:0...15)
GTM_TOM0_CHx_CM0 (x=0-15)
TOM0 Channel x CCU0 Compare Register
 $(0800C_H + x * 0040_H)$
Reset Value: 00000000_H
GTM_TOM1_CHx_CM0 (x=0-15)
TOM1 Channel x CCU0 Compare Register
 $(0880C_H + x * 0040_H)$
Reset Value: 00000000_H


Field	Bits	Type	Description
CM0	[15:0]	rw	TOM CCU0 compare register Setting CM0 < CM1 configures a duty cycle of 100%.

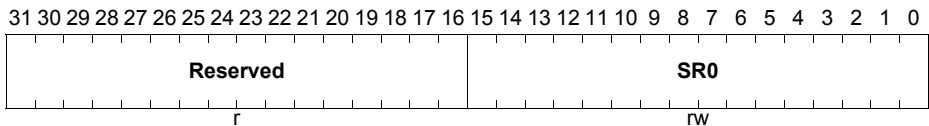
Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.11.8.13 Register TOMi_CHx_SR0 (x:0...15)
GTM_TOM0_CHx_SR0 (x=0-15)
TOM0 Channel x CCU0 Compare Shadow Register
 (08004_H+x*0040_H)

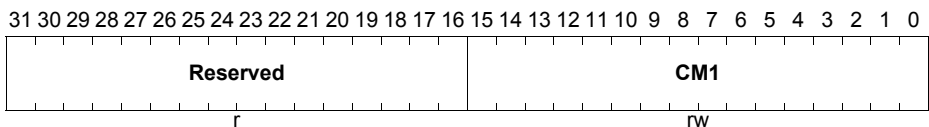
Reset Value: 00000000_H
GTM_TOM1_CHx_SR0 (x=0-15)
TOM1 Channel x CCU0 Compare Shadow Register
 (08804_H+x*0040_H)

Reset Value: 00000000_H


Field	Bits	Type	Description
SR0	[15:0]	rw	TOM channel x shadow register SR0 for update of compare register CM0
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

26.11.8.14 Register TOMi_CHx_CM1 (x:0...15)
GTM_TOM0_CHx_CM1 (x=0-15)
TOM0 Channel x CCU1 Compare Register
 (08010_H+x*0040_H)

Reset Value: 00000000_H
GTM_TOM1_CHx_CM1 (x=0-15)
TOM1 Channel x CCU1 Compare Register
 (08810_H+x*0040_H)

Reset Value: 00000000_H


Field	Bits	Type	Description
CM1	[15:0]	rw	TOM CCU1 compare register Setting CM1 = 0 configures a duty cycle of 0% independent of the configured value of CM0.

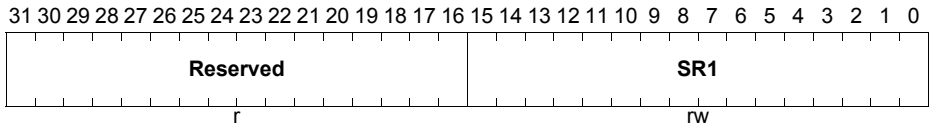
Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.11.8.15 Register TOMi_CHx_SR1 (x:0...15)
GTM_TOM0_CHx_SR1 (x=0-15)
TOM0 Channel x CCU1 Compare Shadow Register
 (08008_H+x*0040_H)

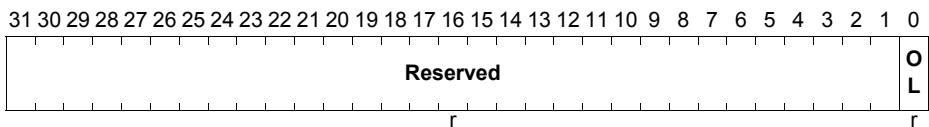
Reset Value: 00000000_H
GTM_TOM1_CHx_SR1 (x=0-15)
TOM1 Channel x CCU1 Compare Shadow Register
 (08808_H+x*0040_H)

Reset Value: 00000000_H


Field	Bits	Type	Description
SR1	[15:0]	rw	TOM channel x shadow register SR1 for update of compare register CM1
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

26.11.8.16 Register TOMi_CHx_STAT (x:0...15)
GTM_TOM0_CHx_STAT (x=0-15)
TOM0 Channel Status Register
 (08018_H+x*0040_H)

Reset Value: 00000000_H
GTM_TOM1_CHx_STAT (x=0-15)
TOM1 Channel Status Register
 (08818_H+x*0040_H)

Reset Value: 00000000_H


Field	Bits	Type	Description
OL	0	r	Output level of output TOM_OUT(x)
Reserved	[31:1]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.11.8.17 Register TOMi_CHx_IRQ_NOTIFY (x:0...15)

GTM_TOM0_CHx_IRQ_NOTIFY (x=0-15)

TOM0 Channel x Interrupt Notification Register

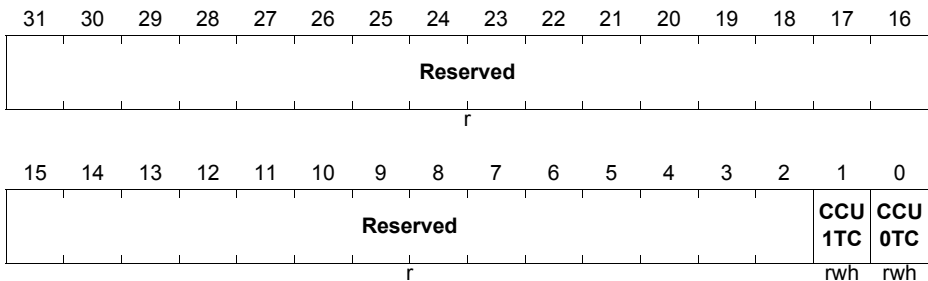
 $(0801C_H + x * 0040_H)$

 Reset Value: 00000000_H

GTM_TOM1_CHx_IRQ_NOTIFY (x=0-15)

TOM1 Channel x Interrupt Notification Register

 $(0881C_H + x * 0040_H)$

 Reset Value: 00000000_H


Field	Bits	Type	Description
CCU0TC	0	rwh	CCU0 Trigger condition interrupt for channel x 0 _B No interrupt occurred 1 _B The condition CN0 >= CM0 was detected. The notification of the interrupt is only triggered one time after reaching the condition CN0 >= CM0. To re-trigger the notification first the condition CN0 < CM0 has to be occurred.
CCU1TC	1	rwh	CCU1 Trigger condition interrupt for channel x 0 _B No interrupt occurred 1 _B The condition CN0 >= CM1 was detected. The notification of the interrupt is only triggered one time after reaching the condition CN0 >= CM1. To re-trigger the notification first the condition CN0 < CM1 has to be occurred.
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.11.8.18 Register TOMi_CHx_IRQ_EN (x:0...15)

GTM_TOM0_CHx_IRQ_EN (x=0-15)

TOM0 Channel x Interrupt Enable Register

(08020_H+x*0040_H)

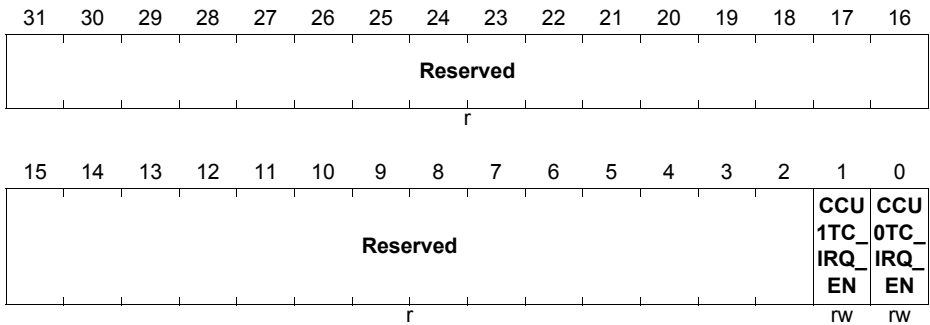
Reset Value: 00000000_H

GTM_TOM1_CHx_IRQ_EN (x=0-15)

TOM1 Channel x Interrupt Enable Register

(08820_H+x*0040_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CCU0TC_IRQ_EN	0	rw	TOM_CCU0TC_IRQ interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
CCU1TC_IRQ_EN	1	rw	TOM_CCU1TC_IRQ interrupt enable See bit 0
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.11.8.19 Register TOMi_CHx_IRQ_FORCINT (x:0...15)

GTM_TOM0_CHx_IRQ_FORCINT (x=0-15)

TOM0 Channel x Software Interrupt Generation Register

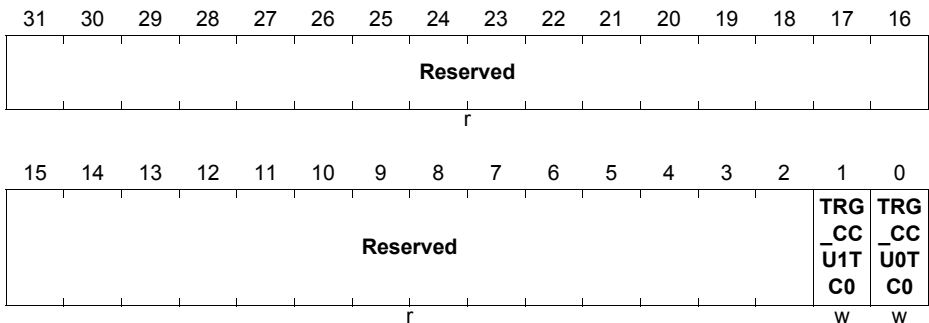
 $(08024_H + x * 0040_H)$

 Reset Value: 00000000_H

GTM_TOM1_CHx_IRQ_FORCINT (x=0-15)

TOM1 Channel x Software Interrupt Generation Register

 $(08824_H + x * 0040_H)$

 Reset Value: 00000000_H


Field	Bits	Type	Description
TRG_CCU0TC0	0	w	Trigger TOM_CCU0TC0_IRQ interrupt by software 0 _B No interrupt triggering 1 _B Assert CCU0TC0_IRQ interrupt for one clock cycle Read as 0. <i>Note: This bit is cleared automatically after interrupt is released</i> Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
TRG_CCU1TC0	1	w	Trigger TOM_CCU1TC0_IRQ interrupt by software 0 _B No interrupt triggering 1 _B Assert CCU1TC0_IRQ interrupt for one clock cycle Read as 0. <i>Note: This bit is cleared automatically after write.</i> Note: This bit is write protected by bit GTM_CTRL.RF_PROT.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.11.8.20 Register TOMi_CHx_IRQ_MODE (x:0...15)

GTM_TOM0_CHx_IRQ_MODE (x=0-15)

TOM0 IRQ Mode Configuration Register

(08028_H+x*0040_H)

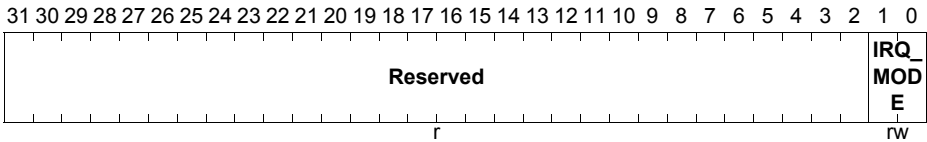
Reset Value: 00000000_H

GTM_TOM1_CHx_IRQ_MODE (x=0-15)

TOM1 IRQ Mode Configuration Register

(08828_H+x*0040_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
IRQ_MODE	[1:0]	rw	IRQ mode selection 00 _B Level mode 01 _B Pulse mode 10 _B Pulse-Notify mode 11 _B Single-Pulse mode Note: The interrupt modes are described in Section 26.2.5 .
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

26.12 ARU-connected Timer Output Module (ATOM)

26.12.1 Overview

The ARU-connected Timer Output Module (ATOM) is able to generate complex output signals without CPU interaction due to its connectivity to the ARU. Typically, output signal characteristics are provided over the ARU connection through submodules connected to ARU like e.g. the MCS, DPLL or PSM. Each ATOM submodule contains eight output channels which can operate independently from each other in several configurable operation modes. A block diagram of the ATOM submodule is depicted in [Figure 26-42](#).

26.12.1.1 ATOM Block diagram

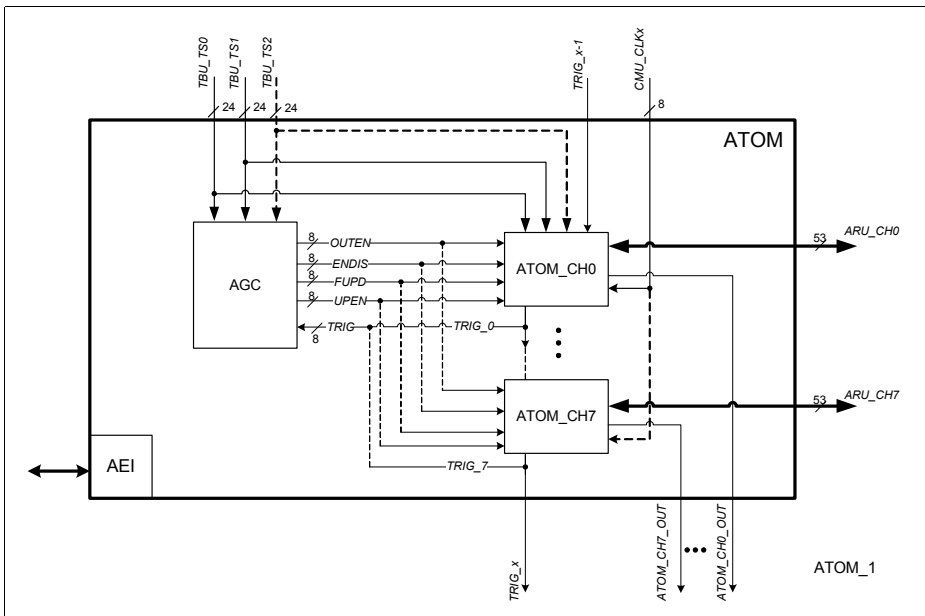


Figure 26-42 ATOM Block diagram

The architecture of the ATOM submodule is similar to the TOM submodule, but there are some differences. First, the ATOM integrates only eight output channels. Hence, there exists one ATOM Global Control subunit (AGC) for the ATOM channels. The ATOM is

Generic Timer Module (GTM)

connected to the ARU and can set up individual read requests from the ARU and write requests to the ARU. Furthermore, the ATOM channels are able to generate signals on behalf of time stamps and the ATOM channels are able to generate a serial output signal on behalf of an internal shift register.

Each ATOM channel provides four modes of operation:

- ATOM Signal Output Mode Immediate (SOMI)
- ATOM Signal Output Mode Compare (SOMC)
- ATOM Signal Output Mode PWM (SOMP)
- ATOM Signal Output Mode Serial (SOMS)

These modes are described in more detail in [Section 26.12.3](#).

The ATOM channels' operation registers (e.g. counter, compare registers) are 24 bit wide. Moreover, the input clocks for the ATOM channels come from the configurable CMU_CLKx signals of the CMU submodule. This gives the freedom to select a programmable input clock for the ATOM channel counters. The ATOM channel is able to generate a serial bit stream, which is shifted out at the ATOM[i]_CH[x]_OUT output. When configured in this serial shift mode (SOMS) the selected CMU clock defines the shift frequency.

Each ATOM channel provides a so called operation and shadow register set. With this architecture it is possible to work with the operation register set, while the shadow register set can be reloaded with new parameters over CPU and/or ARU.

When update via ARU is selected, it is possible to configure if both shadow registers are updated via ARU or only one of the shadow registers is updated for SOMP mode.

On the other hand, the shadow registers can be used to provide data to the ARU when one or both of the compare units inside an ATOM channel match. This feature is only applicable in SOMC mode.

In TOM channels it is possible to reload the content of the operation registers with the content of the corresponding shadow registers and change the clock input signal for the counter register simultaneously. This simultaneous change of the input clock frequency together with reloading the operation registers is also implemented in the ATOM channels.

In addition to the feature that the CPU can select another CMU_CLKx during operation (i.e. updating the shadow register bit field CLK_SRC_SR of the ATOMi_CHx_CTRL register), the selection can also be changed via the ARU. Then, for the clock source update, the ACBI register bits of the ATOMi_CHx_STAT register are used as a shadow register for the new clock source.

In general, the behaviour of the compare units CCU0 and CCU1 and the output signal behaviour is controlled with the ACB bit field inside the ATOMi_CHx_CTRL register when the ARU connection is disabled and the behaviour is controlled via ARU through the ACBI bit field of the ATOMi_CHx_STAT register, when the ARU is enabled.

Generic Timer Module (GTM)

Since the ATOM is connected to the ARU, the shadow registers of an ATOM channel can be reloaded via the ARU connection or via CPU over its AEI interface. When loaded via the ARU interface, the shadow registers act as a buffer between the ARU and the channel operation registers. Thus, a new parameter set for a PWM can be reloaded via ARU into the shadow registers, while the operation registers work on the actual parameter set.

26.12.1.2 ATOM Global control (AGC)

Synchronous start and stop of more than one output channel is possible with the AGC subunit. This subunit has the same functionality as the TGC subunit of the TOM submodule. For a description of the AGC subunit functionality, please refer therefore to [Section 26.11.2](#).

26.12.1.3 ATOM Channel mode overview

Each ATOM channel offers the following different operation modes.

In ATOM Signal Output Mode Immediate (SOMI), the ATOM channels generate an output signal immediately after receiving an ARU word according to the two signal level output bits of the ARU word received through the ACBI bit field. Due to the fact, that the ARU destination channels are served in a round robin order, the output signal can jitter in this mode with a jitter of the ARU round trip time.

In ATOM Signal Output Mode Compare (SOMC), the ATOM channel generates an output signal on behalf of time stamps that are located in the ATOM operation registers. These time stamps are compared with the time stamps, the TBU generates. The ATOM is able to receive new time stamps either by CPU or via the ARU. The new time stamps are directly loaded into the channels operation register. The shadow registers are used as capture registers for the two time base values, when a compare match of the channels operation registers occurs.

In ATOM Signal Output Mode PWM (SOMP), the ATOM channel is able to generate simple and complex PWM output signals like the TOM submodule by comparing its operation registers with a submodule internal counter. In difference to the TOM, the ATOM shadow registers can be reloaded by the CPU and by the ARU in the background, while the channel operates on the operation registers.

In ATOM Signal Output Mode Serial (SOMS), the ATOM channel generates a serial output bit stream on behalf of a shift register. The number of bits shifted and the shift direction is configurable. The shift frequency is determined by one of the CMU_CLKx clock signals. Please refer to [Section 26.12.3.4](#) for further details.

26.12.2 ATOM Channel architecture

Each ATOM channel is able to generate output signals according to four operation modes. The architecture of the ATOM channels is similar to the architecture of the TOM channels. The general architecture of an ATOM channel is depicted in **Figure 26-43**.

26.12.2.1 ATOM Channel architecture

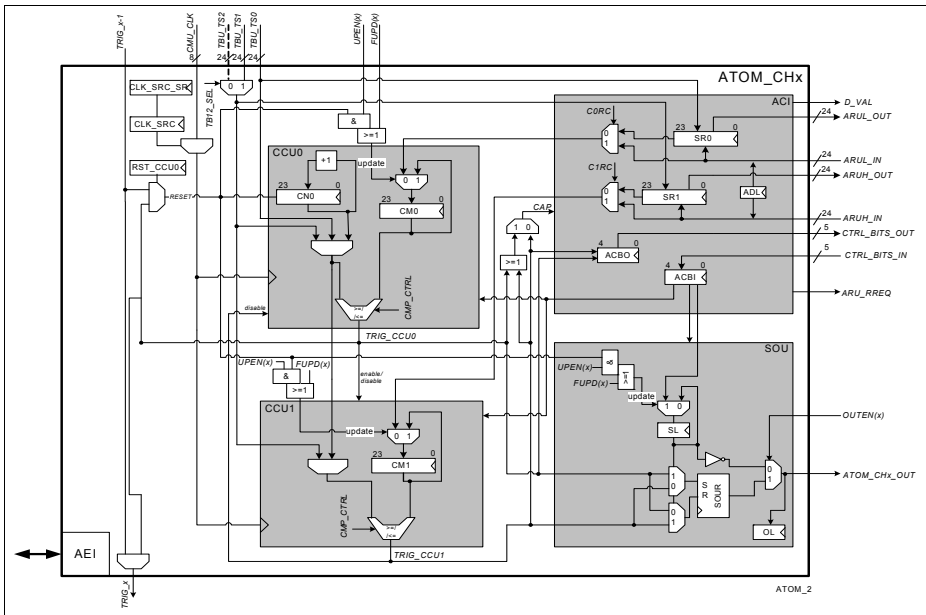


Figure 26-43 ATOM Channel architecture

For all ATOM channels the bit width is 24 of the operation register CN0, CM0 and CM1 and the shadow register SR0 and SR1. The comparators inside CCU0 and CCU1 provide a selectable signed greater/equal or less/equal comparison to compare against the GTM time bases TBU_TS0 and TBU_TS1. If there is a third time base TBU_TS2 implemented inside the GTM, this time base can also be selected inside the ATOM channel with the TB12_SEL bit inside the ATOMi_CHx_CTRL register for comparison. Please refer to TBU **Section 26.9** for further details. For an overview of the implemented TBU submodule version please refer to **Section 26.2.1.1**. The CCU0 and CCU1 units have different tasks for the different ATOM channel modes.

The signed compare is used to detect time base overflows and to guarantee, that a compare match event can be set up for the future even when the time base will first

Generic Timer Module (GTM)

overflow and then reach the compare value. Please note, that for a correct behaviour of this signed compare, the new compare value must not be specified larger/smaller than half of the range of the total time base value (0x7FFFFFFF).

In SOMC mode, the two compare units CCUx can be used in combination to each other. When used in combination, the trigger lines TRIG_CCU0 and TRIG_CCU1 can be used to enable/disable the other compare unit on a match event. Please refer to [Section 26.12.3.2](#) for further details.

The Signal Output Unit (SOU) generates the output signal for each ATOM channel. This output signal level depends on the ATOM channel mode and on the SL bit of the ATOMi_CHx_CTRL register in combination with the two control bits. These two control bits ACB(1) and ACB(0) can either be received via CPU in the ACB register field of the ATOMi_CHx_CTRL register or via ARU in the ACBI bit field of the ATOMi_CHx_STAT register.

The SL bit in the ATOMi_CHx_CTRL register defines in all modes the operational behaviour of the ATOM channel.

When the channel and its output is disabled, the output signal level of the channel is the inverse of the SL bit.

In SOMI and SOMC mode the output signal level depends on the SL, ACB0 and ACB1 bits. In SOMP mode the output signal level depends on the two trigger signals TRIG_CCU0 and TRIG_CCU1 since these two triggers define the PWM timing characteristics and the SL bit defines the level of the duty cycle. In SOMS mode the output signal level is defined by the bit pattern that has to be shifted out by the ATOM channel. The bit pattern is located inside the CM1 register.

The ARU Communication Interface (ACI) subunit is responsible for requesting data routed through ARU to the ATOM channel in SOMI, SOMP and SOMS modes, and additionally for providing data to the ARU in SOMC mode.

In SOMC mode the ACI shadow registers have a different behaviour and are used as output buffer registers for data send to ARU.

26.12.2.2 ARU Communication Interface

The ATOM channels have an ARU Communication Interface (ACI) subunit. This subunit is responsible for data exchange from and to the ARU. This is done with the two implemented registers SR0, SR1, and the ACBI and ACBO bit fields that are part of the ATOMi_CHx_STAT register. The ACI architecture is shown in [Figure 26-44](#).

If the ARU_EN bit is set inside the ATOMi_CHx_CTRL register, the ATOM channel is enabled by setting the enable bits inside the ATOMi_AGC_ENDIS_STAT register and the CPU hasn't written data not equal to zero into the CM0, CM1, SR0, SR1 register, the ATOM channel will first request data from the ARU before the signal generation starts in SOMP, SOMS and SOMC mode.

Generic Timer Module (GTM)

Note: if in SOMP mode there is data inside the CM0 or SR0 register not equal to '0' the channel counter CN0 will start counting immediately, regardless whether the channel has received ARU data yet.

Note: if in SOMS mode there is data inside the CM0 or SR0 register not equal to '0' the channel will start shifting immediately, regardless whether the channel has received ARU data yet.

ACI Architecture overview

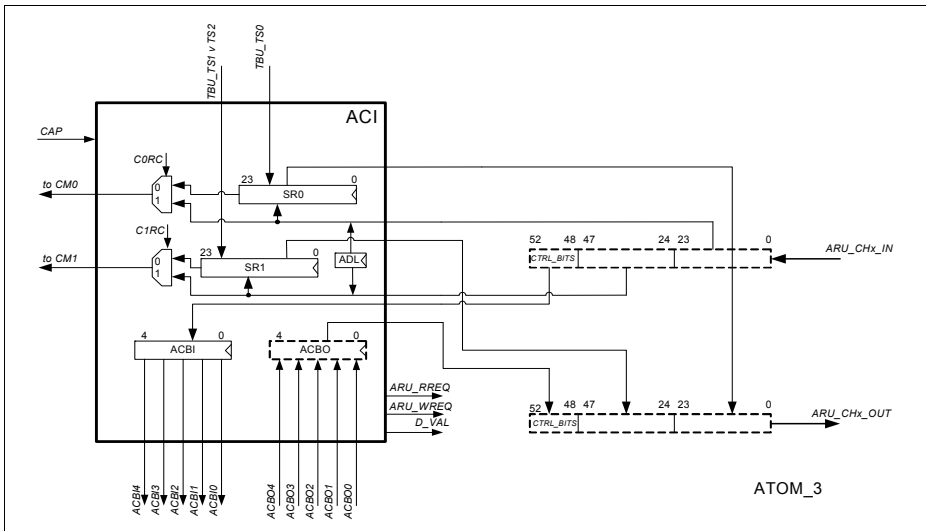


Figure 26-44 ACI Architecture overview

Incoming ARU data (53 bit width signal ARU_CHx_IN) is split into three parts by the ACI and communicated to the ATOM channel registers. In SOMI, SOMP and SOMS modes incoming ARU data ARU_CHx_IN is split in a way that the lower 24 bits of the ARU data (23 down to 0) are stored in the SR0 register, the upper bits (47 down to 24) are stored in the SR1 register and the bits 52 down to 48 (CTRL_BITS) are stored in the ACBI bit field of the register ATOMi_CHx_STAT.

The ATOM channel has to ensure, that in a case when the channel operation registers CM0 and CM1 are updated with the SR0 and SR1 register content and an ARU transfer to these shadow registers happens in parallel that either the old data in both shadow registers is transferred into the operation registers or both new values from the ARU are transferred.

Generic Timer Module (GTM)

In SOMC mode incoming ARU data ARU_CHx_IN is written directly to the ATOM channel operation register in the way that the lower 24 bits (23 down to 0) are written to CM0, and the bits 47 down to 24 are written to register CM1. The bits 52 down to 48 are stored in the ACBI bit field of the ATOMi_CHx_STAT register and control the behaviour of the compare units and the output signal of the ATOM channel.

In SOMC mode the SR0 and SR1 registers serve as capture registers for the time stamps coming from TBU whenever a compare match event is signalled by the CCU0 and/or CCU1 subunits via the CAP signal line. These two time stamps are then provided together with actual ATOM channel status information located in the ACBO bit field to the ARU at the dedicated ARU write address of the ATOM channel when the ARU is enabled.

The encoding of the ARU control bits in the different ATOM operation modes is described in more detail in the following sections.

26.12.3 ATOM Channel modes

As described above, each ATOM channel can operate independently from each other in one of four dedicated output modes:

- ATOM Signal Output Mode Immediate (SOMI)
- ATOM Signal Output Mode Compare (SOMC)
- ATOM Signal Output Mode PWM (SOMP)
- ATOM Signal Output Mode Serial (SOMS)

The Signal Output Mode PWM (SOMP) is principally the same like the output mode for the TOM submodule except the bit erase mode which is not included in the ATOM. In addition, it is possible to reload the shadow registers over the ARU without the need of a CPU interaction. The three other modes provide additional functionality for signal output control. All operation modes are described in more detail in the following sections.

26.12.3.1 ATOM Signal Output Mode Immediate (SOMI)

In ATOM Signal Output Mode Immediate (SOMI), the ATOM channel generates output signals on the ATOM[i]_CH[x]_OUT output port immediate after update of the bit ACBI(0) of register ATOMi_CHx_STAT or ACB(0) bit of register ATOMi_CHx_CTRL.

If ARU access is enabled by setting bit ARU_EN in register ATOMi_CHx_CTRL, the update of the output ATOM[i]_CH[x]_OUT depends on the bit ACBI(0) of register ATOMi_CHx_STAT received at the ACI subunit and the bit SL bit of register ATOMi_CHx_CTRL. The remaining 48 ARU bits (47 down to 0) have no meaning in this mode.

If ARU access is disabled, the update of the output ATOM[i]_CH[x]_OUT depends on the bit ACB(0) and the bit SL of register ATOMi_CHx_CTRL.

The initial ATOM channel port pin ATOM[i]_CH[x]_OUT signal level has to be specified by the SL bit field (

Generic Timer Module (GTM)

=0 defined in this mode, see [Figure 26-43](#)) of the ATOMi_CHx_CTRL register when OUTEN_CTRL register bit field OUTEN_CTRLx is disabled (see [Chapter 26.11.8.9](#)) for details.

In SOMI mode the output behaviour depends on the SL bit of register ATOMi_CHx_CTRL and the bit ACBI(0) of the ATOMi_CHx_STAT register or the bit ACB0 of register ATOMi_CHx_CTRL:

Table 26-33 Signal Output Mode Immediate Status information

SL	ACBI(0)/ ACB(0)	Output behaviour
0	0	Set output to inverse of SL (1)
0	1	Set output to SL (0)
1	0	Set output to inverse of SL (0)
1	1	Set output to SL (1)

The signal level bit ACBI(0) is transferred to the SOU subunit of the ATOM and made visible at the output port according to the table above immediately after the data was received by the ACI. This can introduce a jitter on the output signal since the ARU channels are served in a time multiplexed fashion.

Generic Timer Module (GTM)

Register ATOMi_CHx_CTRL in SOMI mode (x: 0...7)

GTM_ATOM0_CHx_CTRL_SOMI (x=0-7)

 ATOM0 Channel x Control in SOMI mode Register
 (0D004_H+x*0080_H)

 Reset Value: 00000800_H

GTM_ATOM1_CHx_CTRL_SOMI (x=0-7)

 ATOM1 Channel x Control in SOMI mode Register
 (0D804_H+x*0080_H)

 Reset Value: 00000800_H

GTM_ATOM2_CHx_CTRL_SOMI (x=0-7)

 ATOM2 Channel x Control in SOMI mode Register
 (0E004_H+x*0080_H)

 Reset Value: 00000800_H

GTM_ATOM3_CHx_CTRL_SOMI (x=0-7)

 ATOM3 Channel x Control in SOMI mode Register
 (0E804_H+x*0080_H)

 Reset Value: 00000800_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			NotUsed	NotUsed	Reserved	NotUsed	NotUsed			NotUsed	Reserved			NotUsed	
r			rw	rw	r	rw	rw			rw	r			rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	NotUsed		SL	Reserved		NotUsed			ACB0	ARU_EN	NotUsed	MODE			
r	rw		rw	r		rw			rw	rw	rw	rw			

Field	Bits	Type	Description
MODE	[1:0]	rw	ATOM channel mode select 00 _B ATOM Signal Output Mode Immediate (SOMI)
NotUsed	2	rw	Not used in this mode
ARU_EN	3	rw	ARU Input stream enable 0 _B ARU Input stream disabled 1 _B ARU Input stream enabled
ACB0	4	rw	ACB bit 0 0 _B Set output to inverse of SL bit 1 _B Set output to SL bit
NotUsed	[8:5]	rw	Not used in this mode
Reserved	[10:9]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

Field	Bits	Type	Description
SL	11	rw	Initial signal level after channel is enabled 0_B Low signal level 1_B High signal level Note: After reset and if channel is disabled, the register SOUR is set to the inverse reset value of bit SL (i.e. '1'). If the channel is disabled or the output is disabled, the output ATOM_OUT[x] is set to inverse value of SL.
NotUsed	[14:12]	rw	Not used in this mode
Reserved	15	r	Reserved Read as zero, should be written as zero.
NotUsed	16	rw	Not used in this mode
Reserved	[19:17]	r	Reserved Read as zero, should be written as zero.
NotUsed	20	rw	Not used in this mode
NotUsed	[23:21]	r	Not used in this mode
NotUsed	24	rw	Not used in this mode
Reserved	25	r	Reserved Read as zero, should be written as zero.
NotUsed	26	rw	Not used in this mode
NotUsed	27	rw	Not used in this mode
Reserved	[31:28]	r	Reserved Read as zero, should be written as zero.

26.12.3.2 ATOM Signal Output Mode Compare (SOMC)

Overview

In ATOM Signal Output Mode Compare (SOMC) the output action is performed in dependence of the comparison between input values located in CM0 and/or CM1 registers and the two (three) time base values TBU_TS0 or TBU_TS1 (or TBU_TS2) provided by the TBU. For a description of the time base generation please refer to the TBU specification in [Section 26.9](#). It is configurable, which of the two (three) time bases is to be compared with one or both values in CM0 and CM1.

The behaviour of the two compare units CCU0 and CCU1 is controlled either with the bits 4 down to 2 of ACB bit field inside the ATOMi_CHx_CTRL register, when the ARU connection is disabled or with the ACBI bit field of the ATOMi_CHx_STAT register, when

Generic Timer Module (GTM)

the ARU is enabled. In that case the ACB bit field is updated via the ARU control bits 52 down to 48.

The CCUx trigger signals TRIG_CCU0 and TRIG_CCU1 always create edges, dependent on the predefined signal level in SL bit in combination with two control bits that can be specified by either ARU or CPU within the aforementioned ATOMi_CHx_CTRL or ATOMi_CHx_STAT registers.

In SOMC mode the channel is always disabled after the specified compare match event occurred. The shadow registers are used to store two time stamp values at the match time. The channel can be enabled again by first reading the shadow registers, either by CPU or ARU and by providing new data for CMx registers through CPU or ARU. For a detailed description please refer to [SOMC Mode under CPU control](#) and [SOMC Mode under ARU control](#).

If three time bases exist for the GTM-IP there must be a preselection between TBU_TS1 and TBU_TS2 for the ATOM channel. This can be done with TB12_SEL bit in the ATOMi_CHx_CTRL register.

The comparison in CCU0/1 with time base TBU_TS1 or TBU_TS2 can be done on a greater/equal or less/equal compare according to the CMP_CTRL bit. This control bit has no effect to a compare unit CCU0 or CCU1 that compares against TBU_TS0. In this case always a greater/equal compare is done. The bit CMP_CTRL is part of the ATOMi_CHx_CTRL register.

When configured in SOMC mode, the channel port pin has to be initialized to an initial signal level. This initial level after enabling the ATOM channel is determined by the SL bit in the ATOMi_CHx_CTRL register. If the output is disabled, the signal level is set to the inverse level of the SL bit.

If the channel is disabled, the register SOUR is set to the SL bit in the ATOMi_CHx_CTRL register.

On a compare match event the shadow register SR0 and SR1 are used to capture the TBU time stamp values. SR0 always holds TBU_TS0 and SR1 either holds TBU_TS1 or TBU_TS2 dependent on the TB12_SEL bit in the ATOMi_CHx_CTRL register.

Please note, that when the channel is disabled and the compare registers are written, the compare registers CMx are loaded with the written value and the channel starts with the comparison on behalf of this values, when the channel is enabled.

SOMC Mode under CPU control

As already mentioned above the ATOM channel can be controlled either by CPU or by ARU. When the channel should be controlled by CPU, the ARU_EN bit inside the ATOMi_CHx_CTRL register has to be reset.

The output of the ATOM channel is set on a compare match event depending on the ACB10 bit field in combination with the SL bit both located in the ATOMi_CHx_CTRL

Generic Timer Module (GTM)

register. The output behaviour according to the ACB10 bit field in the control register is shown in the following table:

Table 26-34 SOMC Output behaviour

SL	ACB10(5)	ACB10(4)	Output behaviour
0	0	0	No signal level change at output (exception in Figure 26-45 mode ACB42=001)
0	0	1	Set output signal level to 1
0	1	0	Set output signal level to 0
0	1	1	Toggle output signal level (exception in Figure 26-45 mode ACB42=001)
1	0	0	No signal level change at output (exception in Figure 26-45 mode ACB42=001)
1	0	1	Set output signal level to 0
1	1	0	Set output signal level to 1
1	1	1	Toggle output signal level (exception in Figure 26-45 mode ACB42=001)

The capture/compare strategy of the two CCUx units can be controlled with the ACB42 bit field inside the ATOMi_CHx_CTRL register. The meaning of these bits is shown in the following table:

Table 26-35 Bit definitions for Capture / Compare operation

ACB42(8)	ACB42(7)	ACB42(6)	CCUx control
0	0	0	Serve First: Compare in CCU0 using TBU_TS0 and in parallel in CCU1 using TBU_TS1 or TBU_TS2. Disable other CCUx on compare match. Output signal level on the compare match of the matching CCUx unit is defined by combination of SL, ACB10(5) and ACB10(4). Details see Figure 26-45
0	0	1	Serve First: Compare in CCU0 using TBU_TS0 and in parallel in CCU1 using TBU_TS1 or TBU_TS2. Disable other CCUx on compare match. Output signal level on the compare match of the matching CCUx unit is defined by combination of SL, ACB10(5) and ACB10(4). Details see Figure 26-45

Generic Timer Module (GTM)
Table 26-35 Bit definitions for Capture / Compare operation (cont'd)

ACB42(8)	ACB42(7)	ACB42(6)	CCUx control
0	1	0	Compare in CCU0 only, use time base TBU_TS0. Output signal level is defined by combination of SL, ACB10(5) and ACB10(4) bits.
0	1	1	Compare in CCU1 only, use time base TBU_TS1 or TBU_TS2. Output signal level is defined by combination of SL, ACB10(5) and ACB10(4) bits.
1	0	0	Serve Last: Compare in CCU0 and then in CCU1 using TBU_TS0. Output signal level when CCU0 matches is defined by combination of SL, ACB10(5) and ACB10(4). On the CCU1 match the output level is toggled.
1	0	1	Serve Last: Compare in CCU0 and then in CCU1 using TBU_TS1 or TBU_TS2. Output signal level when CCU0 matches is defined by combination of SL, ACB10(5) and ACB10(4). On the CCU1 match the output level is toggled.
1	1	0	Serve Last: Compare in CCU0 using TBU_TS0 and then in CCU1 using TBU_TS1 or TBU_TS2. Output signal level when CCU1 matches is defined by combination of SL, ACB10(5) and ACB10(4).
1	1	1	Not used when ARU disabled.

The behaviour of the ACBI/ACB42 bit combinations '000' and '001' is described in more detail in [Figure 26-45](#).

ATOM CCUx Serve first definition

Generic Timer Module (GTM)

ACB4	ACB3	ACB2	ACB1	ACB0	SL	CCU0 match	CCU1 match	Pin level new
0	0	0	0	0	0	0	1	hold
							1	hold
							1	hold
0	0	0	0	1	0	0	1	1
							1	1
							1	1
0	0	0	1	0	0	0	1	0
							1	0
							1	0
0	0	0	1	1	0	0	1	toggle
							1	toggle
							1	toggle
0	0	0	0	0	1	0	1	hold
							1	hold
							1	hold
0	0	0	0	1	1	0	1	0
							1	0
							1	0
0	0	0	1	0	1	0	1	1
							1	0
							1	1
0	0	0	1	1	1	0	1	toggle
							1	toggle
							1	toggle
0	0	1	0	0	0	0	1	hold
							1	toggle
							1	hold
0	0	1	0	1	0	0	1	0
							1	0
							1	1
0	0	1	1	0	0	0	1	1
							1	0
							1	0
0	0	1	1	1	0	0	1	toggle
							1	hold
							1	toggle
0	0	1	0	0	1	0	1	hold
							1	toggle
							1	hold
0	0	1	0	1	1	0	1	1
							1	0
							1	1
0	0	1	1	0	1	0	1	0
							1	0
							1	1
0	0	1	1	1	1	0	1	toggle
							1	hold
							1	toggle

ATOM_tab

Figure 26-45 ATOM CCUx Serve first definition

Generic Timer Module (GTM)

If the ATOM channel is enabled, the CM0 and/or CM1 registers and the ACB42 bit field of the ATOMi_CHx_CTRL register can be updated by the CPU as long as the first match event occurs in case of a serve last compare strategy or as long as the overall match event in case of the other compare strategies.

After a compare match event that causes an update of the shadow registers SR0/SR1 and before reading the SR0 and/or SR1 register via ARU, the update of the registers CM0 and/or CM1 is possible but has no effect.

To set up a new compare action, first the SR0 and/or SR1 register containing captured values have to be read and then new compare values have to be written into the register CM0 and/or CM1.

Which CMx register has to be updated depends on the compare strategy defined in the ACB42 bit field of the channel control register. Since the channel immediately starts with the comparison after the CMx register was/were written, the compare strategy has to be updated before the CMx registers are written.

For the serve last compare strategies, if the register CM0 and CM1 are updated, it can happen that one or both compare values are already located in the past. In any way the ATOM channel will first wait until both compare values are written, before it starts the time base comparisons to avoid a deadlock.

The CPU can check at any time if at least one of the ATOM channels' capture compare registers contains valid data and waits for a compare event to happen. This is signalled by the DV bit inside the ATOMi_CHx_STAT register.

Note: For serve last compare strategies, if DV bit is currently not set, writing to CM0 or CM1 sets immediately the DV bit although the compare is only started if both values are written.

In SOMC mode and CCUx control mode 'serve last' exist an exception for update of register CM0/CM1. If in this mode the CCU0 compare match event occurred, the update of register CM0/CM1 via CPU is blocked until the CCU1 compare match event.

In the serve last mode (ACB42="100" or ACB42="101") it is possible to generate very small spikes on the output pin by loading CM0 and CM1 with two time stamp values for TBU_TS0, TBU_TS1 or TBU_TS2 close together. The output pin will then be set or reset dependent on the SL bit and the specified ACB10(5) and ACB10(4) bits in the ACB10 bit field of the ATOMi_CHx_CTRL register on the first match event and the output will toggle on the second compare event in the CCU1 compare unit.

It is important to note, that the bigger (smaller) time stamp has to be loaded into the CM1 register, since the CCU0 will enable the CCU1 once it has reached its comparison time stamp. The order of the comparison time stamps depends on the defined greater/equal or less/equal comparison of the CCUx units.

In addition to storing the captured time stamps in the shadow registers, the ATOM channel provides the result of the compare match event in the ACBO(4) and ACBO(3)

Generic Timer Module (GTM)

bits of the ATOMi_CHx_STAT register. The meaning of the bits is shown in the following table:

Table 26-36 ACBO3 / 4 Bit encoding

ACBO(4)	ACBO(3)	Indication
0	1	CCU0 compare match occurred
1	0	CCU1 compare match occurred

Please note, that in case of the 'serve last' compare strategy, when the SLA-bit in the ATOMi_CHx_CTRL register is not set, the ACBO(4) bit is always set and the ACBO(3) bit is always reset after the compare match event occurred.

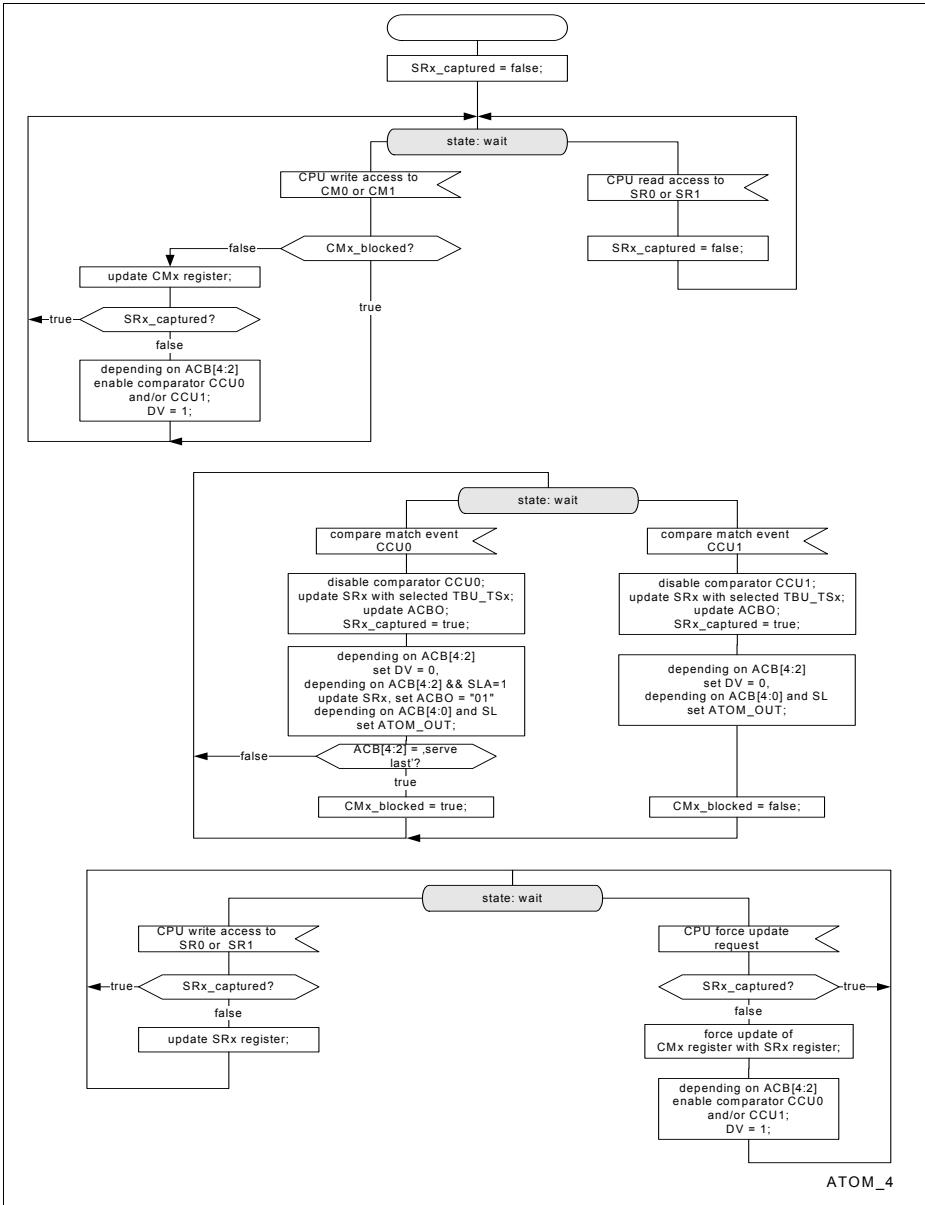
The ACBO bit field is reset, when the DV bit is set.

Depending on the capture compare unit where the time base matched the interrupt CCU0TCx_IRQ or CCU1TCx_IRQ is raised.

The behaviour of an ATOM channel in SOMC mode under CPU control is visualized in [Figure 26-46](#).

SOMC state diagram for channel under CPU control

Generic Timer Module (GTM)



ATOM_4

Figure 26-46 SOMC state diagram

Generic Timer Module (GTM)
SOMC Mode under ARU control

When the channel should be controlled by ARU, the ARU_EN bit inside the ATOMi_CHx_CTRL register has to be set.

In case, the ATOM channel is under ARU control the content for the compare registers CM0 and CM1 as well as the update of the compare strategy can be loaded via the 53 bit ARU word.

The ARU word 23 to 0 is loaded into the CM0 register while the ARU word 47 to 24 is loaded into the CM1 register. The five ARU control bits 52 to 48 are loaded into the ACBI bit field of the ATOMi_CHx_STAT register and control the channel compare strategy as well as the output behaviour in case of compare match events.

For the five ARU control bits 52 to 48 the bits 49 and 48 are loaded into the ACBI bits 1 and 0. The output behaviour also depends on the setting of the SL bit inside of the ATOMi_CHx_CTRL register and is shown in the following table:

Table 26-37 SL Bit encoding

SL	ACBI(1)	ACBI(0)	Output behaviour
0	0	0	No signal level change at output (exception in Figure 26-45 mode ACB42=001)
0	0	1	Set output signal level to 1
0	1	0	Set output signal level to 0
0	1	1	Toggle output signal level (exception in Figure 26-45 mode ACB42=001)
1	0	0	No signal level change at output (exception in Figure 26-45 mode ACB42=001)
1	0	1	Set output signal level to 0
1	1	0	Set output signal level to 1
1	1	1	Toggle output signal level (exception in Figure 26-45 mode ACB42=001)

For the five ARU control bits 52 to 48 the bits 52 to 50 are loaded into the ACBI bits 4 to 2. With these three bits the capture/compare units CCUX can be controlled as shown in the following table:

Table 26-38 CCU control via ACBI[4:2] control bits

ACBI(4)	ACBI(3)	ACBI(2)	CCUx control
0	0	0	Serve First: Compare in CCU0 using TBU_TS0 and in parallel in CCU1 using TBU_TS1 or TBU_TS2. Disable other CCUx on compare match. Output signal level on the compare match of the matching CCUx unit is defined by combination of SL, ACBI(1) and ACBI(0). Details see Figure 26-45
0	0	1	Serve First: Compare in CCU0 using TBU_TS0 and in parallel in CCU1 using TBU_TS1 or TBU_TS2. Disable other CCUx on compare match. Output signal level on the compare match of the matching CCUx unit is defined by combination of SL, ACBI(1) and ACBI(0). Details see Figure 26-45
0	1	0	Compare in CCU0 only, use time base TBU_TS0. Output signal level is defined by combination of SL, ACBI(1) and ACBI(0) bits.
0	1	1	Compare in CCU1 only, use time base TBU_TS1 or TBU_TS2. Output signal level is defined by combination of SL, ACBI(1) and ACBI(0) bits.
1	0	0	Serve Last: Compare in CCU0 and then in CCU1 using TBU_TS0. Output signal level when CCU0 matches is defined by combination of SL, ACBI(1) and ACBI(0). On the CCU1 match the output level is toggled.
1	0	1	Serve Last: Compare in CCU0 and then in CCU1 using TBU_TS1 or TBU_TS2. Output signal level when CCU0 matches is defined by combination of SL, ACBI(1) and ACBI(0). On the CCU1 match the output level is toggled.
1	1	0	Serve Last: Compare in CCU0 using TBU_TS0 and then in CCU1 using TBU_TS1 or TBU_TS2. Output signal level when CCU1 matches is defined by combination of SL, ACBI(1) and ACBI(0).
1	1	1	Change ARU read address to ATOM_RDADDR1 DV flag is not set. Neither ACBI(1) nor ACBI(0) is evaluated.

Generic Timer Module (GTM)

It is important to note that the bit combination "111" for the ACBI(4), ACBI(3) and ACBI(2) bits forces the channel to request new compare values from another destination read address defined in the ATOM_RDADDR1 bit field of the ATOMi_CHx_RDADDR register. After data was successfully received and the compare event occurred the ATOM channel switches back to ATOM_RDADDR0 to receive the next data from there.

After the specified compare match event, the captured time stamps are stored in SR0 and SR1 and the compare result is stored in the ACBO bit field of the ATOMi_CHx_STAT register. The meaning of the ACBO(4) and ACBO(3) bits of the ATOMi_CHx_STAT is shown in the following table:

Table 26-39 ACBO[4:3] bid encoding

ACBO(4)	ACBO(3)	Return value to ARU
0	1	CCU0 compare match occurred
1	0	CCU1 compare match occurred

Please note, that in case of the 'serve last' compare strategy, when the SLA-bit in the ATOMi_CHx_CTRL register is not set, the ACBO(4) bit is always set and the ACBO(3) bit is always reset after the compare match event occurred.

The ACBO bit field is reset, when the DV bit is set.

Depending on the capture compare unit where the time base matched the interrupt CCU0TCx_IRQ or CCU1TCx_IRQ is raised.

When CCU0 and CCU1 is used for comparison it is possible to generate very small spikes on the output pin by loading CM0 and CM1 with two time stamp values for TBU_TS0, TBU_TS1 or TBU_TS2 close together. The output pin will then be set or reset dependent on the SL bit and the specified ACBI(0) and ACBI(1) bits in the ACBI bit field of the ATOMi_CHx_STAT register on the first match event and the output will toggle on the second match event.

It is important to note, that the bigger (smaller) time stamp has to be loaded into the CM1 register, since the CCU0 will enable the CCU1 once it has reached its comparison time stamp. The order of the comparison time stamps depends on the defined greater/equal or less/equal comparison of the CCUx units.

For compare strategy 'serve last' the CCU0 and CCU1 compare match may occur sequentially. During different phases of compare match the CPU access rights to registers CM0 and CM1 and the WR REQ are depicted in the following figure.

Generic Timer Module (GTM)

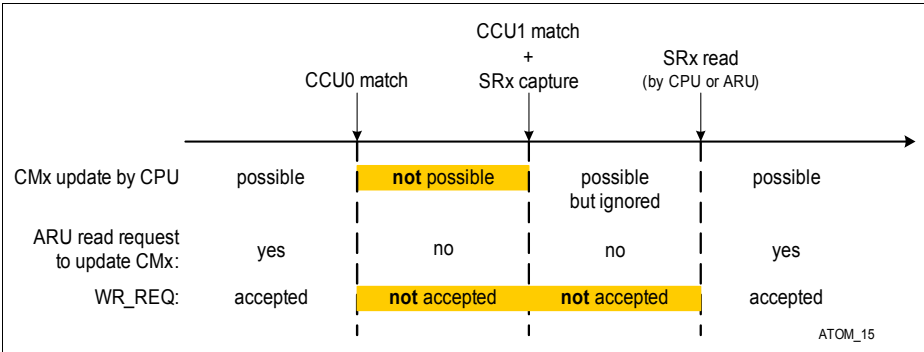


Figure 26-47 CPU access rights in case of compare strategy 'serve last'

ARU Non-Blocking mode

When the compare registers are updated via ARU the update behaviour of the channel is configurable with the ABM bit inside the ATOMi_CHx_CTRL register. When the ABM bit is reset, the ATOM channel is in ARU non-blocking mode.

In this ARU non-blocking mode, data received via ARU is continuously transferred to the registers CM0 and CM1 and the bit field ACBI of register ATOMi_CHx_STAT as long as no specified compare match event occurs.

After a compare match event that causes an update of the shadow register SR0/SR1 and before reading the SR0/SR1 register via CPU or ARU, the update of the registers CM0/ CM1 via CPU or ARU is possible but has no effect.

To set up a new compare action, first the SR0/SR1 registers containing captured values have to be read and then new compare values have to be written into the register CM0/CM1. This can be done either by ARU or by CPU.

When the CPU does the register accesses, only one of the shadow registers has to be read. Dependent on the compare strategy, the CPU has to write one or both of the compare registers.

In SOMC mode and CCUx control mode 'serve last' exist an exception for update of register CM0/CM1. If in this mode the CCU0 compare match event occurred, the update of register CM0/CM1 via CPU or ARU is blocked until the CCU1 compare match event occurs.

The CPU can check at any time if the ATOM channel has received valid data from the ARU and waits for a compare event to happen. This is signalled by the DV bit inside the ATOMi_CHx_STAT register.

The behaviour of an ATOM channel in SOMC mode, when ARU is enabled and ARU blocking mode is disabled is shown in [Figure 26-48](#).

SOMC State diagram for SOMC mode, ARU enabled, ABM disabled

Generic Timer Module (GTM)

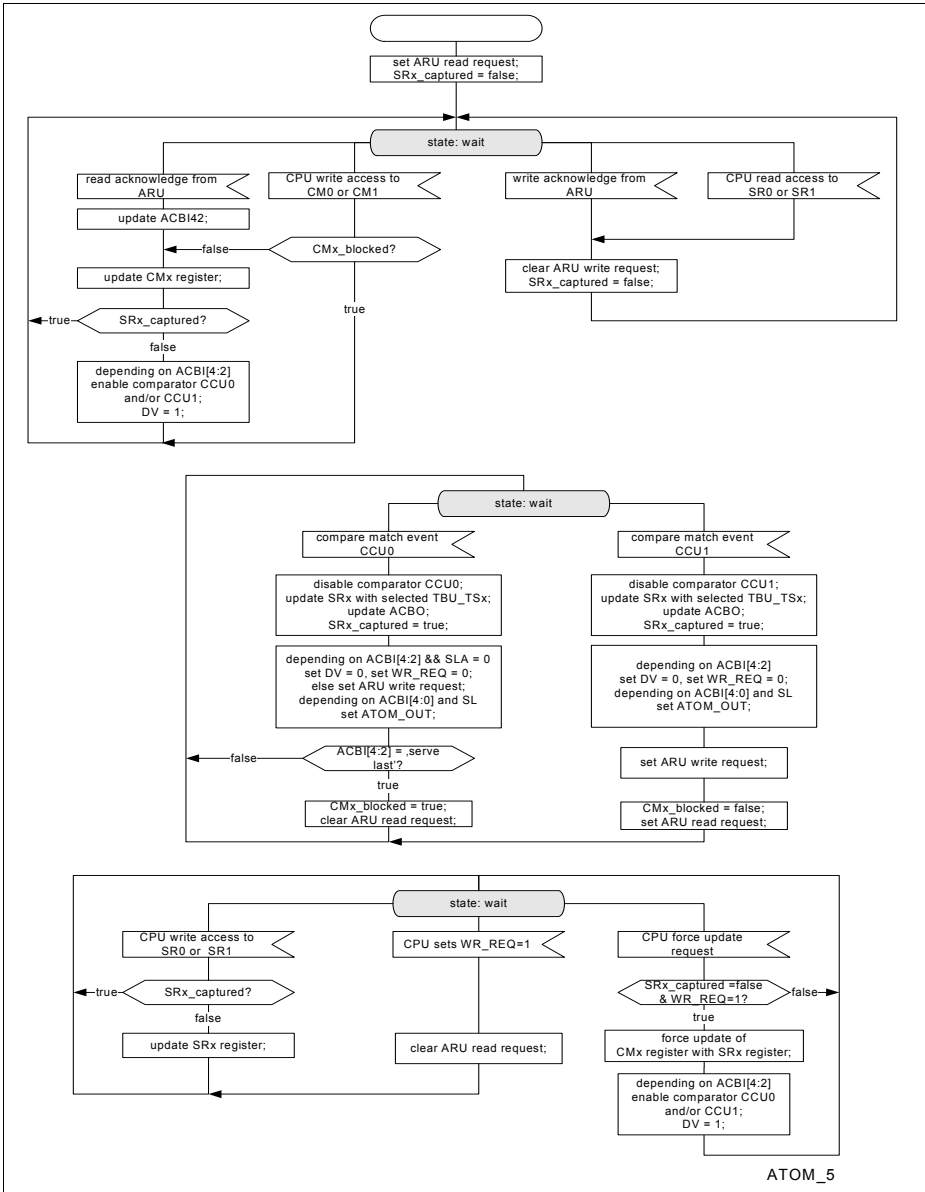


Figure 26-48 SOMC State diagram for SOMC mode, ARU enabled, ABM disabled

Generic Timer Module (GTM)**ARU Blocking mode**

When the compare registers are updated by ARU, the ATOM channel can be configured to receive ARU data in a blocking manner. This can be configured by setting the ABM bit in the ATOMi_CHx_CTRL register.

If the ABM and ARU_EN bits are set, the (one) two compare values for CM0 and/or CM1 can be provided by ARU or CPU. If the compare registers CM0 and/or CM1 are/is updated, the ATOM channel waits for the compare match event to happen. No further data is requested from the ARU.

When the specified compare match event happens, the shadow registers SR0 and SR1 are updated together with the ACBO bits in the ATOMi_CHx_STAT register. The data in the shadow registers is marked as valid for the ARU and the DV bit is reset inside the ATOMi_CHx_CTRL register.

If the register SR0 and SR1 holding the captured TBU time stamp values are read by either the ARU or the CPU, the next write access to or update of the register CM0 or CM1 via ARU or the CPU enables the new compare match check again.

At least one of the registers SR0 or SR1 has to be read, before new data is requested from ARU.

The CPU can check at any time if the ATOM channel has received valid data from the ARU and waits for a compare event to happen. This is signalled by a set DV bit inside the ATOMi_CHx_STAT register.

The behaviour of an ATOM channel in SOMC mode, when ARU is enabled and ARU blocking mode is enabled is shown in [Figure 26-49](#).

SOMC State diagram for SOMC mode, ARU enabled and ABM enabled

Generic Timer Module (GTM)

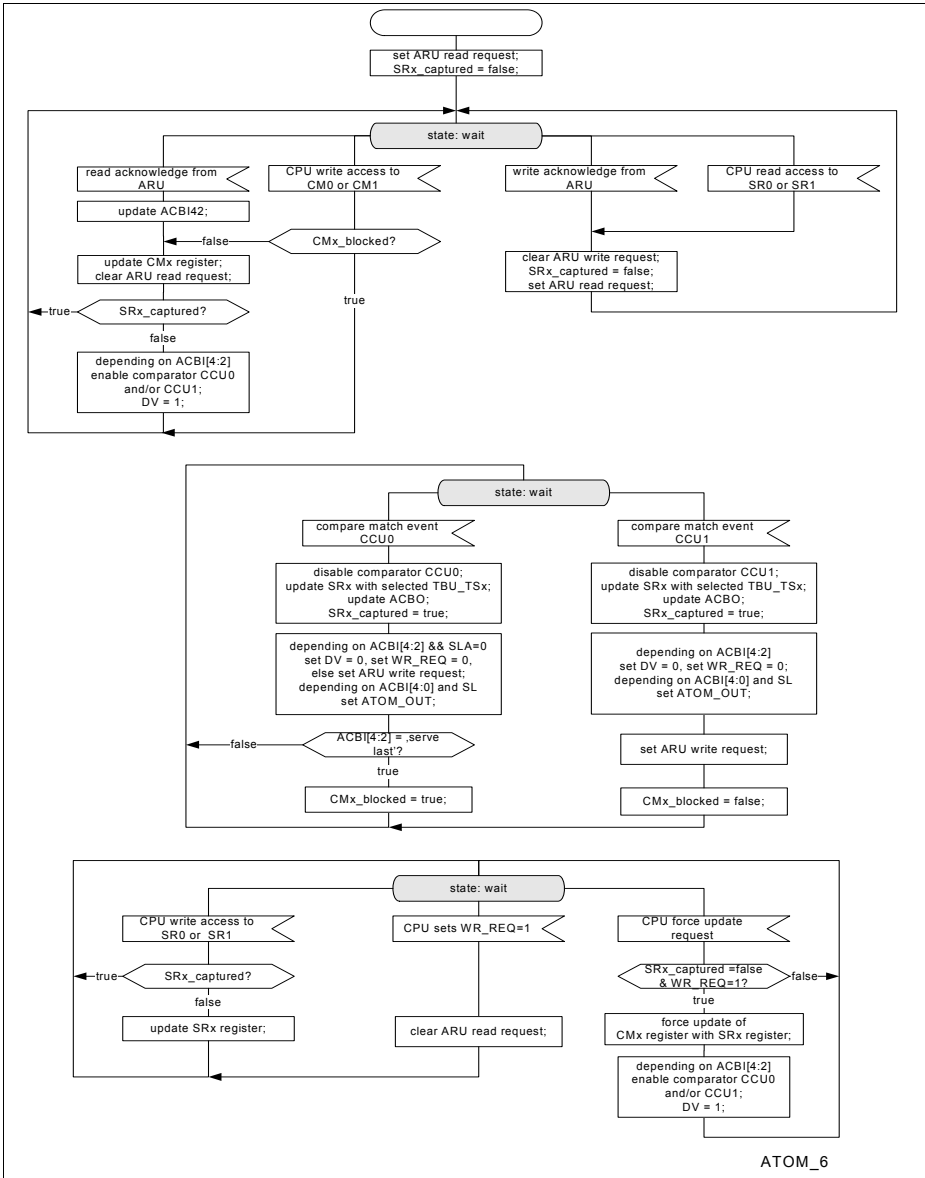


Figure 26-49 SOMC State diagram for SOMC mode, ARU enabled and ABM enabled

Generic Timer Module (GTM)**ATOM SOMC Late update mechanism**

Although, the ATOM channel may be controlled by data received via the ARU, the CPU is able to request at any time a late update of the compare register. This can be initiated by setting the WR_REQ bit inside the ATOMi_CHx_CTRL register. By doing this, the ATOM will request no further data from ARU (if ARU access was enabled). The channel will in any case continue to compare against the values stored inside the compare registers (if bit DV was set). The CPU can now update the new compare values until the compare event happens by writing to the shadow registers, and force the ATOM channel to update the compare registers by writing to the force update register bits in the AGC register.

If the WR_REQ bit is set and a compare match event happens, any further access to the shadow registers SR0, SR1 is blocked and the force update of this channel is blocked. In addition, the WRF bit is set in the ATOMi_CHx_STAT register. Thus, the CPU can determine that the late update failed by reading the WRF bit.

If a compare match event already happened, the WR_REQ bit could not be set until the channel is unlocked for a new compare match event by reading the shadow registers. In addition, the WRF bit is set if the CPU tries to write the WR_REQ bit in that case.

If between a correct WR_REQ bit set, a correct shadow register write, and before the force update is requested by the AGC a match event occurs on the old compare values, the WRF bit will be set.

The WRF bit will be set in any case if the CPU tries to write to a blocked shadow register.

The WR_REQ bit and the DV bit will be reset on a compare match event.

A blocked force update mechanism will be enabled again after a read access to the register SR0 or SR1 by either the ARU or the CPU.

The ATOM SOMC late update mechanism from CPU is shown in [Figure 26-50](#).

SOMC State diagram for late update requests by CPU

Generic Timer Module (GTM)

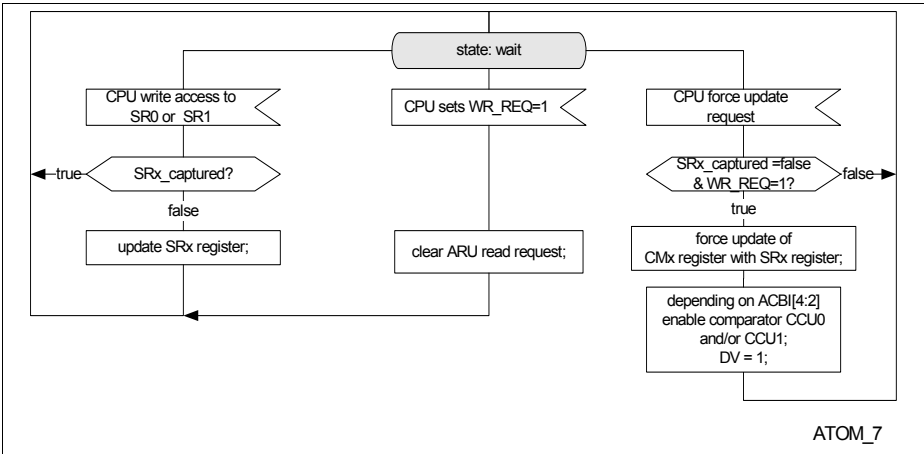


Figure 26-50 SOMC State diagram for late update requests by CPU

Generic Timer Module (GTM)

Register ATOMi_CHx_CTRL in SOMC mode (x: 0...7)

GTM_ATOM0_CHx_CTRL_SOMC (x=0-7)

 ATOM0 Channel x Control in SOMC mode Register
 (0D004_H+x*0080_H)

 Reset Value: 00000800_H

GTM_ATOM1_CHx_CTRL_SOMC (x=0-7)

 ATOM1 Channel x Control in SOMC mode Register
 (0D804_H+x*0080_H)

 Reset Value: 00000800_H

GTM_ATOM2_CHx_CTRL_SOMC (x=0-7)

 ATOM2 Channel x Control in SOMC mode Register
 (0E004_H+x*0080_H)

 Reset Value: 00000800_H

GTM_ATOM3_CHx_CTRL_SOMC (x=0-7)

 ATOM3 Channel x Control in SOMC mode Register
 (0E804_H+x*0080_H)

 Reset Value: 00000800_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				ABM	NotUsed	SLA	TRIG OUT	NotUsed			NotUsed	Reserved			WR_ REQ
r				rw	rw	rw	rw	rw			rw	r			rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rese rved	NotUsed			SL	Rese rved	CMP _CT RL	ACB42		ACB10		ARU _EN	TB12 _SE L	MODE		
r	rw			rw	r	rw	rw		rw		rw	rw	rw		

Field	Bits	Type	Description
MODE	[1:0]	rw	ATOM channel mode select 01 _B ATOM Signal Output Mode Compare (SOMC)
TB12_SEL	2	rw	Select time base value TBU_TS1 or TBU_TS2. 0 _B TBU_TS1 selected for comparison 1 _B TBU_TS2 selected for comparison Note: This bit is only applicable if three time bases are present in the GTM. Otherwise, this bit is reserved.
ARU_EN	3	rw	ARU Input stream enable 0 _B ARU Input stream disabled 1 _B ARU Input stream enabled

Generic Timer Module (GTM)

Field	Bits	Type	Description
ACB10	[5:4]	rw	<p>Signal level control bits</p> <p>00_B No signal level change at output (exception in Figure 26-45 mode ACB42=001).</p> <p>01_B Set output signal level to 1 when SL bit = 0 else output signal level to 0.</p> <p>10_B Set output signal level to 0 when SL bit = 0 else output signal level to 1.</p> <p>11_B Toggle output signal level (exception in Figure 26-45 mode ACB42=001).</p> <p>Note: These bits are only applicable if ARU_EN = '0'.</p>
ACB42	[8:6]	rw	<p>ATOM control bits ACB(4), ACB(3), ACB(2)</p> <p>000_B Compare in CCU0 and CCU1 in parallel, disable the CCUx on a compare match on either of compare units. Use TBU_TS0 in CCU0 and TBU_TS1 or TBU_TS2 in CCU1.</p> <p>001_B Compare in CCU0 and CCU1 in parallel, disable the CCUx on a compare match on either compare units. Use TBU_TS0 in CCU0 and TBU_TS1 or TBU_TS2 in CCU1.</p> <p>010_B Compare in CCU0 only against TBU_TS0.</p> <p>011_B Compare in CCU1 only against TBU_TS1 or TBU_TS2.</p> <p>100_B Compare first in CCU0 and then in CCU1. Use TBU_TS0.</p> <p>101_B Compare first in CCU0 and then in CCU1. Use TBU_TS1 or TBU_TS2.</p> <p>110_B Compare first in CCU0 and then in CCU1. Use TBU_TS0 in CCU0 and TBU_TS1 or TBU_TS2 in CCU1.</p> <p>111_B Reserved.</p> <p>Note: These bits are only applicable if ARU_EN = '0'.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
CMP_CTRL L	9	rw	CCUx compare strategy select 0 _B Greater/equal compare against TBU time base values (TBU_TSx >= CMx) 1 _B Less/equal compare against TBU time base values (TBU_TSx <= CMx) Note: The compare unit CCU0 or CCU1 that compares against TBU_TS0 (depending on CCUx control mode defined by ACBI(4:2) or ACB42) always perform a greater/equal comparison, independent on CMP_CTRL bit.
Reserved	10	r	Reserved Read as zero, should be written as zero.
SL	11	rw	Initial signal level after channel enable 0 _B Low signal level 1 _B High signal level Note: If the output is disabled, the output ATOM_OUT[x] is set to inverse value of SL. Note: If the channel and output are disabled, in MODE=01 (SOMC mode) the output register of SOU unit is set to value of SL. If the output is enabled afterwards, the output ATOM_OUT[x] is equal to the value of SL.
NotUsed	[14:12]	rw	Not used in this mode
Reserved	15	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

Field	Bits	Type	Description
WR_REQ	16	rw	CPU write request bit 0 _B No late update requested by CPU 1 _B Late update requested by CPU Note: The CPU can disable subsequent ARU read requests by the channel and can update the shadow registers with new compare values, while the compare units operate on old compare values received by former ARU accesses, if occurred. Note: On a compare match event, the WR_REQ bit will be reset by hardware. Note: At the point of the force update only the shadow registers SR0 and SR1 are transferred into the CM0, CM1 registers. The output action is still defined by the ACBI bit field described by the ARU together with the old compare values for CM0/CM1.
Reserved	[19:17]	r	Reserved Read as zero, should be written as zero.
NotUsed	20	rw	Not used in this mode
NotUsed	[23:21]	r	Not used in this mode
TRIGOUT	24	rw	Trigger output selection (output signal TRIG_CHx) of module ATOM_CHx 0 _B TRIG_x[is TRIG_x-1] 1 _B TRIG_x[is TRIG_CCU0

Generic Timer Module (GTM)

Field	Bits	Type	Description
SLA	25	rw	<p>Serve last ARU communication strategy</p> <p>0_B Capture SRx time stamps after CCU0 match event not provided to ARU</p> <p>1_B Capture SRx time stamps after CCU0 match event provided to ARU</p> <p>Note: Please note, that setting of this bit has only effect, when ACBI(4:2) is configured for serve last compare strategy ("100", "101", or "110").</p> <p>Note: When this bit is not set, the captured time stamps in the shadow registers SRx are only provided after the CCU1 match occurred. The ACBO(4:3) bits always return "10" in that case.</p> <p>Note: By setting this bit, the ATOM channel also provides the captured time stamps after the CCU0 match event to the ARU. The ACBO(4:3) bits are set to "01" in that case. After the CCU1 match event, the time stamps are captured again in the SRx registers and provided to the ARU. The ACBO(4:3) bits are set to "10". When the data in the shadow registers after the CCU0 match was not consumed by an ARU destination and the CCU1 match occurs, the data in the shadow registers is overwritten by the new captured time stamps. The ATOM channel does not request new data from the ARU when the CCU0 match values are read from an ARU destination.</p>
NotUsed	26	rw	Not used in this mode
ABM	27	rw	<p>ARU blocking mode</p> <p>0_B ARU blocking mode disabled: ATOM reads continuously from ARU and updates CM0, CM1 independent of pending compare match event</p> <p>1_B ARU blocking mode enabled: after updating CM0, CM1 via ARU, no new data is read from ARU until compare match event occurred and SR0 and/or SR1 are read.</p>
Reserved	[31:28]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

26.12.3.3 ATOM Signal Output Mode PWM (SOMP)

In ATOM Signal Output Mode PWM (SOMP) the ATOM submodule channel is able to generate complex PWM signals with different duty cycles and periods. Duty cycles and periods can be changed synchronously and asynchronously. Synchronous change of the duty cycle and/or period means that the duty cycle or period duration changes after the end of the preceding period. An asynchronous change of period and/or duty cycle means that the duration changes during the actual running PWM period.

The signal level of the pulse generated inside the period can be configured inside the channel control register (SL bit of ATOMi_CHx_CTRL register). The initial signal output level for the channel is the erase pulse level defined by the SL bit. [Figure 26-51](#) clarifies this behaviour.

In SOMP mode, depending on configuration bits RST_CCU0 of register ATOMi_CHx_CTRL the counter register CN0 can be reset either when the counter value is equal to the compare value CM0 or when signaled by the ATOMi trigger signal TRIG [x-1] of the preceding channel.

In this case, if UPEN_CTRLx=1, also the working register CM0, CM1 and CLK_SRC are updated.

PWM Output behaviour with respect to the SL bit in the ATOMi_CHx_CTRL register

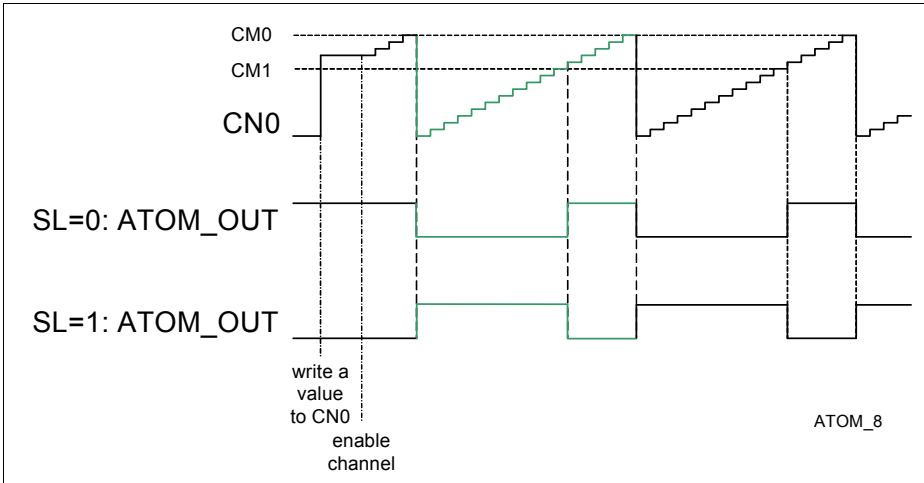


Figure 26-51 PWM Output behaviour with respect to the SL bit in the ATOMi_CHx_CTRL register

On an asynchronous update, it is guaranteed, that no spike occurs at the output port of the channel due to a too late update of the operation registers. The behaviour of the output signal due to the different possibilities of an asynchronous update during a PWM period is shown in [Figure 26-52](#).

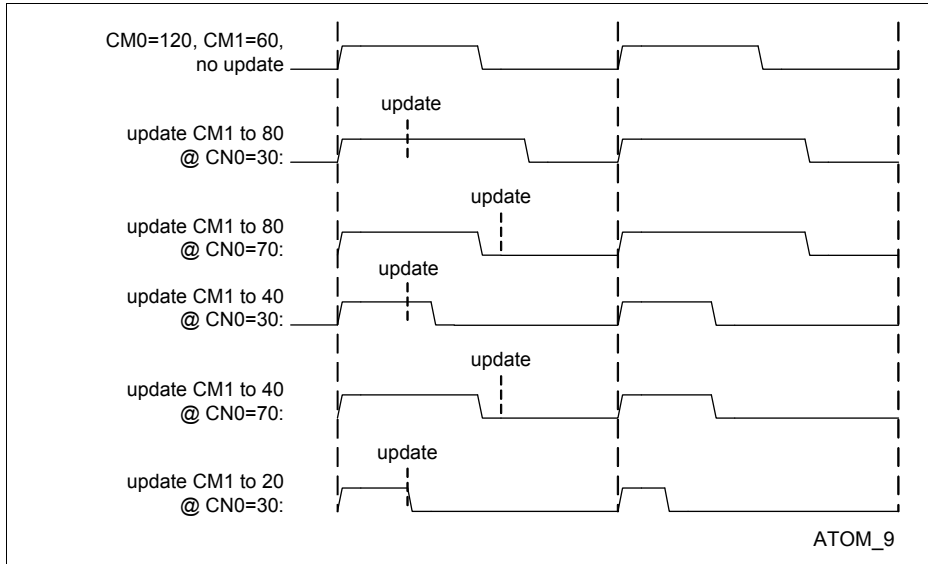
PWM Output behaviour in case of an asynchronous update of the duty cycle


Figure 26-52 PWM Output behaviour in case of an asynchronous update of the duty cycle

The duration of the pulse high or low time and period is measured with the counter in subunit CCU0. The trigger of the counter is one of the eight CMU clock signals configurable in the channel control register `ATOMi_CHx_CTRL`. The register `CM0` holds the duration of the period and the register `CM1` holds the duration of the duty cycle in clock ticks of the selected CMU clock.

If counter register `CN0` of channel `x` is reset by its own CCU0 unit (i.e. the compare match of $CN0 \geq CM0$ configured by `RST CCU0=0`), following statements are valid:

- if $CM0=0$ or $CM0=1$, 0% duty cycle ($=\overline{SL}$) is generated independent of `CM1`. The counter `CN0` is not counting.
- the configuration of `CM1=0` represents 0% duty cycle ($=\overline{SL}$) at the output
- the configuration of $CM1 \geq CM0$ represents 100% duty cycle ($=SL$)

If counter register `CN0` of `x` channel is reset by the trigger signal coming from another channel or the assigned TIM module (configured by `RST CCU0=1`), following statements are valid:

- `CM0` defines the edge to `SL` value, `CM1` defines the edge to \overline{SL} value
- if $CM0=CM1$, the output is 100% `SL` (`CM0` has higher priority)

Generic Timer Module (GTM)

- if CM0=0, the output stays at its last value (CN0 stops counting)

In case the counter value CN0 reaches the compare value in register CM0 or the channel receives an external update trigger via the FUPD(x) signal, a synchronous update is performed. A synchronous update means that the registers CM0 and CM1 are updated with the content of the shadow registers SR0 and SR1 and the CLK_SRC register is updated with the value of the CLK_SRC_SR register.

The clock source for the counter can be changed synchronously at the end of a period. If ARU access is disabled, this is done by using the bit field CLK_SRC_SR of register ATOMi_CHx_CTRL as shadow registers for the next CMU clock source.

If ARU access is enabled, the bits ACBI(4), ACBI(3) and ACBI(2) received via ARU and stored in register ATOM_i_CHx_STAT are used as shadow register for the update of the CMU clock source register CLK_SRC.

For the synchronous update mechanism the generation of a complex PWM output waveform is possible without CPU interaction by reloading the shadow registers SR0, SR1 and the ACBI bit field over the ACI subunit from the ARU, while the ATOM channel operates on the CM0 and CM1 registers.

This internal update mechanism is established, when the old PWM period ends. The shadow registers are loaded into the operation registers, the counter register is reset, the new clock source according to the CLK_SRC_SR or ACBI(4), ACBI(3) and ACBI(2) bits is selected and the new PWM generation starts.

In parallel, the ATOM channel issues a read request to the ARU to reload the shadow registers with new values while the ATOM channel operates on the operation registers. To guarantee the reloading, the PWM period must not be smaller than the worst case ARU round trip time and source for the PWM characteristic must provide the new data within this time. Otherwise, the old PWM values are used from the shadow registers.

When updated over the ARU the user has to ensure that the new period duration is located in the lower (bits 23 to 0) and the duty cycle duration is located in the upper (bits 47 to 24) ARU data word and the new clock source is specified in the ARU control bits 52 to 50.

This pipelined data stream character is shown in [Figure 26-53](#).

ARU Data input stream pipeline structure for SOMP mode

Generic Timer Module (GTM)

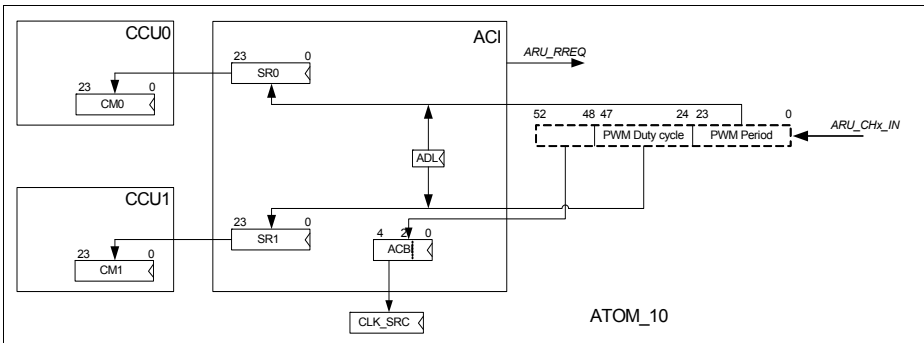


Figure 26-53 ARU Data input stream pipeline structure for SOMP mode

When an ARU transfer is in progress which means the ARU_RREQ is served by the ARU, the ACI locks the update mechanism of CM0, CM1 and CLK_SRC until the read request has finished. The CCU0 and CCU1 operate on the old values when the update mechanism is locked.

The shadow registers SR0 and SR1 can also be updated over the AEI bus interface. When updated via the AEI bus the CM0 and CM1 update mechanism has to be locked via the AGC_GLB_CTRL register with the UPENx signal in the AGC subunit. To select the new clock source in this case, the CPU has to write to the CLK_SRC_SR bit field of the ATOMi_CHx_CTRL register.

For an asynchronous update of the duty cycle and/or period the new values must be written directly into the compare registers CM0 and/or CM1 while the counter CN0 continues counting. This update can be done only via the AEI bus interface immediately by the CPU or by the FUPD(x) trigger signal triggered from the AGC global trigger logic. Values received through the ARU interface are never loaded asynchronously into the operation registers CM0 and CM1. Therefore, the ATOM channel can generate a PWM signal on the output port pin ATOM[i]_CH[x]_OUT on behalf of the content of the CM0 and CM1 registers, while it receives new PWM values via the ARU interface ACI in its shadow registers.

On a compare match of CN0 and CM0 or CM1 the output signal level of ATOM[i]_CH[x]_OUT is toggled according to the signal level output bit SL in the ATOMi_CHx_CTRL register.

Thus, the duty cycle output level can be changed during runtime by writing the new duty cycle level into the SL bit of the channel configuration register. The new signal level becomes active for the next trigger CCU_TRIGx (since bit SL is written).

Since the ATOM[i]_CH[x]_OUT signal level is defined as the erse duty cycle output level when the ATOM channel is enabled, a PWM period can be shifted earlier by writing an initial offset value to CN0 register. By doing this, the ATOM channel first counts until CN0 reaches CM0 and then it toggles the output signal at ATOM[i]_CH[x]_OUT.

SOMP One-shot mode

The ATOM channel can operate in One-shot mode when the OSM bit is set in the channel control register. One-shot mode means that a single pulse with the pulse level defined in bit SL is generated on the output line.

First the channel has to be enabled by setting the corresponding `ENDIS_STAT` value.

In One-shot mode the counter `CN0` will not be increment once the channel is enabled.

A write access to the register `CN0` triggers the start of pulse generation (i.e. the increment of the counter register `CN0`).

If the counter `CN0` is reset from `CM0` back to zero, the first edge at `ATOM[i]_CH[x]_OUT` is generated.

To avoid an update of `CMx` register with content of `SRx` register at this point in time, the automatic update should be disabled by setting `UPEN_CTRLx = 00` (in register `ATOMi_CHx_CTRL`).

The second edge is generated if `CN0` is greater or equal than `CM1` (i.e. `CN0` was increment until it has reached `CM1` or `CN0` is greater than `CM1` after an update of `CM1`).

If the counter `CN0` has reached the value of `CM0` a second time, the counter stops.

The new value of `CN0` determines the start delay of the first edge. The delay time of the first edge is given by $(CM0 - CN0)$ multiplied with period defined by current value of `CLK_SRC`.

Figure 26-54 clarifies the pulse generation in SOMP One-shot mode.

PWM Output with respect to configuration bit `SL` in One-shot mode: trigger by writing to `CN0`

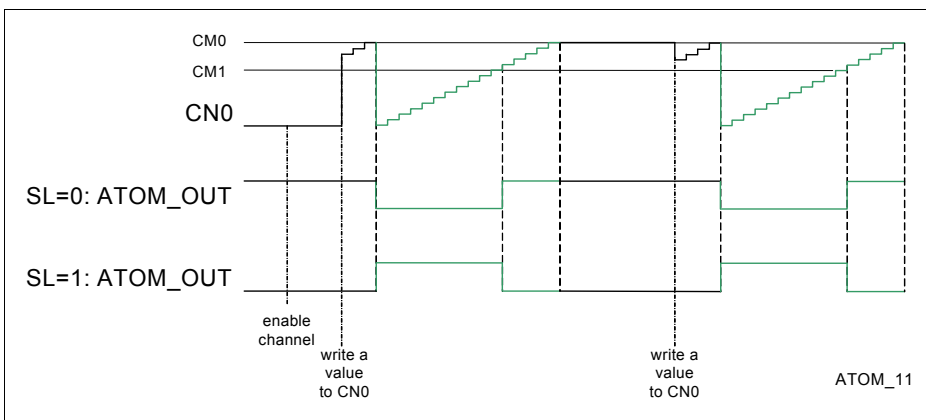


Figure 26-54 PWM Output with respect to configuration bit `SL` in One-shot mode:

Generic Timer Module (GTM)**trigger by writing to CN0**

Further output of single pulses can be started by a write access to register CN0.

If CN0 is already incrementing (i.e. started by writing to CN0 a value $CN0_{start} < CM0$), the affect of a second write access to CN0 depends on the phase of CN0:

phase 1: update of CN0 before CN0 reaches first time CM0

phase 2: update of CN0 after CN0 has reached first time CM0 but less than CM1

phase 3: update of CN0 after CN0 has reached first time CM0 and CN0 is greater than or equal CM1

In phase 1: writing to counter CN0 a value $CN0_{new} < CM0$ leads to shift of first edge (generated if CN0 reaches CM0 first time) by the time $CM0 - CN0_{new}$.

In phase 2: writing to incrementing counter CN0 a value $CN0_{new} < CM1$ while $CM0_{old}$ is below CM1 leads to a lengthening of the pulse. The counter CN0 stops if it reaches CM0.

In phase 3: writing to incrementing counter CN0 a value $CN0_{new}$ while $CN0_{old}$ is already greater than or equal CM1 leads to an immediate restart of a single pulse generation inclusive the initial defined by $CM0 - CN0_{new}$.

Generic Timer Module (GTM)

Register ATOMi_CHx_CTRL in SOMP mode (x: 0...7)

GTM_ATOM0_CHx_CTRL_SOMP (x=0-7)

ATOM0 Channel x Control in SOMP mode Register

 $(0D004_H + x * 0080_H)$

 Reset Value: 00000800_H

GTM_ATOM1_CHx_CTRL_SOMP (x=0-7)

ATOM1 Channel x Control in SOMP mode Register

 $(0D804_H + x * 0080_H)$

 Reset Value: 00000800_H

GTM_ATOM2_CHx_CTRL_SOMP (x=0-7)

ATOM2 Channel x Control in SOMP mode Register

 $(0E004_H + x * 0080_H)$

 Reset Value: 00000800_H

GTM_ATOM3_CHx_CTRL_SOMP (x=0-7)

ATOM3 Channel x Control in SOMP mode Register

 $(0E804_H + x * 0080_H)$

 Reset Value: 00000800_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			NotUsed	OSM	Reserved	TRIG OUT	NotUsed			RST_CC U0	Reserved			NotUsed	
r			rw	rw	r	rw	rw			rw	r			rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	CLK_SRC_SR		SL	Reserved		NotUsed			ADL	ARU_EN	NotUsed	MODE			
r	rw		rw	r		rw			r	rw	rw	rw			

Field	Bits	Type	Description
MODE	[1:0]	rw	ATOM channel mode select 10 _B ATOM Signal Output Mode PWM (SOMP)
NotUsed	2	rw	Not used in this mode
ARU_EN	3	rw	ARU Input stream enable 0 _B ARU Input stream disabled 1 _B ARU Input stream enabled

Generic Timer Module (GTM)

Field	Bits	Type	Description
ADL	[5:4]	r	ARU data select for SOMP 00 _B Load both ARU words into shadow registers 01 _B Load ARU low word (Bits 23...0) into shadow register SR0 10 _B Load ARU high word (Bits 47...24) into shadow register SR1 11 _B Reserved Note: This bit field is only used in SOMP mode to select the ARU data source.
NotUsed	[8:6]	r	Not used in this mode
Reserved	[10:9]	r	Reserved Read as zero, should be written as zero.
SL	11	rw	Signal level for pulse of PWM 0 _B Low signal level 1 _B High signal level If the output is disabled, the output ATOM_OUT[x] is set to inverse value of SL.
CLK_SRC_SR	[14:12]	rw	Shadow register for CMU clock source register CLK_SRC 000 _B CMU_CLK0 selected 001 _B CMU_CLK1 selected 010 _B CMU_CLK2 selected 011 _B CMU_CLK3 selected 100 _B CMU_CLK4 selected 101 _B CMU_CLK5 selected 110 _B CMU_CLK6 selected 111 _B CMU_CLK7 selected Note: This register is a shadow register for the CMU_CLKx select. Thus, if the CMU_CLK source for PWM generation should be changed during operation, the old CMU_CLK has to operate until the update of the ATOM channels internal CLK_SRC register by the CLK_SRC_SR content is done either by an end of a period or a FORCE_UPDATE. Note: After (channel) reset the selected CLK_SRC value is the SYS_CLK (input of Global Clock Divider). To use one of the CMU_CLKx, it is required to perform a forced update of CLK_SRC with the value of CLK_SRC_SR value before/with enabling the channel. Note: These bits are only applicable if ARU_EN=0.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	15	r	Reserved Read as zero, should be written as zero.
NotUsed	16	rw	Not used in this mode
Reserved	[19:17]	r	Reserved Read as zero, should be written as zero.
RST_CCU0	20	rw	Reset source of CCU0 0_B Reset counter register CN0 to 0 on matching comparison with CM0 1_B Reset counter register CN0 to 0 on trigger TRIG _[x-1] <i>Note: If RST_CCU0=1 and UPEN_STRLx=1 are set, TRIG_[x-1] triggers also the update of working register (CM0, CM1 and CLK_SRC).</i>
NotUsed	[23:21]	r	Not used in this mode
TRIGOUT	24	rw	Trigger output selection (output signal TRIG_CHx) of module ATOM_CHx. 0_B TRIG _[x] is TRIG _[x-1] 1_B TRIG _[x] is TRIG_CCU0
Reserved	25	r	Reserved Read as zero, should be written as zero.
OSM	26	rw	One-shot mode 0_B Continuous PWM generation after channel enable 1_B A single pulse is generated
NotUsed	27	rw	Not used in this mode
Reserved	[31:28]	r	Reserved Read as zero, should be written as zero.

26.12.3.4 ATOM Signal Output Mode Serial (SOMS)

In ATOM Signal Output Mode Serial (SOMS) the ATOM channel acts as a serial output shift register where the content of the CM1 register in the CCU1 unit is shifted out whenever the unit is triggered by the selected CMU_CLK input clock signal. The shift direction is configurable with the ACB(0) bit inside the ATOM_i_CH_x_CTRL register when ARU is disabled and the ACBI(0) bit inside the ATOM_i_CH_x_STAT register when ARU is enabled.

The data inside the CM1 register has to be aligned according to the selected shift direction in the ACB(0)/ACBI(0) bit. This means that when a right shift is selected, that

Generic Timer Module (GTM)

the data word has to be aligned to bit 0 of the CM1 register and when a left shift is selected, that the data has to be aligned to bit 23 of the CM1 register.

SOMS Mode output generation

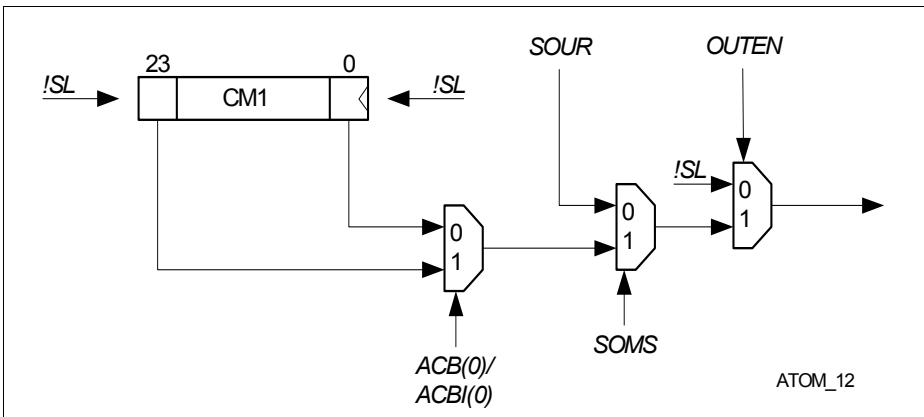


Figure 26-55 SOMS Mode output generation

Figure 12.3.4.1 shows the output generation in case of SOMS mode is selected.

In SOMS mode CCU0 runs in counter/compare mode and counts the number of bits shifted out so far. The total number of bits that should be shifted is defined as CM0. The total number of bits that are visible at ATOM_OUT is CM0+1.

When the output is disabled the ATOM_OUT is set to the inverse SL bit definition.

When the content of the CM1 register is shifted out, the inverse signal level is shifted into the CM1 register.

When the output is enabled while UPEN_CTRL[x] is disabled, the ATOM_OUT signal level is defined by CM1 bit 0 or 23, dependent on the shift direction defined by ACB(0) or ACBI(0) register setting. Figure 12.3.4.2 should clarify the ATOM channel startup behaviour in this case for right shift. For left shift the CM1 bit 0 in 12.3.4.2 has to be replaced by CM1 bit 23.

SOMS Output signal level at startup, UPEN_CTRL[x] disabled

Generic Timer Module (GTM)

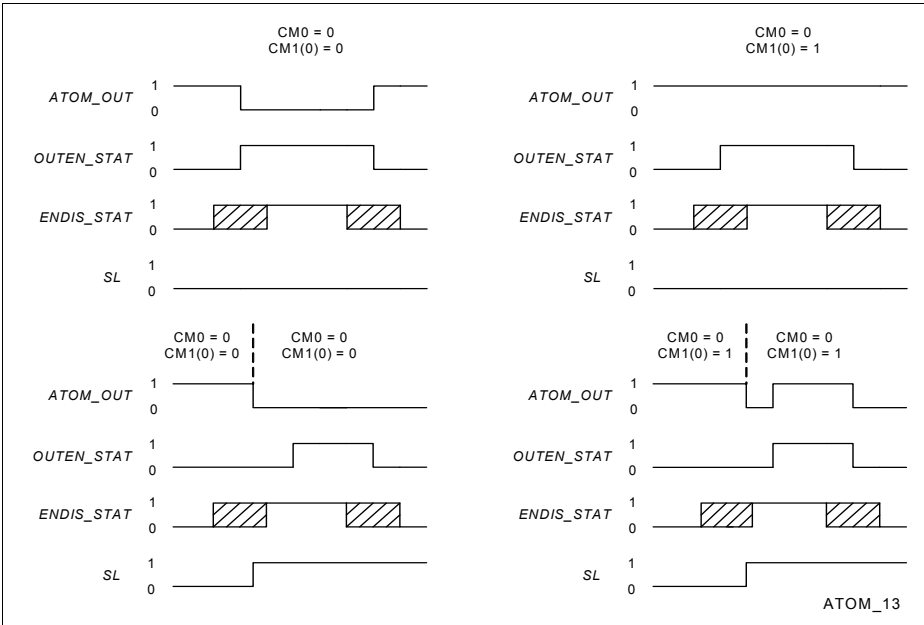


Figure 26-56 SOMS Output signal level at startup, `UPEN_CTRL[x]` disabled

If `UPEN_CTRL[x]` is set and the channel is enabled, the output level is defined by bit 0 or 23 of `CM1` register dependent on the shift direction. Figure 12.3.4.3 shows the output behaviour in that case.

SOMS Output signal level at startup, `UPEN_CTRL[x]` enabled

Generic Timer Module (GTM)

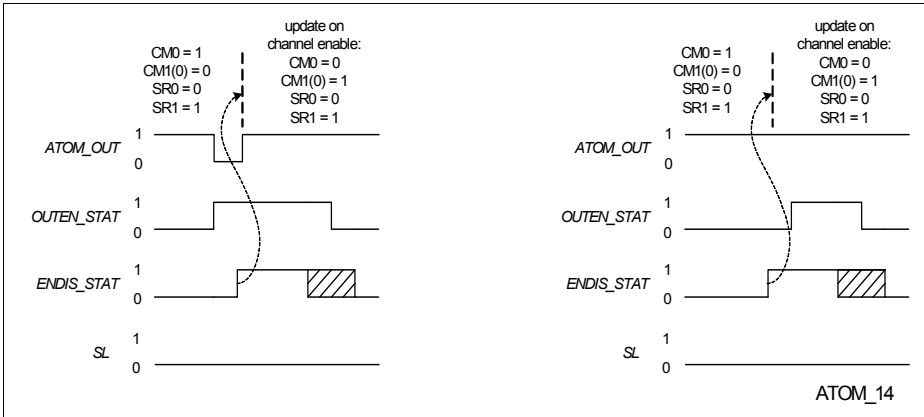


Figure 26-57 SOMS Output signal level at startup, UPEN_CTRL[x] enabled

When the serial data to be shifted is provided via ARU the number of bits that should be shifted has to be defined in the lower 24 bits of the ARU word (23 to 0) and the data that is to be shifted has to be defined in the ARU bits 47 to 24 aligned according to the shift direction. This shift direction has to be defined in the ARU word bit 48 (SL0 bit).

If bit UPEN_CTRL[x] of a channel x is set, after update of CM0/CM1 register with the content of the SR0/SR1 register, a new ARU read request is set up.

If bit UPEN_CTRL[x] of a channel x is not set, no (further) ARU read request is set up (because the SR0/SR1 register are never used for update) and the ATOM may stop shifting after CN0 has reached CM0. Note, that in this case also no automatic restart of shifting is possible.

If a channel is enabled with the settings SOMS mode and ARU_EN = 1, the first received values from ARU are stored in register SR0 and SR1. If CN0 and CM0 are 0 (i.e. CN0 is not counting) and the update of channel x is enabled (UPEN_CTRL[x]=1), an immediate update of the register CM0 and CM1 is also done. This update of CM0 and CM1 triggers the start of shifting.

It is recommended to configure the ATOM channel in One-shot mode when the ARU_EN bit is not set, since the ATOM channel would reload new values from the shadow registers when CN0 reaches CM0.

SOMS mode with ARU_EN = 1 and OSM = 0, UPEN_CTRL[x] = 1:

In case of bit ARU_EN is set and bit OSM is not set, the channel is running in the SOMS continuous mode. Then, if the content of the CM0 register equals the counter CN0, the CM0 and CM1 registers are reloaded with the SR0 and SR1 content and new values are requested from the ARU. If the update of the shadow registers does not happen before

Generic Timer Module (GTM)

CN0 reaches CM0 the old values of SR0 and SR1 are used to reload the operation registers.

In contrast to controlling the channel via AEI, the shift direction defined by ARU word bit 48 has only effect after the update of CMx operation registers from the SRx registers.

SOMS mode with ARU_EN = 1 and OSM = 1, UPEN_CTRL[x] = 1:

In case of bit ARU_EN is set and bit OSM is set, the channel is running in the SOMS one-shot mode. Then, if the content of the CM0 register equals the counter CN0 and if new values are available in SR0 and SR1 (bit DV set), the CM0 and CM1 registers are reloaded with the SR0 and SR1 content and new values are requested from the ARU. If no new values are available in SR0 and SR1, the register CM0 and CM1 will not be updated, the counter CN0 stops and the ATOM channel continues to request new data from ARU. A later reception of new ARU data in SR0 and SR1 will immediately force the update of the register CM0 and CM1 and restart the counter CN0.

SOMS mode with ARU_EN = 0 and OSM = 0, UPEN_CTRL[x] = 1:

In case of bit ARU_EN is not set and bit OSM is not set, the ATOM channel updates its CM0/CM1 register with the content of the SR0/SR1 register and restarts shifting immediately. The first bit of new CM1 register value will be applied at the output without any gap to the last bit of the previous CM1 register value.

SOMS mode with ARU_EN = 0 and OSM = 1, UPEN_CTRL[x] = 1:

In case of bit ARU_EN is not set and bit OSM is set, the ATOM channel stops shifting when CN0 reaches CM0 and no update of CM0 and CM1 is performed.

Then, the shifting of the channel can be restarted again by writing a zero (0) to the CN0 register again. Please note, that the CN0 register should be written with a zero since the CN0 register counts the number of bits shifted out by the ATOM channel.

The writing of a zero to CN0 causes also an immediate update of CM0/CM1 register with the content of SR0/SR1 register.

Interrupts in SOMS mode

In ATOM Signal Output Mode Serial only the interrupt CCU0TC (ATOM[i]_CH[x]_IRQ_NOTIFY) in case of CN0 >= CM0 is generated. The interrupt CCU1TC has no meaning and is not generated.

Generic Timer Module (GTM)

Register ATOMi_CHx_CTRL in SOMS mode (x: 0...7)

GTM_ATOM0_CHx_CTRL_SOMS (x=0-7)

ATOM0 Channel x Control in SOMS mode Register

 $(0D004_H + x * 0080_H)$

 Reset Value: 00000800_H

GTM_ATOM1_CHx_CTRL_SOMS (x=0-7)

ATOM1 Channel x Control in SOMS mode Register

 $(0D804_H + x * 0080_H)$

 Reset Value: 00000800_H

GTM_ATOM2_CHx_CTRL_SOMS (x=0-7)

ATOM2 Channel x Control in SOMS mode Register

 $(0E004_H + x * 0080_H)$

 Reset Value: 00000800_H

GTM_ATOM3_CHx_CTRL_SOMS (x=0-7)

ATOM3 Channel x Control in SOMS mode Register

 $(0E804_H + x * 0080_H)$

 Reset Value: 00000800_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			NotUsed	OSM	Reserved	NotUsed	NotUsed			NotUsed	Reserved			NotUsed	
r			rw	rw	r	rw	rw			rw	r			rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	CLK_SRC		SL	Reserved		NotUsed			ACB0	ARU_EN	NotUsed	MODE			
r	rw		rw	r		rw			rw	rw	rw	rw			

Field	Bits	Type	Description
MODE	[1:0]	rw	ATOM channel mode select 11 _B ATOM Signal Output Mode Serial (SOMS)
NotUsed	2	rw	Not used in this mode
ARU_EN	3	rw	ARU Input stream enable 0 _B ARU Input stream disabled 1 _B ARU Input stream enabled

Generic Timer Module (GTM)

Field	Bits	Type	Description
ACB0	4	rw	Shift direction for CM1 register 0_B Right shift of data is started from bit 0 of CM1 1_B Left shift of data is started from bit 23 of CM1 Note: the data that has to be shifted out has to be aligned inside the CM1 register according to the defined shift direction. Note: this bit is only applicable if ARU_EN = '0'. Note: if the direction (ACB0) is changed the output ATOM_OUT[x] switches immediately to the other 'first' bit of CM1 (bit 0 if ACB0 = 0, bit 23 if ACB0 = 1).
NotUsed	[8:5]	rw	Not used in this mode
Reserved	[10:9]	r	Reserved Read as zero, should be written as zero.
SL	11	rw	Defines signal level when channel and output is disable 0_B High signal level 1_B Low signal level Note: If the channel is disabled or the output is disabled, the output ATOM_OUT[x] is set to inverse value of SL. Note: If the output is enabled, the output ATOM_OUT[x] is set to bit 0 or 23 of CM1 register. Note: The inverse value of SL is shifted into the CM1 register. Note: An enable or disable of the channel x has no effect on ATOM_OUT[x].

Generic Timer Module (GTM)

Field	Bits	Type	Description
CLK_SRC	[14:12]	rw	Shift frequency select for channel 000 _B CMU_CLK0 selected 001 _B CMU_CLK1 selected 010 _B CMU_CLK2 selected 011 _B CMU_CLK3 selected 100 _B CMU_CLK4 selected 101 _B CMU_CLK5 selected 110 _B CMU_CLK6 selected 111 _B CMU_CLK7 selected Note: This register is a shadow register for the CMU_CLKx select. Thus, if the channel should operate on another CMU_CLK then CMU_CKL0 at the beginning, the different CMU_CLK has to be specified inside this register and the CMU_CLK has to be configured with a FORCE_UPDATE in that case before the channel operation would start.
Reserved	15	r	Reserved Read as zero, should be written as zero.
NotUsed	16	rw	Not used in this mode
Reserved	[19:17]	r	Reserved Read as zero, should be written as zero.
NotUsed	20	rw	Not used in this mode
NotUsed	[23:21]	r	Not used in this mode
NotUsed	24	rw	Not used in this mode
Reserved	25	r	Reserved Read as zero, should be written as zero.
OSM	26	rw	One-shot mode 0 _B Continuous shifting is enabled 1 _B Channel stops, after number of bits defined in CM0 is shifted out
NotUsed	27	rw	Not used in this mode
Reserved	[31:28]	r	Reserved Read as zero, should be written as zero.

26.12.4 ATOM Interrupt signals

The following table describes ATOM interrupt signals:

Table 26-40 ATOM interrupt signals

Signal	Description
CCU0TCx_IRQ	CCU0 Trigger condition interrupt for channel x
CCU1TCx_IRQ	CCU1 Trigger condition interrupt for channel x

26.12.5 ATOM Register overview

The following table shows a conclusion of ATOM register address offset and initial values.

Table 26-41 ATOM Register overview

Register name	Description	Details in Section
ATOMi_AGC_GLB_CTRL	AGC Global control register	Section 26.12.6.1
ATOMi_AGC_ENDIS_CTRL	AGC Enable/disable control register	Section 26.11.8.2
ATOMi_AGC_ENDIS_STAT	AGC Enable/disable status register (represents status of ATOM channels)	Section 26.11.8.3
ATOMi_AGC_ACT_TB	AGC Action time base register	Section 26.11.8.4
ATOMi_AGC_OUTEN_CTRL	AGC Output enable control register	Section 26.11.8.5
ATOMi_AGC_OUTEN_STAT	AGC Output enable status register	Section 26.11.8.6
ATOMi_AGC_FUPD_CTRL	AGC Force update control register	Section 26.11.8.7
ATOMi_AGC_INT_TRIG	AGC Internal trigger control register	Section 26.11.8.8
ATOMi_CHx_CTRL	ATOM Channel x control register (x=0...7)	Section 26.12.6.2
ATOMi_CHx_STAT	ATOM Channel x status register (x=0...7)	Section 26.12.6.3
ATOMi_CHx_RDADDR	ATOM Channel x ARU read address register (x=0...7)	Section 26.12.6.4

Generic Timer Module (GTM)
Table 26-41 ATOM Register overview (cont'd)

Register name	Description	Details in Section
ATOMi_CHx_CN0	ATOM Channel x CCU0 counter register (x=0...7)	Section 26.12.6.5
ATOMi_CHx_CM0	ATOM Channel x CCU0 compare register (x=0...7)	Section 26.12.6.6
ATOMi_CHx_SR0	ATOM Channel x CCU0 compare shadow register (x=0...7)	Section 26.12.6.7
ATOMi_CHx_CM1	ATOM Channel x CCU1 compare register (x=0...7)	Section 26.12.6.8
ATOMi_CHx_SR1	ATOM Channel x CCU1 compare shadow register (x=0...7)	Section 26.12.6.9
ATOMi_CHx_IRQ_NOTIFY	ATOM channel x interrupt notification register (x=0...7)	Section 26.12.6.10
ATOMi_CHx_IRQ_EN	ATOM channel x interrupt enable register (x=0...7)	Section 26.12.6.11
ATOMi_CHx_IRQ_FORCINT	ATOM channel x software interrupt generation register (x=0...7)	Section 26.12.6.12
ATOMi_CHx_IRQ_MODE	IRQ mode configuration register (x=0...7)	Section 26.12.6.13

26.12.6 ATOM Register description

All of the following registers are 32-bit only accessible.

26.12.6.1 Register ATOM_i_AGC_GLB_CTRL

GTM_ATOM_i_AGC_GLB_CTRL (i=0-3)

ATOM_i AGC Global control register(0D040_H+i*800_H) **Reset Value: 00000000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPEN_CT RL7		UPEN_CT RL6		UPEN_CT RL5		UPEN_CT RL4		UPEN_CT RL3		UPEN_CT RL2		UPEN_CT RL1		UPEN_CT RL0	
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RST _CH 7	RST _CH 6	RST _CH 5	RST _CH 4	RST _CH 3	RST _CH 2	RST _CH 1	RST _CH 0	Reserved							HOS T_T RIG
w	w	w	w	w	w	w	w	r							w

Field	Bits	Type	Description
HOST_TRIGGER	0	w	<p>Trigger request signal (see AGC) to update the register ENDIS_STAT and OUTEN_STAT</p> <p>0_B no trigger request 1_B set trigger request Read as 0.</p> <p><i>Note:</i> This flag is cleared automatically after triggering the update</p>
Reserved	[7:1]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>
RST_CH0	8	w	<p>Software reset of channel 0</p> <p>0_B No action 1_B Reset channel Read as 0.</p> <p><i>Note:</i> This bit is cleared automatically after write by CPU. The channel registers are set to their reset values and channel operation is stopped immediately. The S-r FlipFlop SOUR is reset to '1'.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
RST_CH1	9	w	Software reset of channel 1 See bit 8
RST_CH2	10	w	Software reset of channel 2 See bit 8
RST_CH3	11	w	Software reset of channel 3 See bit 8
RST_CH4	12	w	Software reset of channel 4 See bit 8
RST_CH5	13	w	Software reset of channel 5 See bit 8
RST_CH6	14	w	Software reset of channel 6 See bit 8
RST_CH7	15	w	Software reset of channel 7 See bit 8
UPEN_CT RL0	[17:16]	rw	ATOM channel 0 enable update of register CM0, CM1 and CLK_SRC from SR0, SR1 and CLK_SRC_SR Write of following double bit values is possible: 00 _B don't care, bits 1:0 will not be change 01 _B update disabled: is read as 00 (see below) 10 _B update enabled: is read as 11 (see below) 11 _B don't care, bits 1:0 will not be changed Read of following double values means: 00 _B channel disabled 11 _B channel enabled
UPEN_CT RL1	[19:18]	rw	ATOM channel 1 enable update of register CM0, CM1 and CLK_SRC See bits 17:16
UPEN_CT RL2	[21:20]	rw	ATOM channel 2 enable update of register CM0, CM1 and CLK_SRC See bits 17:16
UPEN_CT RL3	[23:22]	rw	ATOM channel 3 enable update of register CM0, CM1 and CLK_SRC See bits 17:16
UPEN_CT RL4	[25:24]	rw	ATOM channel 4 enable update of register CM0, CM1 and CLK_SRC See bits 17:16

Generic Timer Module (GTM)

Field	Bits	Type	Description
UPEN_CTL RL5	[27:26]	rw	ATOM channel 5 enable update of register CM0, CM1 and CLK_SRC See bits 17:16
UPEN_CTL RL6	[29:28]	rw	ATOM channel 6 enable update of register CM0, CM1 and CLK_SRC See bits 17:16
UPEN_CTL RL7	[31:30]	rw	ATOM channel 7 enable update of register CM0, CM1 and CLK_SRC See bits 17:16

26.12.6.2 Register ATOMi_CHx_CTRL (x: 0...7)
GTM_ATOM0_CHx_CTRL (x=0-7)
ATOM0 Channel x Control Register
 $(0D004_H + x * 0080_H)$
Reset Value: 00000800_H
GTM_ATOM1_CHx_CTRL (x=0-7)
ATOM1 Channel x Control Register
 $(0D804_H + x * 0080_H)$
Reset Value: 00000800_H
GTM_ATOM2_CHx_CTRL (x=0-7)
ATOM2 Channel x Control Register
 $(0E004_H + x * 0080_H)$
Reset Value: 00000800_H
GTM_ATOM3_CHx_CTRL (x=0-7)
ATOM3 Channel x Control Register
 $(0E804_H + x * 0080_H)$
Reset Value: 00000800_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				ABM	OSM	SLA	TRIG OUT	Reserved			RST _CC U0	Reserved			WR REQ
r				rw	rw	rw	rw	r			rw	r			rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rese rved	CLK_SRCCLK_ SRC_SR		SL	Rese rved	CMP _CT RL	ACB					ARU _EN	TB12 _SE L	MODE		
r	rw		rw	r	rw	rw					rw	rw	rw		

Generic Timer Module (GTM)

Field	Bits	Type	Description
MODE	[1:0]	rw	ATOM channel mode select 00 _B ATOM Signal Output Mode Immediate (SOMI) 01 _B ATOM Signal Output Mode Compare (SOMC) 10 _B ATOM Signal Output Mode PWM (SOMP) 11 _B ATOM Signal Output Mode Serial (SOMS)
TB12_SEL	2	rw	Select time base value TBU_TS1 or TBU_TS2. 0 _B TBU_TS1 selected for comparison 1 _B TBU_TS2 selected for comparison Note: this bit is only applicable in SOMC mode.
ARU_EN	3	rw	ARU Input stream enable 0 _B ARU Input stream disabled 1 _B ARU Input stream enabled
ACB	[8:4]	rw	ATOM Mode control bits Note: These bits have different meaning in the different ATOM channel modes. Please refer to the mode description sections. Note: These bits are only applicable when ARU_EN = '0'.
CMP_CTR L	9	rw	CCUx compare strategy select 0 _B Greater/equal compare against TBU time base values (TBU_TSx >= CMx) 1 _B Less/equal compare against TBU time base values (TBU_TSx <= CMx) Note: this bit is only applicable in SOMC mode.
Reserved	10	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

Field	Bits	Type	Description
SL	11	rw	<p>Initial signal level</p> <p>0_B Low signal level 1_B High signal level</p> <p>Note: If the output is disabled, the output ATOM_OUT[x] is set to inverse SL independent of ATOM channel mode.</p> <p>Note: In SOMP, SOMI, and SOMS mode, if the channel is disabled, the internal register SOUR inside ATOM sub unit SOU is set to inverse value of SL. By enabling the channel the register SOUR is not changed. Thus, if the output is enabled afterwards, the output ATOM_OUT[x] is the inverse value of SL.</p> <p>Note: In SOMC mode, if the channel is disabled, the internal register SOUR inside ATOM sub unit SOU is set to value of SL. By enabling the channel the register SOUR is not changed. Thus, if the output is enabled and the channel is disabled, the output ATOM_OUT[x] is the value of SL.</p> <p>Note: In SOMS mode, this bit is only applicable when the channel and its outputs are disabled.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
CLK_SRC CLK_SRC _SR	[14:12]	rw	<p>actual CMU clock source (SOMS)/ shadow register for CMU clock source (SOMP)</p> <p>000_B CMU_CLK0 selected 001_B CMU_CLK1 selected 010_B CMU_CLK2 selected 011_B CMU_CLK3 selected 100_B CMU_CLK4 selected 101_B CMU_CLK5 selected 110_B CMU_CLK6 selected 111_B CMU_CLK7 selected</p> <p>Note: This register is a shadow register for the CMU_CLKx select. Thus, if the CMU_CLK source for PWM generation should be changed during operation, the old CMU_CLK has to operate until the update of the ATOM channels internal CLK_SRC register by the CLK_SRC_SR content is done either by an end of a period or a FORCE_UPDATE.</p> <p>Note: After (channel) reset the selected CLK_SRC value is the SYS_CLK (input of Global Clock Divider). To use in SOMP mode one of the CMU_CLKx, it is required to perform a forced update of CLK_SRC with the value of CLK_SRC_SR value before/with enabling the channel.</p>
Reserved	15	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>
WR_REQ	16	rw	<p>CPU Write request bit for late compare register update</p>
Reserved	[19:17]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>
RST_CCU 0	20	rw	<p>Reset source of CCU0</p> <p>0_B Reset counter register CN0 to 0 on matching comparison with CM0 1_B Reset counter register CN0 to 0 on trigger TRIG[x-1]</p> <p>Note: If RST_CCU0=1 and UPEN_CTRLx=1 are set , TRIG[x-1] triggers also the update of work register (CM0, CM1, and CLK_SRC).</p>
Reserved	[23:21]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRIGOUT	24	rw	<p>Trigger output selection (output signal TRIG_CHx) of module ATOM_CHx.</p> <p>0_B TRIG_x[is TRIG_x-1]</p> <p>1_B TRIG_x[is TRIG_CCU0</p> <p>Note: this bit is only applicable in SOMC and SOMP mode.</p>
SLA	25	rw	<p>Serve last ARU communication strategy</p> <p>0_B Capture SRx time stamps after CCU0 match event not provided to ARU</p> <p>1_B Capture SRx time stamps after CCU0 match event provided to ARU</p> <p>Note: This bit is only applicable in SOMC mode.</p> <p>Note: Please note, that setting of this bit has only effect, when ACBI(4:2) is configured for serve last compare strategy ("100", "101", or "110").</p> <p>Note: When this bit is not set, the captured time stamps in the shadow registers SRx are only provided after the CCU1 match occurred. The ACBO(4:3) bits always return "10" in that case.</p> <p>Note: By setting this bit, the ATOM channel also provides the captured time stamps after the CCU0 match event to the ARU. The ACBO(4:3) bits are set to "01" in that case. After the CCU1 match event, the time stamps are captured again in the SRx registers and provided to the ARU. The ACBO(4:3) bits are set to "10". When the data in the shadow registers after the CCU0 match was not consumed by an ARU destination and the CCU1 match occurs, the data in the shadow registers is overwritten by the new captured time stamps. The ATOM channel does not request new data from the ARU when the CCU0 match values are read from an ARU destination.</p>
OSM	26	rw	<p>One-shot mode</p> <p>0_B Continuous PWM generation after channel enable</p> <p>1_B A single pulse is generated</p> <p>Note: this bit is only applicable in SOMP and SOMS modes.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
ABM	27	rw	ARU blocking mode 0 _B ARU blocking mode disabled: ATOM reads continuously from ARU and updates CM0, CM1 independent of pending compare match event 1 _B ARU blocking mode enabled: after updating CM0,CM1 via ARU, no new data is read from ARU until compare match event is occurred. Note: this bit is only applicable in SOMC mode.
Reserved	[31:28]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.12.6.3 Register ATOMi_CHx_STAT (x: 0...7)

GTM_ATOM0_CHx_STAT (x=0-7)

ATOM0 Channel x Status Register

 $(0D01C_H + x * 0080_H)$

 Reset Value: 00000000_H

GTM_ATOM1_CHx_STAT (x=0-7)

ATOM1 Channel x Status Register

 $(0D81C_H + x * 0080_H)$

 Reset Value: 00000000_H

GTM_ATOM2_CHx_STAT (x=0-7)

ATOM2 Channel x Status Register

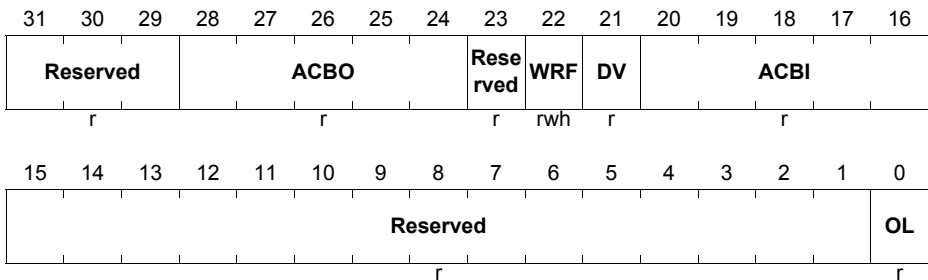
 $(0E01C_H + x * 0080_H)$

 Reset Value: 00000000_H

GTM_ATOM3_CHx_STAT (x=0-7)

ATOM3 Channel x Status Register

 $(0E81C_H + x * 0080_H)$

 Reset Value: 00000000_H


Field	Bits	Type	Description
OL	0	r	Actual output signal level of ATOM_CHx_OUT 0 _B Actual output signal level is low 1 _B Actual output signal level is high
Reserved	[15:1]	r	Reserved Read as zero, should be written as zero
ACBI	[20:16]	r	ATOM Mode control bits received through ARU Note: This register serves as a mirror for the five ARU control bits received through the ARU interface. The bits are valid, when the DV bit is set.

Generic Timer Module (GTM)

Field	Bits	Type	Description
DV	21	r	Valid ARU Data stored in compare registers 0 _B No valid data was received by ARU 1 _B Valid data received by ARU and stored in CM0 and/or CM1 Note: This bit is only applicable in SOMC mode. The CPU can determine the status of the ARU transfers with this bit. After the compare event occurred, the bit is reset by hardware.
WRF	22	rwh	Write request of CPU failed for late update 0 _B Late update was successful, CCUx units wait for comparison. 1 _B Late update failed. The bit WRF can be reset by writing a 1 to it. Note: This bit is only applicable in SOMC mode.
Reserved	23	r	Reserved Read as zero, should be written as zero
ACBO	[28:24]	r	ATOM Internal status bits ACBO[3] = 1: CCU0 Compare match occurred ACBO[4] = 1: CCU1 Compare match occurred Note: These bits are only set in SOMC mode. Note: ACBO is reset to 0b00000 on an update of register CM0 or CM1 (via ARU or CPU) Note: In SMOC mode these bits sent as ARU control bits 52...48.
Reserved	[31:29]	r	Reserved Read as zero, should be written as zero

26.12.6.4 Register ATOMi_CHx_RDADDR (x: 0...7)

GTM_ATOM0_CHx_RDADDR (x=0-7)

ATOM0 Channel x ARU Read Address Register

(0D000_H+x*0080_H)

Reset Value: 01FE01FE_H

GTM_ATOM1_CHx_RDADDR (x=0-7)

ATOM1 Channel x ARU Read Address Register

(0D800_H+x*0080_H)

Reset Value: 01FE01FE_H

GTM_ATOM2_CHx_RDADDR (x=0-7)

ATOM2 Channel x ARU Read Address Register

(0E000_H+x*0080_H)

Reset Value: 01FE01FE_H

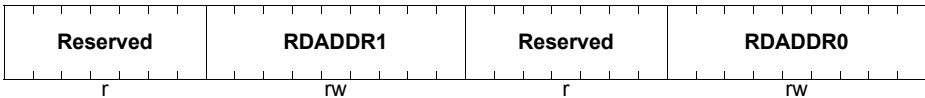
GTM_ATOM3_CHx_RDADDR (x=0-7)

ATOM3 Channel x ARU Read Address Register

(0E800_H+x*0080_H)

Reset Value: 01FE01FE_H

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0



Field	Bits	Type	Description
RDADDR0	[8:0]	rw	<p>ARU Read address 0</p> <p>Note: This read address is used by the ATOM channel to receive data from ARU immediately after the channel and ARU access is enabled (see ATOMi_CHx_CTRL register for details).</p> <p>Note: This bit field is only writable if channel is disabled.</p>
Reserved	[15:9]	r	<p>Read as zero, should be written as zero</p> <p>Read as zero, should be written as zero.</p>
RDADDR1	[24:16]	rw	<p>ARU Read address 1</p> <p>Note: The ATOM channel switches to this read address, when requested in the ARU control bits 52 to 48 with the pattern "111--". The channel switches back to the RDADDR0 after one ARU data package was received on RDADDR1.</p> <p>Note: This read address is only applicable in SOMC mode.</p>
Reserved	[31:25]	r	<p>Reserved</p> <p>Read as zero, should be written as zero</p>

Generic Timer Module (GTM)

26.12.6.5 Register ATOMi_CHx_CN0 (x: 0...7)

GTM_ATOM0_CHx_CN0 (x=0-7)

ATOM0 Channel x CCU0 Counter Register

(0D018_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM1_CHx_CN0 (x=0-7)

ATOM1 Channel x CCU0 Counter Register

(0D818_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM2_CHx_CN0 (x=0-7)

ATOM2 Channel x CCU0 Counter Register

(0E018_H+x*0080_H)

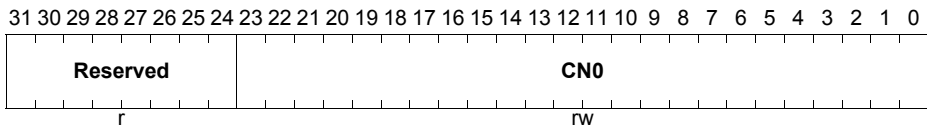
Reset Value: 00000000_H

GTM_ATOM3_CHx_CN0 (x=0-7)

ATOM3 Channel x CCU0 Counter Register

(0E818_H+x*0080_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CN0	[23:0]	rw	ATOM CCU0 counter register
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.12.6.6 Register ATOMi_CHx_CM0 (x: 0...7)

GTM_ATOM0_CHx_CM0 (x=0-7)

ATOM0 Channel x CCU0 Compare Register

(0D010_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM1_CHx_CM0 (x=0-7)

ATOM1 Channel x CCU0 Compare Register

(0D810_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM2_CHx_CM0 (x=0-7)

ATOM2 Channel x CCU0 Compare Register

(0E010_H+x*0080_H)

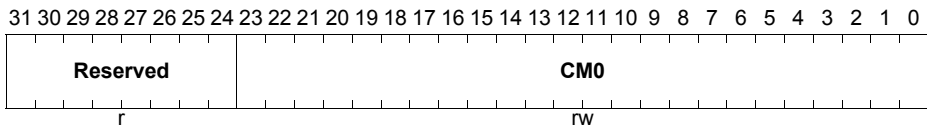
Reset Value: 00000000_H

GTM_ATOM3_CHx_CM0 (x=0-7)

ATOM3 Channel x CCU0 Compare Register

(0E810_H+x*0080_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CM0	[23:0]	rw	ATOM CCU0 compare register Note: This register is write protected in SOMC mode and returns AEI_STATUS='10' on write access, when in serve last compare strategy the first match of CCU0 occurred.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.12.6.7 Register ATOMi_CHx_SR0 (x: 0...7)

GTM_ATOM0_CHx_SR0 (x=0-7)

ATOM0 Channel x CCU0 Compare Shadow Register

(0D008_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM1_CHx_SR0 (x=0-7)

ATOM1 Channel x CCU0 Compare Shadow Register

(0D808_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM2_CHx_SR0 (x=0-7)

ATOM2 Channel x CCU0 Compare Shadow Register

(0E008_H+x*0080_H)

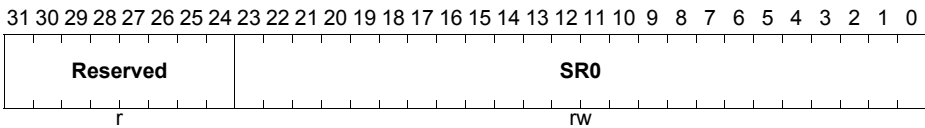
Reset Value: 00000000_H

GTM_ATOM3_CHx_SR0 (x=0-7)

ATOM3 Channel x CCU0 Compare Shadow Register

(0E808_H+x*0080_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
SR0	[23:0]	rw	ATOM channel x shadow register SR0 Note: The SR0 register is used as shadow register for CM0 in SOMP and SOMS modes and is used as capture register for time base TBU_TS0 in SOMC mode.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.12.6.8 Register ATOMi_CHx_CM1 (x: 0...7)

GTM_ATOM0_CHx_CM1 (x=0-7)

ATOM0 Channel x CCU1 Compare Register

(0D014_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM1_CHx_CM1 (x=0-7)

ATOM1 Channel x CCU1 Compare Register

(0D814_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM2_CHx_CM1 (x=0-7)

ATOM2 Channel x CCU1 Compare Register

(0E014_H+x*0080_H)

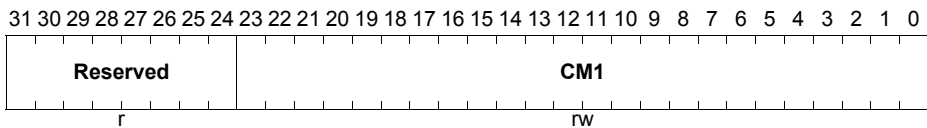
Reset Value: 00000000_H

GTM_ATOM3_CHx_CM1 (x=0-7)

ATOM3 Channel x CCU1 Compare Register

(0E814_H+x*0080_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CM1	[23:0]	rw	ATOM CCU1 compare register Note: This register is write protected in SOMC mode and returns AEI_STATUS='10' on write access, when in serve last compare strategy the first match of CCU0 occurred.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.12.6.9 Register ATOMi_CHx_SR1 (x: 0...7)

GTM_ATOM0_CHx_SR1 (x=0-7)

ATOM0 Channel x CCU1 Compare Shadow Register

(0D00C_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM1_CHx_SR1 (x=0-7)

ATOM1 Channel x CCU1 Compare Shadow Register

(0D80C_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM2_CHx_SR1 (x=0-7)

ATOM2 Channel x CCU1 Compare Shadow Register

(0E00C_H+x*0080_H)

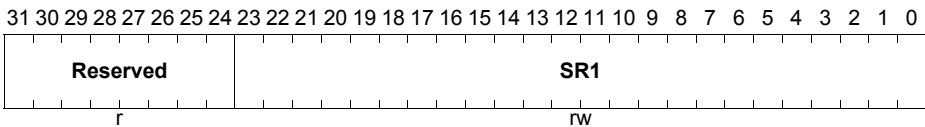
Reset Value: 00000000_H

GTM_ATOM3_CHx_SR1 (x=0-7)

ATOM3 Channel x CCU1 Compare Shadow Register

(0E80C_H+x*0080_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
SR1	[23:0]	rw	ATOM channel x shadow register SR0 Note: The SR1 register is used as shadow register for CM1 in SOMP and SOMS modes and is used as capture register for time base TBU_TS1 or TBU_TS2 (when selected in ATOMi_CHx_CTRL register) in SOMC mode.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

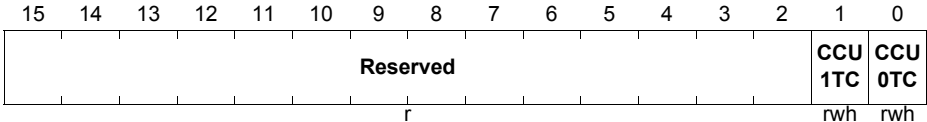
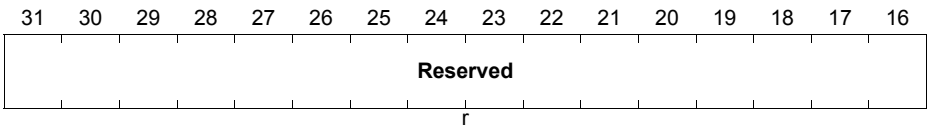
26.12.6.10 Register ATOMi_CHx_IRQ_NOTIFY (x:0...7)

GTM_ATOM0_CHx_IRQ_NOTIFY (x=0-7)
 ATOM0 Channel x Interrupt Notification Register
 (0D020_H+x*0080_H) Reset Value: 00000000_H

GTM_ATOM1_CHx_IRQ_NOTIFY (x=0-7)
 ATOM1 Channel x Interrupt Notification Register
 (0D820_H+x*0080_H) Reset Value: 00000000_H

GTM_ATOM2_CHx_IRQ_NOTIFY (x=0-7)
 ATOM2 Channel x Interrupt Notification Register
 (0E020_H+x*0080_H) Reset Value: 00000000_H

GTM_ATOM3_CHx_IRQ_NOTIFY (x=0-7)
 ATOM3 Channel x Interrupt Notification Register
 (0E820_H+x*0080_H) Reset Value: 00000000_H



Field	Bits	Type	Description
CCU0TC	0	rwh	CCU0 Trigger condition interrupt for channel x 0 _B No interrupt occurred. 1 _B CCU0 Trigger condition interrupt was raised by ATOM channel x. Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
CCU1TC	1	rwh	CCU1 Trigger condition interrupt for channel x See bit 0.
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.12.6.11 Register ATOMi_CHx_IRQ_EN (x:0...7)

GTM_ATOM0_CHx_IRQ_EN (x=0-7)

ATOM0 Channel x Interrupt Enable Register

(0D024_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM1_CHx_IRQ_EN (x=0-7)

ATOM1 Channel x Interrupt Enable Register

(0D824_H+x*0080_H)

Reset Value: 00000000_H

GTM_ATOM2_CHx_IRQ_EN (x=0-7)

ATOM2 Channel x Interrupt Enable Register

(0E024_H+x*0080_H)

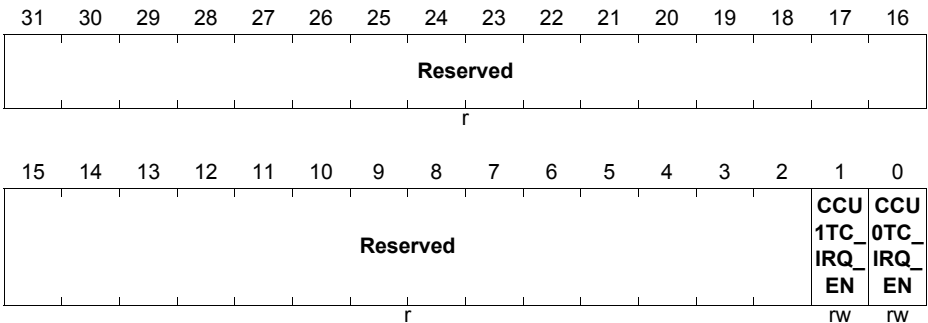
Reset Value: 00000000_H

GTM_ATOM3_CHx_IRQ_EN (x=0-7)

ATOM3 Channel x Interrupt Enable Register

(0E824_H+x*0080_H)

Reset Value: 00000000_H



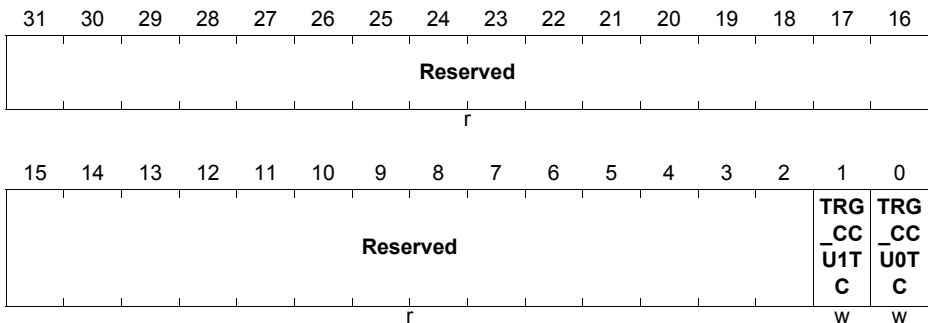
Field	Bits	Type	Description
CCU0TC_IRQ_EN	0	rw	ATOM_CCU0TC_IRQ interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM. 1 _B Enable interrupt, interrupt is visible outside GTM.
CCU1TC_IRQ_EN	1	rw	ATOM_CCU1TC_IRQ interrupt enable See bit 0.
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

26.12.6.12 Register ATOMi_CHx_IRQ_FORCINT (x:0...7)
GTM_ATOM0_CHx_IRQ_FORCINT (x=0-7)
ATOM0 Channel x Software Interrupt Generation Register

 (0D028_H+x*0080_H) Reset Value: 00000000_H
GTM_ATOM1_CHx_IRQ_FORCINT (x=0-7)
ATOM1 Channel x Software Interrupt Generation Register

 (0D828_H+x*0080_H) Reset Value: 00000000_H
GTM_ATOM2_CHx_IRQ_FORCINT (x=0-7)
ATOM2 Channel x Software Interrupt Generation Register

 (0E028_H+x*0080_H) Reset Value: 00000000_H
GTM_ATOM3_CHx_IRQ_FORCINT (x=0-7)
ATOM3 Channel x Software Interrupt Generation Register

 (0E828_H+x*0080_H) Reset Value: 00000000_H


Field	Bits	Type	Description
TRG_CC U0TC	0	w	Trigger ATOM_CCU0TC_IRQ interrupt by software 0 _B No interrupt triggering. 1 _B Assert CCU0TC_IRQ interrupt for one clock cycle. Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
TRG_CC U1TC	1	w	Trigger ATOM_CCU1TC_IRQ interrupt by software 0 _B No interrupt triggering. 1 _B Assert CCU1TC_IRQ interrupt for one clock cycle. Note: This bit is cleared automatically after interrupt is released.
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

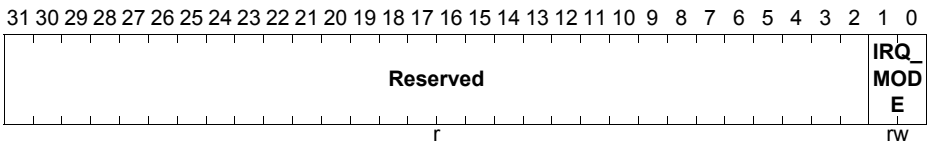
26.12.6.13 Register ATOMi_CHx_IRQ_MODE (x:0...7)

GTM_ATOM0_CHx_IRQ_MODE (x=0-7)
 ATOM0 IRQ Mode Configuration Register
 (0D02C_H+x*0080_H) Reset Value: 00000000_H

GTM_ATOM1_CHx_IRQ_MODE (x=0-7)
 ATOM1 IRQ Mode Configuration Register
 (0D82C_H+x*0080_H) Reset Value: 00000000_H

GTM_ATOM2_CHx_IRQ_MODE (x=0-7)
 ATOM2 IRQ Mode Configuration Register
 (0E02C_H+x*0080_H) Reset Value: 00000000_H

GTM_ATOM3_CHx_IRQ_MODE (x=0-7)
 ATOM3 IRQ Mode Configuration Register
 (0E82C_H+x*0080_H) Reset Value: 00000000_H



Field	Bits	Type	Description
IRQ_MODE	[1:0]	rw	IRQ mode selection 00 _B Level mode 01 _B Pulse mode 10 _B Pulse-Notify mode 11 _B Single-Pulse mode Note: The interrupt modes are described in Section 26.2.5 .
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

26.13 Multi Channel Sequencer (MCS)

26.13.1 Overview

The Multi Channel Sequencer (MCS) submodule is a generic data processing module that is connected to the ARU.

One of its major applications is to calculate complex output sequences that may depend on the time base values of the TBU and are processed in combination with the ATOM submodule.

Other applications can use the MCS submodule to perform extended data processing of input data resulting from the TIM submodule that are provided to the CPU (e.g. using the PSM submodule).

Moreover, some applications may process data provided by the CPU within the MCS submodule, and the calculated results are sent to the outputs using the ATOM submodules.

26.13.1.1 Architecture

Figure 26-58 gives an overview of the MCS architecture.

MCS architecture

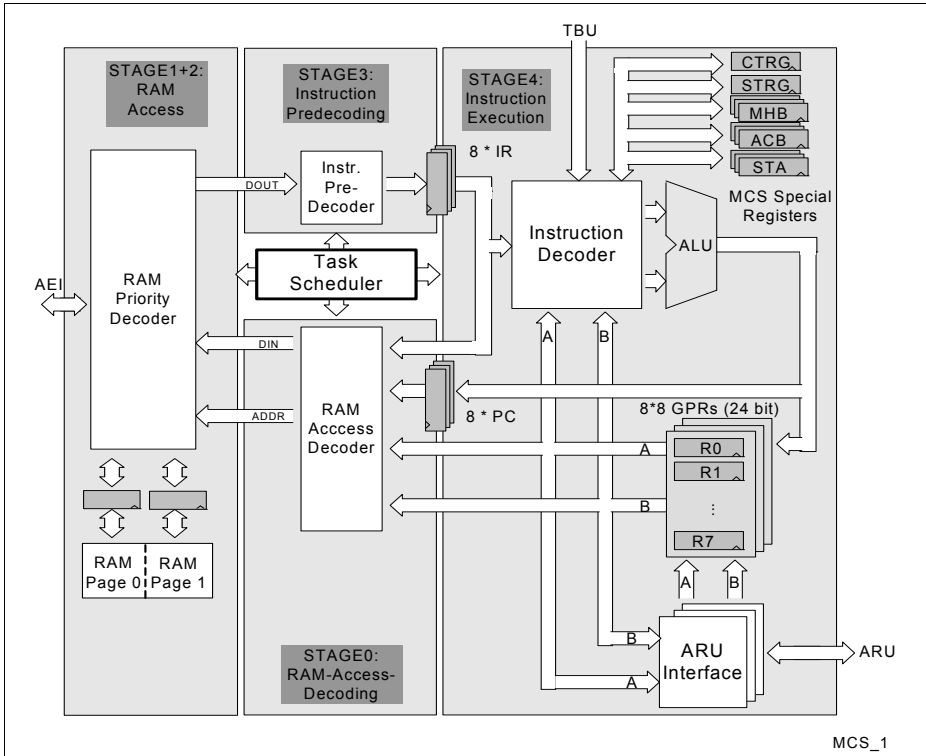


Figure 26-58 MCS architecture

The MCS submodule mainly embeds a single data path with five pipeline stages, consisting of a simple 24-bit Arithmetic Logic Unit (ALU), several decoders, and a connection to two RAM pages located outside of the MCS submodule.

The data path of the MCS is shared by eight so called MCS-channels, whereas each MCS-channel executes a dedicated micro-program that is stored inside the RAM pages connected to the MCS submodule. The execution of the different MCS-channels is controlled by a central task scheduler.

Both RAM pages may contain arbitrary sized code and data sections that are accessible by all MCS-channels and the CPU via AEI.

The MCS submodule supports a memory layout of up to 2^{12} memory locations each 32 bit wide leading to a maximum address range from 0 to $2^{14}-4$.

Generic Timer Module (GTM)

This address space of the MCS is divided into two seamless memory pages.

Memory page 0 begins from address 0 and ranges to address MP0-[i] and memory page 1 ranges from MP0 to MP1-[i].

The parameters MP0 and MP1 are defined externally by the memory configuration submodule MCFG of [Section 26.14](#).

If an ECC Error occurs while an MCS-channel reads data from a memory page, the corresponding MCS-channel is disabled and the ERR bit in register STA is raised.

After a resetting the MCS submodule, the content of both memory pages is initialized with zeros.

An MCS-channel can also be considered as an individual task of a processor that is scheduled at a specific point in time.

A more detailed description of the scheduling can be found in [Section 26.13.1.2](#).

Moreover, each MCS-channel has a dedicated ARU interface for communication with other ARU connected modules, an Instruction Register (IR), a Program Counter Register (PC), a Status Register (STA), an ARU Control Bit Register (ACB), a Memory High Byte Register (MHB) and a Register Bank with eight 24 bit general purpose registers (R0, R1, ...R7).

All the registers, mentioned above, are only visible within its dedicated MCS-channel and thus the MCS-channels cannot exchange data using registers.

The only exception is the common 16 bit wide trigger register that can be accessed by all MCS channels in order to trigger other MCS-channels located in the same submodule. STRG is used to set a bits and CTRG is used to clear bits in the trigger register.

To enable triggering of MCS-channels by CPU, it can set bits in the common trigger register by writing to MCS[i]_STRG and clear bits by writing to MCSi_CTRG.

Whenever data has to be exchanged between different MCS-channels, the connected RAM pages, which are accessible by all MCS-channels, must be used.

However, since the data registers are writable by the CPU, an MCS channel may also exchange data with the CPU using its data registers.

The main actions of the different pipeline stages are as follows:

Pipeline stage 0 performs a setup of address, input data, and control signals for the next RAM access of a specific MCS-channel.

The actual RAM access of a specific MCS-channel is executed in pipeline stage 1 and 2, assuming an external connection of a synchronous RAM with a latency of one clock cycle.

The RAM priority decoder arbitrates RAM accesses that are requested by the CPU via AEI and by the active MCS-channel.

Generic Timer Module (GTM)

If both, CPU and an MCS-channel request a memory access to the same memory page the MCS-channel is prioritized.

Pipeline stage 3 performs pre-decoding of instruction and data resulting from the RAM.

Finally, in pipeline stage 4 the current instruction is executed.

Since the internal registers of the MCS can be updated by different sources (MCS write access, AEI write access and ARU read access) a write conflict occurs if more than one source wants to write to the same register. In this cascade the result of the register is unpredictable.

However, the software should setup its application in a way that such conflicts do not occur.

One exception is the common trigger register which may be written by multiple sources (different MCS channels and AEI) in order to enable fast triggering of different MCS channels. Typically, the software should setup its application in a manner that different sources should not write the same bits in the trigger register.

26.13.1.2 Scheduling

The MCS submodule provides two different scheduling schemes: round-robin schedule and accelerated schedule.

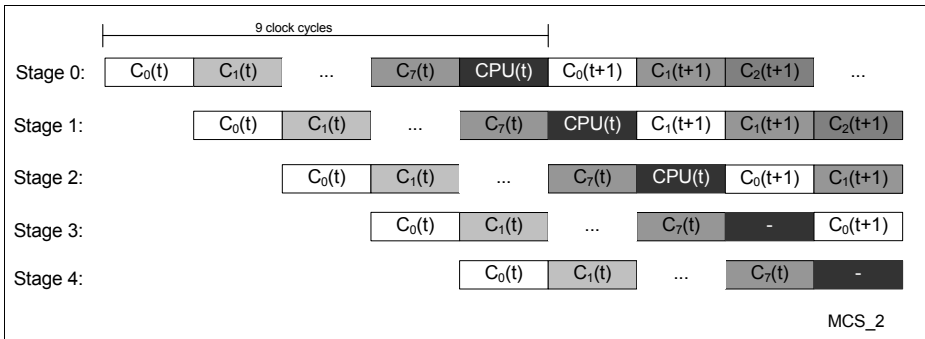
The scheduling scheme can be selected by the SCHED bit in the global MCSi_CTRL register.

The round-robin order scheduling assigns all MCS-channels an equal amount of time slices.

In addition, the scheduler also assigns one time slice to the CPU, in order to guarantee at least one memory access by the CPU within each round-trip cycle.

Figure 26-59 shows the round-robin scheduling with 8 MCS-channels (C_0 to C_7) that are scheduled together with a single CPU access.

Scheduling

Generic Timer Module (GTM)

Figure 26-59 Scheduling

The figure also shows which MCS-channel is activated in specific pipeline stage at a specific point in time.

The execution time of an MCS-channel in a specific pipeline stage is always one clock cycle.

The index t marks all instruction parts of the corresponding MCS-channels belonging to the same round-trip cycle.

Consequently, a single cycle instruction of an MCS-channel requires an effective execution time of 9 clock cycles, ignoring the five clock cycles of pipeline latency.

To improve memory bandwidth between CPU and MCS memory, the time slices of any suspended MCS-channel is also granted to the CPU.

An MCS-channel can be suspended due to the following reasons:

- An MCS-channel is executing a blocking read or write request to an ARU connected submodule (instruction ARD, AWR, ARDI, AWRI).
- An MCS-channel waits on a register match event (instruction WURM), in order to wait on a time base value or a trigger event from another MCS channel.
- An MCS-channel is disabled.

The round-robin scheduling leads to a deterministic round trip time for the whole submodule, however it may waste clock cycles by scheduling MCS-channels that are not able to run at a specific point in time assuming that there is no high CPU bandwidth required.

The second scheduling mode, the accelerated scheduling mode, improves the computational performance of the MCS in the following way:

Initially, the accelerated scheduling mode behaves similar to round-robin scheduling mode.

Whenever the scheduler detects an MCS-channel that is suspended, due to its suspended state (or it is already scheduled in stage 0, 1, 2, or 3), the scheduler is

Generic Timer Module (GTM)

selecting the next non-suspended MCS-channel that would follow if round-robin scheduling is continued.

Considering the timing of [Figure 26-59](#) in conjunction with the accelerated scheduling scheme, a single cycle instruction of an MCS-channel requires an effective execution time between 5 and 9 clock cycles, depending on the number of suspended MCS-channels.

In summary, the round-robin scheduling mode grants time slices of suspended MCS-channels to the CPU and the accelerated scheduling mode grants time slices of suspended MCS-channels to non-suspended MCS-channels.

26.13.2 Instruction Set

This section describes the entire instruction set of the MCS submodule.

After the introduction of the different instruction formats in [Section 26.13.2.1](#), the individual instructions are described in [Section](#) to [Section 26.13.2.7](#).

In general, each instruction is 32 bit wide but the duration of each instruction varies between several instruction cycles. An instruction cycle is defined as the time in SYS_CLK clock cycles that rest between two consecutive instructions of a channel.

As already described in [Section 26.13.1.2](#), the number of required clock cycles for a single instruction cycle can vary in the range of 5 to 9 clock cycles, depending on the number suspended MCS-channels, when the accelerated scheduling scheme is selected inside the MCSi_CTRL register.

In the round robin scheduling scheme, each single cycle instruction takes exactly 9 clock cycles.

Before the instruction formats and the actual instructions are described, some commonly used terms, abbreviations and expressions are introduced:

OREG: The operation register set $OREG = \{R0, R1, R2, \dots, R7, STA, ACB, CTRG, STRG, TBU_TS0, TBU_TS1, TBU_TS2, MHB\}$ includes all MCS accessible internal registers, as well as the global time bases TBU_TS0, TBU_TS1, and TBU_TS2 that are provided by the submodule TBU.

AREG: The ARU register set $AREG = \{R0, R1, R2, \dots, R7, ZERO\}$ includes the all registers that can be written by incoming ARU transfers (ARD, ARDI instructions). These registers include all eight general purpose registers. The dummy register ZERO may be used to discard an incoming 24 bit ARU word.

LIT4: The set $LIT4 = \{0, 1, \dots, 15\}$ is an unsigned 4 bit literal.

LIT16: The set $LIT16 = \{0, 1, \dots, 2^{16} - 1\}$ is an unsigned 16 bit literal.

LIT24: The set $LIT24 = \{0, 1, \dots, 2^{24} - 1\}$ is an unsigned 24 bit literal.

BIT SELECTION: The expression $VAR[i]$ represents the i-th bit of a variable VAR.

Generic Timer Module (GTM)

BIT RANGE SELECTION: The expression VAR[m:n] represents the bit slice of variable VAR that is ranging from bit n to bit m.

MEMORY ADDRESSING: The expression MEM(X) represents the 32 bit value at address X of the memory.

Address X ranges between 0 and $2^{14}-4$, whereas X must be an integral multiple of 4.

The expression MEM(X)[m:n] represents the bit slice ranging from bit n to m of the 32 bit word at memory location X.

ARU ADDRESSING: In the case of ARU reading, the expression ARU(X) represents the 53 bit ARU word of ARU channel at address X.

The read address X ranges between 0 and 2^9-1 .

In the case of ARU writing, the expression ARU(X) represents a 53 bit ARU word that is written to an ARU channel indexed by the index X.

The index X selects a single ARU write channel from the pool of the MCS submodule's allocated ARU write channels.

An MCS submodule has 24 ARU write channels, indexed by values 0 to 23.

The expression ARU(X)[m:n] represents the bit slice ranging from bit n to m of the 53 bit ARU word.

26.13.2.1 Instruction Format

The first instruction format, called literal instruction format, embeds a primary 4 bit opcode OPC0, a 24 bit literal value $C \in \text{LIT}_{24}$, and a 4 bit value A, which may be an element of set OREG or AREG, depending on the actual instruction.

Figure 26-60 shows the bit alignment of the literal instruction format.

Literal Instruction format

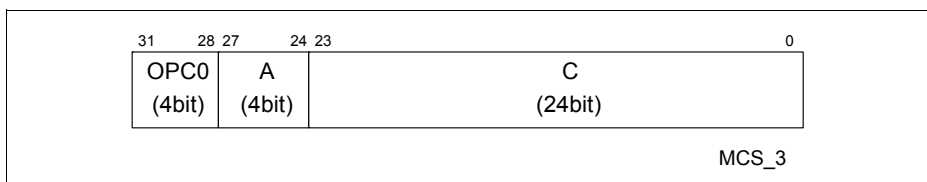


Figure 26-60 Literal Instruction format

The literal instruction format is primarily used for instructions that are accessing a 24 bit literal and a single 24 bit register as operands.

Generic Timer Module (GTM)

In the following subsections the binary codes of a 24 bit literal instruction is defined as "xxxxaaaaccccccccccccccccccccc", whereas the digits 'x' encode the field OPC0, the digits 'a' encode the operand field A, and the digits 'c' encode the 24 bit literal field C.

If an instruction ignores several bits of field, the bits are defined as '-' in its code.

The second instruction format, called double operand instruction format, embeds a 4 bit primary opcode OPC0, a 4 bit secondary opcode OPC1, a 16 bit literal C ∈ LIT16 and two 4 bit values A and B, which may be an element of set OREG, AREG, or LIT4 depending on the actual instruction.

Figure 26-61 shows the bit alignment of the double operand instruction format.

Double Operand Instruction format

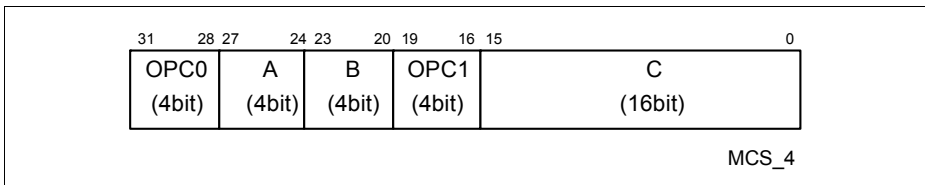


Figure 26-61 Double Operand Instruction format

The double operand instruction format is primarily used for instructions that are accessing two operands stored in the 24 bit registers.

In the following subsections the binary codes of a 16 bit literal instruction is defined as "xxxxaaaabbbbyyycccccccccccccc", whereas the digits 'x' encode the bit field OPC0, 'y' the digits of field OPC2, the digits 'a' encode the operand field A, the digits 'b' the operand field B, and the digits 'c' encode the 16 bit literal field C.

If an instruction ignores several bits of field, the bits are defined as '-' in its code.

26.13.2.2 Data Transfer Instructions

MOVL Instruction

Syntax: MOVL A, C

Operation: A ← C

Status: Z

Duration: 1 instruction cycle

Code: 0001aaaaccccccccccccccccccccc

Description: Transfer literal value C (C ∈ LIT24) to register A (A ∈ OREG).

Generic Timer Module (GTM)

The zero bit Z of status register STA is set, if the transferred value is zero, otherwise the zero bit is cleared.

The program counter PC is increment by the value 4.

MOV Instruction

Syntax: MOV A, B

Operation: $A \leftarrow B$

Status: Z

Duration: 1 instruction cycle

Code: 1010aaaabbbb0000-----

Description: Transfer value B ($B \in \text{OREG}$) to register A ($A \in \text{OREG}$).

The zero bit Z of status register STA is set, if the transferred value is zero, otherwise the zero bit is cleared.

The program counter PC is increment by the value 4.

MRD Instruction

Syntax: MRD A, C

Operation: $A \leftarrow \text{MEM}(C[13:0])[23:0]$;
 $\text{MHB} \leftarrow \text{MEM}(C[13:0])[31:24]$

Status: Z

Duration: 2 instruction cycles

Code: 1010aaaa---0001--cccccccccccc

Description: Transfer the lower 24 bits of memory content at address C ($C \in \text{LIT16}$) to register A ($A \in \text{OREG}$). The upper 8 bits of the memory content at address C are transferred to MHB register.

The zero bit Z of status register STA is set, if the lower 24 bits of the transferred value are zero, otherwise the zero bit is cleared.

If the MHB register is selected as destination register A ($A \in \text{OREG}$), the bits 0 to 7 of the referred memory location are transferred to MHB.

The program counter PC is increment by the value 4.

MWR Instruction

Syntax: MWR A, C

Operation: $\text{MEM}(C[13:0])[23:0] \leftarrow A$;
 $\text{MEM}(C[13:0])[31:24] \leftarrow \text{MHB}$

Status: -

Generic Timer Module (GTM)

Duration: 2 instruction cycles

Code: 1010aaaa----0010--cccccccccccccc

Description: Transfer 24 bit value of register A ($A \in \text{OREG}$) together with the MHB register to the memory location at address C ($C \in \text{LIT16}$).

The 24 bit value of register A is stored in the lower significant bits (bit 0 to 23) of the memory location and the MHB register is stored in bits 24 to 31.

The program counter PC is increment by the value 4.

MWR24 Instruction

Syntax: MWR24 A, C

Operation: $\text{MEM}(C[13:0])[23:0] \leftarrow A$

Status: -

Duration: 3 instruction cycles

Code: 1010aaaa----0111--cccccccccccccc

Description: Transfer 24 bit value of register A ($A \in \text{OREG}$) to memory at address C ($C \in \text{LIT16}$).

The 24 bit value of register A is stored in the lower significant bits (bit 0 to 23) of the memory location and the bits 24 to 31 are left unchanged.

The program counter PC is increment by the value 4.

It should be noted that this operation is not an atomic instruction.

MRDI Instruction

Syntax: MRDI A, B

Operation: $A \leftarrow \text{MEM}(B[13:0])[23:0]$

$\text{MHB} \leftarrow \text{MEM}(B[13:0])[31:24]$

Status: Z

Duration: 2 instruction cycles

Code: 1010aaaabbbb0011-----

Description: Transfer the lower 24 bits of a memory location to register A ($A \in \text{OREG}$) using indirect addressing. The upper 8 bits of the memory location are transferred to MHB register.

The memory location where to read from is defined by the bits 0 to 15 of register B ($B \in \text{OREG}$).

The 24 bit value is received from the lower significant bits (bit 0 to 23) of the memory location.

Generic Timer Module (GTM)

The zero bit Z of status register STA is set, if the lower 24 bit of the transferred value is zero, otherwise the zero bit is cleared.

If the MHB register is selected as destination register A (A ? OREG), the bits 0 to 7 of the referred memory location are transferred to MHB.

The program counter PC is increment by the value 4.

MWRI Instruction

Syntax: MWRI A, B

Operation: MEM(B[13:0])[23:0] ← A;
MEM(B[13:0])[31:24] v MHB

Status: -

Duration: 2 instruction cycles

Code: 1010aaaabbbb0100-----

Description: Transfer 24 bit value of register A ($A \in \text{OREG}$) together with the MHB register to the memory using indirect addressing.

The memory location where to write to is defined by the bits 0 to 15 of register B ($B \in \text{OREG}$).

The 24 bit value is stored in the lower significant bits (bit 0 to 23) of the memory location and the MHB register is stored in bits 24 to 31.

The program counter PC is increment by the value 4.

MWRI24 Instruction

Syntax: MWRI24 A, B

Operation: MEM(B[13:0])[23:0] ← A;

Status: -

Duration: 3 instruction cycles

Code: 1010aaaabbbb1000-----

Description: Transfer 24 bit value of A ($A \in \text{OREG}$) to memory using indirect addressing.

The memory location where to write to is defined by the bits 0 to 15 of register B ($B \in \text{OREG}$).

The 24 bit value is stored in the lower significant bits (bit 0 to 23) of the memory location and the bits 24 to 31 are left unchanged.

The program counter PC is increment by the value 4.

It should be noted that this operation is not an atomic instruction.

POP Instruction
Syntax: POP A

Operation: $A \leftarrow \text{MEM}(\text{R7}[13:0])[23:0]$;
 $\text{MHB} \leftarrow \text{MEM}(\text{R7}[13:0])[31:24]$;
 $\text{R7} \leftarrow \text{R7} - 4$;
 $\text{SP_CNT} \leftarrow \text{SP_CNT} - 1$
Status: Z, EN

Duration: 2 instruction cycles

Code: 1010aaaa---0101-----

Description: Transfer the lower significant bits (bit 0 to 23) from the top of stack to register A ($A \in \text{OREG}$), followed by decrementing the stack pointer register R7 with the value 4. The upper 8 bits (bit 24 to 31) from the top of the stack are transferred to register MHB.

If the MHB register is selected as destination register A ($A \neq \text{OREG}$), the bits 0 to 7 from the top of the stack are transferred to MHB.

The memory location for the top of the stack is identified by the bits 0 to 15 of the stack pointer register R7.

The zero bit Z of status register STA is set, if the lower 24 bit of the transferred value is zero, otherwise the zero bit is cleared.

The program counter PC is increment by the value 4.

The SP_CNT bit field inside the MCSi_CHx_CTRL register is decremented.

If an underflow on the SP_CNT bit field occurs, the STK_ERR[i]_IRQ is raised.

If an underflow on the SP_CNT bit field occurs and the bit HLT_SP_OFLO of register MCSi_CTRL is set, the current MCS-channel is disabled by clearing the EN bit of STA.

PUSH Instruction
Syntax: PUSH A

Operation: $\text{R7} \leftarrow \text{R7} + 4$;
 $\text{SP_CNT} \leftarrow \text{SP_CNT} + 1$;
 $\text{MEM}(\text{R7}[13:0])[23:0] \leftarrow A$;
 $\text{MEM}(\text{R7}[13:0])[31:24] \leftarrow \text{MHB}$
Status: EN

Duration: 2 instruction cycles

Code: 1010aaaa---0110-----

Description: Increment the stack pointer register R7 with the value 4, followed by transferring a 24 bit value of operand A ($A \in \text{OREG}$) together with a MHB register to the

Generic Timer Module (GTM)

new top of the stack. The 24 bit value of A is stored in the bits 0 to 23 and content of the MHB register is stored in the bit 24 to 31 of the memory location.

The memory location for the top of the stack is addressed by the bits 0 to 15 of the stack pointer register.

The program counter PC is increment by the value 4.

The SP_CNT bit field inside the MCSi_CHx_CTRL register is increment.

If an overflow on the SP_CNT bit field occurs, the STK_ERR[i]_IRQ is raised.

If an overflow on the SP_CNT bit field occurs and the bit HLT_SP_OFL of register MCSi_CTRL is set, the current MCS-channel is disabled by clearing the EN bit of STA.

If an overflow on the SP_CNT bit field occurs and the bit HLT_SP_OFL of register MCSi_CTRL is set, the memory write operation for the A and MHB is discard.

26.13.2.3 ARU Instructions

ARD Instruction

Syntax: ARD A, B, C

Operation: $A \leftarrow \text{ARU}(C[8:0])[23:0];$
 $B \leftarrow \text{ARU}(C[8:0])[47:24];$
 $\text{ACB}[4:0] \leftarrow \text{ARU}(C[8:0])[52:48]$

Status: CAT

Duration: suspends current MCS-channel

Code: 1011aaaabbbb0000-----cccccccc

Description: Perform a blocking read access to the ARU and transfer both 24 bit values received at the ARU port to the registers A and B ($A \in \text{AREG}$, $B \in \text{AREG}$), whereas A holds the lower 24 bit ARU word and B holds the upper 24 bit ARU word.

If A and B refer to the same register, only the upper 24 bit ARU word is stored and the lower 24 bit ARU word is discard.

If any transferred 24 bit value from the ARU should not stored in a register, the dummy register $\text{ZERO} \in \text{AREG}$ can be selected in A or B to discard the corresponding ARU data. Actually, all address values of A and B that exceed the range 0 to 7 discard the corresponding ARU data.

The received ARU control bits are stored in the register ACB.

The lower significant bits of the literal C ($C \in \text{LIT16}$) define the ARU address where to read from.

Generic Timer Module (GTM)

At the beginning of the instruction execution the CAT bit in the register STA is always cleared. After the execution of the instruction the CAT bit is updated in order to show if the instruction finished successfully (CAT = 0) or it was cancelled by the CPU (CAT = 1). The program counter PC is increment by the value 4.

ARDI Instruction

Syntax: ARDI A, B

Operation: $A \leftarrow \text{ARU}(\text{R6}[8:0])[23:0];$
 $B \leftarrow \text{ARU}(\text{R6}[8:0])[47:24];$
 $\text{ACB}[4:0] \leftarrow \text{ARU}(\text{R6}[8:0])[52:48]$

Status: CAT

Duration: suspends current MCS-channel

Code: 1011aaaabbbb0100-----

Description: Perform a blocking read access to the ARU and transfer both 24 bit values received at the ARU port to the registers A and B ($A \in \text{AREG}$, $B \in \text{AREG}$), whereas A holds the lower 24 bit ARU word and B holds the upper 24 bit ARU word.

If A and B refer to the same register, only the upper 24 bit ARU word is stored and the lower 24 bit ARU word is discard.

If any transferred 24 bit value from the ARU should not stored in a register, the dummy register $\text{ZERO} \in \text{AREG}$ can be selected in A or B to discard the corresponding ARU data. Actually, all address values of A and B that exceed the range 0 to 7 discard the corresponding ARU data.

The received ARU control bits are stored in the register ACB.

The read address is obtained from the bits 0 to 8 of the channels register R6.

At the beginning of the instruction execution the CAT bit in the register STA is always cleared. After the execution of the instruction the CAT bit is updated in order to show if the instruction finished successfully (CAT = 0) or it was cancelled by the CPU (CAT = 1).

The program counter PC is increment by the value 4.

AWR Instruction

Syntax: AWR A, B, C

Operation: $\text{ARU}(\text{C}[4:0])[23:0] \leftarrow A;$
 $\text{ARU}(\text{C}[4:0])[47:24] \leftarrow B;$
 $\text{ARU}(\text{C}[4:0])[52:48] \leftarrow \text{ACB}[4:0];$

Status: CAT

Duration: suspends current MCS-channel

Generic Timer Module (GTM)

Code: 1011aaaabbbb0001-----cccc

Description: Perform a blocking write access to the ARU and transfer two 24 bit values to the ARU port using the registers A and B ($A \in \text{OREG}$, $B \in \text{OREG}$), whereas A holds the lower 24 bit ARU word and B holds the upper 24 bit ARU word.

The ARU control bits to be sent are taken from the register ACB.

The lower significant bits (bit 0 to bit 4) of the literal C ($C \in \text{LIT16}$) define an index into the pool of ARU write address that is used for writing data.

Each MCS submodule has a pool of several write addresses that can be shared between all MCS-channels arbitrarily.

At the beginning of the instruction execution the CAT bit in the register STA is always cleared. After the execution of the instruction the CAT bit is updated in order to show if the instruction finished successfully (CAT = 0) or it was cancelled by the CPU (CAT = 1).

The program counter PC is increment by the value 4.

AWRI Instruction

Syntax: AWRI A, B

Operation: ARU(R6[4:0])[23:0] ← A;
ARU(R6[4:0])[47:24] ← B;
ARU(R6[4:0])[52:48] ← ACB[4:0];

Status: CAT

Duration: suspends current MCS-channel

Code: 1011aaaabbbb0101-----

Description: Perform a blocking write access to the ARU and transfer two 24 bit values to the ARU port using the registers A and B ($A \in \text{OREG}$, $B \in \text{OREG}$), whereas A holds the lower 24 bit ARU word and B holds the upper 24 bit ARU word.

The ARU control bits to be sent are taken from the register ACB.

The bits 0 to 4 of the register R6 define an index into the pool of ARU write address that is used for writing data.

Each MCS submodule has a pool of several write addresses that can be shared between all MCS-channels arbitrarily.

At the beginning of the instruction execution the CAT bit in the register STA is always cleared. After the execution of the instruction the CAT bit is updated in order to show if the instruction finished successfully (CAT = 0) or it was cancelled by the CPU (CAT = 1).

The program counter PC is increment by the value 4.

NARD Instruction

Syntax: NARD A, B, C

Operation: A ? ARU(C[8:0])[23:0];
B ? ARU(C[8:0])[47:24];
ACB[4:0] ? ARU(C[8:0])[52:48]

Status: SAT

Duration: suspends current MCS-channel for a maximum of one ARU round trip cycle

Code: 1011aaaabbbb0010-----cccccccc

Description: Perform a non-blocking read access to the ARU trying to transfer both 24 bit values received at the ARU port to the registers A and B (A ? AREG, B ? AREG), whereas A holds the lower 24 bit ARU word, B holds the upper 24 bit ARU word, and the ACB register holds the received ARU control bits.

The lower significant bits of the literal C (C ? LIT16) define the ARU address where to read from.

If the transfer finished successfully, the bit SAT of the register STA is set and the transferred values are stored in the registers A, B, and ACB.

If the transfer failed due to missing data at requested source, the bit SAT of the register STA is cleared and registers A, B, and ACB are not changed.

If A and B refer to the same register, only the upper 24 bit ARU word is stored and the lower 24 bit ARU word is discard.

If any transferred 24 bit value from the ARU should not stored in a register, the dummy register ZERO ? AREG can be selected in A or B to discard the corresponding ARU data. Actually, all address values of A and B that exceed the range 0 to 7 discard the corresponding ARU data.

The program counter PC is incremented by the value 4.

NARDI Instruction

Syntax: NARDI A, B

Operation: A ? ARU(R6[8:0])[23:0];
B ? ARU(R6[8:0])[47:24];
ACB[4:0] ? ARU(R6[8:0])[52:48]

Status: SAT

Duration: suspends current MCS-channel for a maximum of one ARU round trip cycle

Code: 1011aaaabbbb0011-----

Description: Perform a non-blocking read access to the ARU trying to transfer both 24 bit values received at the ARU port to the registers A and B (A ? AREG, B ? AREG),

Generic Timer Module (GTM)

whereas A holds the lower 24 bit ARU word, B holds the upper 24 bit ARU word, and the ACB register holds the received ARU control bits.

The read address is obtained from the bits 0 to 8 of the channels register R6.

If the transfer finished successfully, the bit SAT of the register STA is set and the transferred values are stored in the registers A, B, and ACB.

If the transfer failed due to missing data at requested source, the bit SAT of the register STA is cleared and registers A, B, and ACB are not changed.

If A and B refer to the same register, only the upper 24 bit ARU word is stored and the lower 24 bit ARU word is discard.

If any transferred 24 bit value from the ARU should not stored in a register, the dummy register ZERO ? AREG can be selected in A or B to discard the corresponding ARU data. Actually, all address values of A and B that exceed the range 0 to 7 discard the corresponding ARU data.

The program counter PC is incremented by the value 4.

26.13.2.4 Arithmetic Logic Instructions

ADDL Instruction

Syntax: ADDL A, C

Operation: $A \leftarrow A + C$

Status: Z, CY, N, V

Duration: 1 instruction cycle

Code: 0010aaaaccccccccccccccccccccccccc

Description: Perform addition operation of a register A ($A \in \text{OREG}$) with a 24 bit literal value C ($C \in \text{LIT24}$) and store the result in register A.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The carry bit CY of status register STA is set, if an unsigned overflow occurred during addition, otherwise the bit is cleared. An unsigned overflow has occurred, when the result of the operation cannot be represented in the interval $[0; 2^{24}-1]$, assuming that both operands A and C are unsigned values within the interval $[0; 2^{24}-1]$.

The overflow bit V of status register STA is set, if a signed overflow/underflow occurred during addition, otherwise the bit is cleared. A signed overflow/underflow has occurred, when the result of the operation cannot be represented in the interval $[-2^{23}; 2^{23}-1]$, assuming that both operands A and C are signed values within the interval $[-2^{23}; 2^{23}-1]$.

The negative bit N of status register STA equals the most significant bit of the operation result, in order to determine if a calculated signed result is negative ($N=1$) or positive ($N=0$), assuming that no overflow/underflow occurred.

The program counter PC is increment by the value 4.

ADD Instruction

Syntax: ADD A, B

Operation: $A \leftarrow A + B$

Status: Z, CY, N, V

Duration: 1 instruction cycle

Code: 1100aaaabbbb0000-----

Description: Perform addition operation of a register A ($A \in \text{OREG}$) with an operand B ($B \in \text{OREG}$) and store the result in register A.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

Generic Timer Module (GTM)

The carry bit CY of status register STA is set, if an unsigned overflow occurred during addition, otherwise the bit is cleared. An unsigned overflow has occurred, when the result of the operation cannot be represented in the interval $[0; 2^{24}-1]$, assuming that both operands A and B are unsigned values within the interval $[0; 2^{24}-1]$.

The overflow bit V of status register STA is set, if a signed overflow occurred during addition, otherwise the bit is cleared. A signed overflow/underflow has occurred, when the result of the operation cannot be represented in the interval $[-2^{23}; 2^{23}-1]$, assuming that both operands A and B are signed values within the interval $[-2^{23}; 2^{23}-1]$.

The negative bit N of status register STA equals the most significant bit of the operation result, in order to determine if a calculated signed result is negative (N=1) or positive (N=0), assuming that no overflow/underflow occurred.

The program counter PC is increment by the value 4.

SUBL Instruction

Syntax: SUBL A, C

Operation: $A \leftarrow A - C$

Status: Z, CY, N, V

Duration: 1 instruction cycle

Code: 0011aaaaccccccccccccccccccccccccc

Description: Perform subtraction operation of a register A ($A \in \text{OREG}$) with a 24 bit literal value C ($C \in \text{LIT24}$) and store the result in register A.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The carry bit CY of status register STA is set, if an unsigned underflow occurred during subtraction, otherwise the bit is cleared. An unsigned underflow has occurred, when the result of the operation cannot be represented in the interval $[0; 2^{24}-1]$, assuming that both operands A and C are unsigned values within the interval $[0; 2^{24}-1]$.

The overflow bit V of status register STA is set, if a signed overflow/underflow occurred during subtraction, otherwise the bit is cleared. A signed overflow/underflow has occurred, when the result of the operation cannot be represented in the interval $[-2^{23}; 2^{23}-1]$, assuming that both operands A and C are signed values within the interval $[-2^{23}; 2^{23}-1]$.

The negative bit N of status register STA equals the most significant bit of the operation result, in order to determine if a calculated signed result is negative (N=1) or positive (N=0), assuming that no overflow/underflow occurred.

The program counter PC is increment by the value 4.

SUB Instruction**Syntax:** SUB A, B**Operation:** $A \leftarrow A - B$ **Status:** Z, CY, N, V**Duration:** 1 instruction cycle**Code:** 1100aaaabbbb0001-----**Description:** Perform subtraction operation of a register A ($A \in \text{OREG}$) with an operand B ($B \in \text{OREG}$) and store the result in register A.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The carry bit CY of status register STA is set, if an unsigned underflow occurred during subtraction, otherwise the bit is cleared. An unsigned underflow has occurred, when the result of the operation cannot be represented in the interval $[0; 2^{24}-1]$, assuming that both operands A and B are unsigned values within the interval $[0; 2^{24}-1]$.

The overflow bit V of status register STA is set, if a signed overflow/underflow occurred during subtraction, otherwise the bit is cleared. A signed overflow/underflow has occurred, when the result of the operation cannot be represented in the interval $[-2^{23}; 2^{23}-1]$, assuming that both operands A and B are signed values within the interval $[-2^{23}; 2^{23}-1]$.

The negative bit N of status register STA equals the most significant bit of the operation result, in order to determine if a calculated signed result is negative ($N=1$) or positive ($N=0$), assuming that no overflow/underflow occurred.

The program counter PC is increment by the value 4.

NEG Instruction**Syntax:** NEG A, B**Operation:** $A \leftarrow -B$ **Status:** Z, N, V**Duration:** 1 instruction cycle**Code:** 1100aaaabbbb0010-----**Description:** Perform negation operation (2's Complement) with an operand B ($B \in \text{OREG}$) and store the result in a register A ($A \in \text{OREG}$).

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The overflow bit V of status register STA is set, if a signed overflow/underflow occurred during subtraction, otherwise the bit is cleared. A signed overflow/underflow has occurred, when the result of the operation cannot be represented in the interval $[-2^{23}; 2^{23}-1]$.

Generic Timer Module (GTM)

1], assuming that both operands A and B are signed values within the interval $[-2^{23}, 2^{23}-1]$.

The negative bit N of status register STA equals the most significant bit of the operation result, in order to determine if a calculated signed result is negative (N=1) or positive (N=0), assuming that no overflow/underflow occurred.

The program counter PC is incremented by the value 4.

ANDL Instruction

Syntax: ANDL A, C

Operation: $A \leftarrow A \text{ AND } C$

Status: Z

Duration: 1 instruction cycle

Code: 0100aaaacccccccccccccccccccccccc

Description: Perform bitwise AND conjunction of a register A ($A \in \text{OREG}$) with a 24 bit literal value C ($C \in \text{LIT24}$) and store the result in register A.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The program counter PC is incremented by the value 4.

AND Instruction

Syntax: AND A, B

Operation: $A \leftarrow A \text{ AND } B$

Status: Z

Duration: 1 instruction cycle

Code: 1100aaaabbbb0011-----

Description: Perform bitwise AND conjunction of a register A ($A \in \text{OREG}$) with an operand B ($B \in \text{OREG}$) and store the result in register A.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The program counter PC is incremented by the value 4.

ORL Instruction

Syntax: ORL A, C

Operation: $A \leftarrow A \text{ OR } C$

Status: Z

Duration: 1 instruction cycle

Generic Timer Module (GTM)

Code: 0101aaaaccccccccccccccccccccccccc

Description: Perform bitwise OR conjunction of a register A ($A \in \text{OREG}$) with a 24 bit literal value C ($C \in \text{LIT24}$) and store the result in register A.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The program counter PC is increment by the value 4.

OR Instruction

Syntax: OR A, B

Operation: $A \leftarrow A \text{ OR } B$

Status: Z

Duration: 1 instruction cycle

Code: 1100aaaabbbb0100-----

Description: Perform bitwise OR conjunction of a register A ($A \in \text{OREG}$) with an operand B ($B \in \text{OREG}$) and store the result in register A.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The program counter PC is increment by the value 4.

XORL Instruction

Syntax: XORL A, C

Operation: $A \leftarrow A \text{ XOR } C$

Status: Z

Duration: 1 instruction cycle

Code: 0110aaaaccccccccccccccccccccccccc

Description: Perform bitwise XOR conjunction of a register A ($A \in \text{OREG}$) with a 24 bit literal value C ($C \in \text{LIT24}$) and store the result in register A.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The program counter PC is increment by the value 4.

XOR Instruction

Syntax: XOR A, B

Operation: $A \leftarrow A \text{ XOR } B$

Status: Z

Duration: 1 instruction cycle

Generic Timer Module (GTM)

Code: 1100aaaabbbb0101-----

Description: Perform bitwise XOR conjunction of a register A ($A \in \text{OREG}$) with an operand B ($B \in \text{OREG}$) and store the result in register A.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The program counter PC is increment by the value 4.

SHR Instruction

Syntax: SHR A, C

Operation: $A \leftarrow A \gg C$

Status: Z, CY

Duration: 1 instruction cycle

Code: 1100aaaa----0110-----cccc

Description: Perform right shift operation C ($C \in \text{LIT16}$) times of register A ($A \in \text{OREG}$), whereas C must be in the range [0; 24]. The most significant bits that are shifted into A are cleared.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The carry bit CY of status register STA is updated to the last LSB that is shifted out of the register.

The program counter PC is increment by the value 4._

SHL Instruction

Syntax: SHL A, C

Operation: $A \leftarrow A \ll C$

Status: Z, CY

Duration: 1 instruction cycle

Code: 1100aaaa----0111-----cccc

Description: Perform left shift operation C ($C \in \text{LIT16}$) times of register A ($A \in \text{OREG}$), whereas C must be in the range [0; 24]. The lower significant bits that are shifted into A are cleared.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The carry bit CY of status register STA is updated to the last MSB that is shifted out of the register.

The program counter PC is increment by the value 4.

26.13.2.5 Test Instructions

ATUL Instruction

Syntax: ATUL A, C

Operation: A - C

Status: Z, CY

Duration: 1 instruction cycle

Code: 0111aaaacccccccccccccccccccccccc

Description: Arithmetic Test with an unsigned operand A ($A \in \text{OREG}$) and an unsigned 24 bit literal value C ($C \in \text{LIT24}$).

The carry bit CY of status register STA is set if unsigned operand A is less than unsigned literal C.

Otherwise, the carry bit CY of status register STA is cleared if unsigned operand A is greater than or equal to unsigned literal C.

The zero bit Z of status register STA is set, if A equals to C.

Otherwise, the zero bit Z of status register STA is cleared, if A is unequal to C.

The program counter PC is increment by the value 4.

ATU Instruction

Syntax: ATU A, B

Operation: A - B

Status: Z, CY

Duration: 1 instruction cycle

Code: 1101aaaabbbb0000-----

Description: Arithmetic Test with an unsigned operand A ($A \in \text{OREG}$) and an unsigned operand B ($B \in \text{OREG}$).

The carry bit CY of status register STA is set if unsigned operand A is less than unsigned operand B.

Otherwise, the carry bit CY of status register STA is cleared if unsigned operand A is greater than or equal to unsigned operand B.

The zero bit Z of status register STA is set, if A equals to B.

Otherwise, the zero bit Z of status register STA is cleared, if A is unequal to B.

The program counter PC is increment by the value 4.

ATSL Instruction**Syntax:** ATSL A, C**Operation:** A - C**Status:** Z, CY**Duration:** 1 instruction cycle**Code:** 1000aaaaccccccccccccccccccccccccc**Description:** Arithmetic Test with a signed operand A ($A \in \text{OREG}$) and a signed 24 bit literal value C ($C \in \text{LIT24}$).

The carry bit CY of status register STA is set if signed operand A is less than signed literal C.

Otherwise, the carry bit CY of status register STA is cleared if signed operand A is greater than or equal to signed literal C.

The zero bit Z of status register STA is set, if A equals to C.

Otherwise, the zero bit Z of status register STA is cleared, if A is unequal to C.

The program counter PC is increment by the value 4.

ATS Instruction**Syntax:** ATS A, B**Operation:** A - B**Status:** Z, CY**Duration:** 1 instruction cycle**Code:** 1101aaaabbbb0001-----**Description:** Arithmetic Test with a signed operand A ($A \in \text{OREG}$) and a signed operand B ($B \in \text{OREG}$).

The carry bit CY of status register STA is set if signed operand A is less than signed operand B.

Otherwise, the carry bit CY of status register STA is cleared if signed operand A is greater than or equal to signed operand B.

The zero bit Z of status register STA is set, if A equals to B.

Otherwise, the zero bit Z of status register STA is cleared, if A is unequal to B.

The program counter PC is increment by the value 4.

BTL Instruction**Syntax:** BTL A, C**Operation:** A AND C

Generic Timer Module (GTM)**Status:** Z**Duration:** 1 instruction cycle**Code:** 1001aaaaccccccccccccccccccccccc**Description:** Bit test of an operand A ($A \in \text{OREG}$) with a 24 bit literal bit mask C ($C \in \text{LIT24}$).

The bit test is performed by applying a bitwise logical AND operation with operand A and the bit mask C without storing the result.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The program counter PC is increment by the value 4.

BT Instruction**Syntax:** BT A, B**Operation:** A AND B**Status:** Z**Duration:** 1 instruction cycle**Code:** 1101aaaabbbb0010-----**Description:** Bit test of an operand A ($A \in \text{OREG}$) with an operand B ($B \in \text{OREG}$), whereas usually one of the operands is a register holding a bit mask.

The bit test is performed by applying a bitwise logical AND operation with register A and register B without storing the result.

The zero bit Z of status register STA is set, if the calculated value is zero, otherwise the zero bit is cleared.

The program counter PC is increment by the value 4.

26.13.2.6 Control Flow Instructions**JMP Instruction****Syntax:** JMP C**Operation:** $\text{PC} \leftarrow \text{C}[13:0]$ **Status:** -**Duration:** 1 instruction cycle**Code:** 1110-----0000--cccccccccccc**Description:** Execute unconditional jump to the memory location C ($C \in \text{LIT16}$).

The program counter PC is loaded with literal C.

JBS Instruction**Syntax:** JBS A, B, C[13:0]**Operation:** $PC \leftarrow C$ if A[B] is set**Status:** -**Duration:** 1 instruction cycle**Code:** 1110aaaabbbb0001--cccccccccccc**Description:** Execute conditional jump to the memory location C ($C \in \text{LIT16}$).

The program counter PC is loaded with literal C, if the bit at position B ($B \in \text{LIT4}$) of operand A ($A \in \text{OREG}$) is set.

Otherwise, if the bit is cleared, the program counter PC is increment by the value 4.

JBC Instruction**Syntax:** JBC A, B, C[13:0]**Operation:** $PC \leftarrow C$ if A[B] is cleared**Status:** -**Duration:** 1 instruction cycle**Code:** 1110aaaabbbb0010--cccccccccccc**Description:** Execute conditional jump to the memory location C ($C \in \text{LIT16}$).

The program counter PC is loaded with literal C, if the bit at position B ($B \in \text{LIT4}$) of operand A ($A \in \text{OREG}$) is cleared.

Otherwise, if the bit is set, the program counter PC is increment by the value 4.

CALL Instruction**Syntax:** CALL C**Operation:** $R7 \leftarrow R7 + 4;$ $\text{MEM}(R7[15:0])[15:0] \leftarrow PC + 4;$ $PC \leftarrow C[13:0];$ $SP_CNT \leftarrow SP_CNT + 1$ **Status:** EN**Duration:** 2 instruction cycles**Code:** 1110-----0011--cccccccccccc**Description:** Call subprogram at memory location C ($C \in \text{LIT16}$).

The stack pointer register R7 is increment by the value 4.

The memory location for the top of the stack is identified by the bits 0 to 15 of the stack pointer register.

Generic Timer Module (GTM)

After the stack pointer is increment, the increment value of the PC is transferred to the top of the stack.

The program counter PC is loaded with literal C.

The SP_CNT bit field inside the MCSi_CHx_CTRL register is increment.

If an overflow on the SP_CNT bit field occurs, the STK_ERR[i]_IRQ is raised.

If an overflow on the SP_CNT bit field occurs and the bit HLT_SP_OFL of register MCSi_CTRL is set, the channel current MCS-channel is disabled by clearing the EN bit of STA.

If an overflow on the SP_CNT bit field occurs and the bit HLT_SP_OFL of register MCSi_CTRL is set, the memory write operation of the increment PC is discarding.

RET Instruction

Syntax: RET

Operation: PC ← MEM(R7[15:0])[15:0];

R7 ← R7 - 4;

SP_CNT ← SP_CNT - 1

Status: EN

Duration: 2 instruction cycles

Code: 1110-----0100-----

Description: Return from subprogram.

The program counter PC is loaded with current value on the top of the stack.

Finally, the stack pointer register R7 is decremented by the value 4.

The memory location for the top of the stack is identified by the bits 0 to 15 of the stack pointer register.

The SP_CNT bit field inside the MCSi_CHx_CTRL register is decremented.

If an underflow on the SP_CNT bit field occurs, the STK_ERR[i]_IRQ is raised.

If an underflow on the SP_CNT bit field occurs and the bit HLT_SP_OFL of register MCSi_CTRL is set, the channel current MCS-channel is disabled by clearing the EN bit of STA.

26.13.2.7 Other Instructions**WURM Instruction**

Syntax: WURM A B C

Operation: Wait until register match

Generic Timer Module (GTM)

Status: CWT

Duration: suspends current MCS-channel

Code: 1111aaaabbbb0000cccccccccccccccc

Description: Suspend current MCS-channel until the following register match condition occurs:

$A = (B \text{ AND } \text{MASK})$

whereas $A \in \text{OREG}$, $B \in \text{OREG}$, AND is a bitwise AND operation with bitmask MASK. The bits 16 to 23 of MASK are set to true and the bits 0 to 15 are copied from the instructions literal $C \in \text{LIT16}$.

If the match condition is true at the beginning of the instruction execution, the instruction does not suspend the channel and the program counter PC is increment by the value 4.

This instruction can be used to wait for one or more trigger events generated by other MCS-channels or the CPU. In this case register B is the trigger register STRG, A is a general purpose register holding the bits with the trigger condition to wait for and C is the bitmask that enables trigger bits of interest.

The trigger bits can be set by other MCS channels with a write access (e.g. using a MOVL instruction) to the STRG register or the CPU with a write access to the MCSi_STRG register.

The trigger bits are not cleared automatically by hardware after resuming an MCS-channel, but they have to be cleared explicitly with a write access to the register CTRG by the MCS-channel or with a write access to the register MCSi_CTRG by the CPU.

Please note that more than one channel can wait for the same trigger bit to continue.

The instruction can also be used to wait on a specific time/angle event provided by the TBU. In this case register B is the interesting TBU register TBU_TS0, TBU_TS1, or TBU_TS2, register A is a general purpose register holding the value to wait for and bitmask C should be set to 0xFFFF.

At the beginning of the instruction execution the CWT bit in the register STA is always cleared. After the execution of the instruction the CWT bit is updated in order to show if the instruction finished successfully ($\text{CWT} = 0$) or it was cancelled by the CPU ($\text{CWT} = 1$).

If the CWT bit is set simultaneously with the occurrence of the register match condition, the register match condition is prioritized resulting in a cleared CWT bit.

The program counter PC is increment by the value 4, when the trigger bit is set and the channel continues its operation.

NOP Instruction

Syntax: NOP

Operation: -

Status: -

Duration: 1 instruction cycle

Code: 0000-----

Description: No operation is performed.

The program counter PC is increment by the value 4.

26.13.3 MCS Internal Registers

This section describes MCS internal registers which are partly only accessible by the corresponding MCS-channel itself and partly global to all channels. Please see [Table 26-42](#) for clarification.

These registers can be directly accessed with the entire MCS instruction set, e.g. using the ORL instruction to set a specific bit.

26.13.3.1 MCS Internal Registers Overview

The table describes the MCS internal registers. Only parts of this register set can be accessed by the CPU:

Table 26-42 MCS Internal Registers Overview

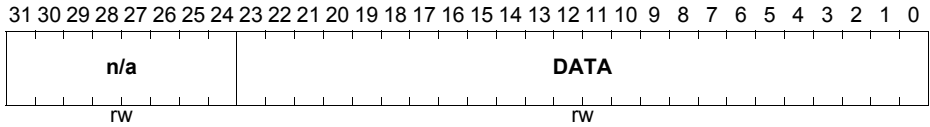
Register Name	Description	Details in Section
Rx	General purpose register x (x: 0...7), MCS Task internal register	Section 26.13.3.2
STA	Status register, MCS Task internal register	Section 26.13.3.3
ACB	ARU Control Bit register, MCS Task internal register	Section 26.13.3.4
CTRG	Clear Trigger Bits register, MCS Global register from task view	Section 26.13.3.5
STRG	Set Trigger Bits register, MCS Global register from task view	Section 26.13.3.6
TBU_TS0	TBU Timestamp TS0 register, MCS Global register from task view	Section 26.13.3.7
TBU_TS1	TBU Timestamp TS1 register, MCS Global register from task view	Section 26.13.3.8
TBU_TS2	TBU Timestamp TS2 register, MCS Global register from task view	Section 26.13.3.9
MHB	Memory High Byte register, MCS Task internal register	Section 26.13.3.10

Generic Timer Module (GTM)

26.13.3.2 General purpose register Rx (x:0...7)

Rx (x=0-7)

General Purpose Register x (0 + x_H) Reset Value: 000000_H



Field	Bits	Type	Description
DATA	[23:0]	rw	Data field of general purpose register Note: Register R7 is also used as stack pointer register, if stack operations are used in the MCS micro program. Note: Register R6 used also as index/address register for indirect ARU addressing instructions.
n/a	[31:24]	rw	Not applicable

Generic Timer Module (GTM)
26.13.3.3 Register STA

Note: If both, the CPU and the MCS-channel are writing to the same register at the clock cycle, the value of the CPU is written to the register and the value of the MCS-channel is discarding.

STA
Status Register
(8_H)
Reset Value: 00000XX_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
n/a								Reserved				SP_CNT				
r								r				r				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reserved						SAT	CWT	CAT	N	V	Z	CY	MCA	ERR	IRQ	EN
r						r	r	r	r	r	r	rh	rwh	rwh	rw	rw

Field	Bits	Type	Description
EN	0	rw	Enable current MCS-channel 0 _B Disable current MCS-channel. 1 _B Enable current MCS-channel.
IRQ	1	rw	Trigger IRQ 0 _B No triggered IRQ signal. 1 _B Trigger IRQ signal. Note: An MCS-channel trigger an IRQ by writing value 1 to bit IRQ. Writing a value 0 to this bit does not cancel the IRQ, and thus has no effect. Note: An MCS-channel can read the IRQ bit in order to determine the current state of the IRQ handling. The MCS-channel reads a value 1 if an IRQ was released but not cleared by CPU. If an MCS-channel reads a value 0 no IRQ was released or it has been cleared by CPU. Note: The IRQ bit can only be cleared by CPU, by writing a 1 to the corresponding MCSi_CHx_NOTIFY register (see Section 26.13.4.6).

Generic Timer Module (GTM)

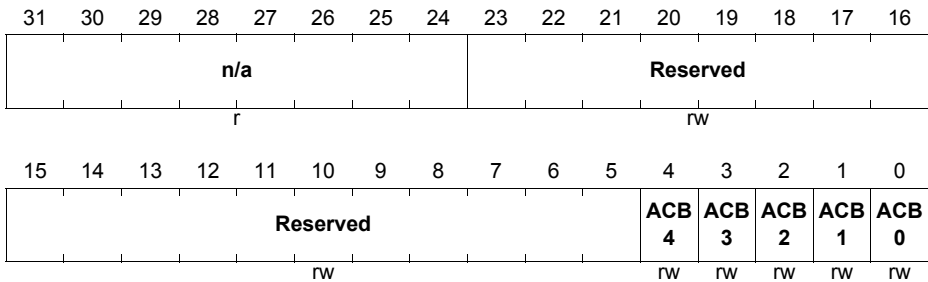
Field	Bits	Type	Description
ERR	2	rwh	<p>Set Error Signal</p> <p>0_B No Error occurred 1_B Error occurred</p> <p>Note: The ERR signal of an MCS-channel reflects Error status that may be caused by one of the following conditions:</p> <ul style="list-style-type: none"> • MCS-channel program sets the ERR signal by writing to this bit • ?ECC RAM Error occurred while accessing the connected RAM pages (also disables MCS-channel by clearing bit EN) • Decoding an instruction with an invalid opcode (also disables MCS-channel by clearing bit EN) <p>Note: If the GTM includes a MON submodule, the ERR signal is always captured by this module.</p> <p>Note: An MCS-channel releases an error signal by writing value 1 to bit ERR. Writing a value 0 to this bit does not cancel the error signal, and thus has no effect.</p> <p>Note: An MCS-channel can read the ERR bit in order to determine the current state of the error signal. The MCS-channel reads a value 1 if an ERR occurred previously, but not cleared by CPU. If an MCS-channel reads a value 0 no error was set or it has been cleared by CPU.</p> <p>Note: The ERR bit can only be cleared by CPU, by writing a 1 to the MCSi_ERR register (see Register MCSi_ERR).</p>
MCA	3	rwh	<p>MON Activity signalling for MCS channel</p> <p>0_B No activity signalled to submodule MON. 1_B Activity signalled to submodule MON.</p> <p>Note: When this bit is set the corresponding channel in the MON submodule register MON_ACTIVITY is set (see Section 26.20.8.2. This bit is automatically cleared after writing it by the MCS channel program.</p>
CY	4	rh	<p>Carry bit</p> <p>The carry bit is updated by several arithmetic and logic instructions. In arithmetic operations, the carry bit indicates an unsigned under/overflow.</p>
Z	5	r	<p>Zero bit</p> <p>The zero bit is updated by several arithmetic, logic and data transfer instructions to indicate a result of zero.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
V	6	r	Overflow bit The overflow bit is updated by arithmetic instructions in order to indicate a signed under/overflow.
N	7	r	Negative bit The negative bit is updated by arithmetic instructions in order to indicate a negative result.
CAT	8	r	Cancel ARU transfer bit 0 _B Last ARU transfer was not cancelled. 1 _B CPU cancelled last ARU transfer. <i>Note: This bit is updated after each ARU transfer and it should be evaluated immediately after the ARU instruction. Otherwise, the CPU could set the bit leading to a bad status information in the MCS program.</i>
CWT	9	r	Cancel WURM instruction bit 0 _B Last WURM instruction was not cancelled. 1 _B CPU cancelled last WURM instruction of channel. <i>Note: This bit is updated after each WURM instruction and it should be evaluated immediately after the WURM instruction. Otherwise, the CPU could set the bit leading to a bad status information in the MCS program.</i>
SAT	10	r	Successful ARU transfer bit 0 _B Non-blocking ARU transfer failed due to missing data 1 _B Non-blocking ARU transfer finished successfully
Reserved	[15:11]	r	Reserved Read as zero, should be written as zero
SP_CNT	[18:16]	r	Stack pointer counter value Actual stack depth of channel. The bit field is increment on behalf of a CALL or PUSH instruction and decremented on behalf of a RET or POP instruction. The MCS channel STK_ERR_IRQ is raised, when an overflow or underflow is detected on this bit field.
Reserved	[23:19]	r	Reserved Read as zero, should be written as zero
n/a	[31:24]	r	Not applicable

Generic Timer Module (GTM)

Note: Writing to bits of the register STA with instructions that do implicitly a read-modify-write operation (e.g. "ANDL STA, 0xFFFFFE" or "OR STA, R0") is dangerous, since writing back the possibly modified content of the read access (which reflects status information) may cause undesirable results. A secure way for writing of bits of the register STA is to use instructions that do not read the content of STA (e.g. "MOVL STA, 0x0" or "MOV STA, R1").

26.13.3.4 Register ACB
ACB
ARU Control Bit Register
(9_H)
Reset Value: 01FE00XX_H


Field	Bits	Type	Description
ACB0	0	rw	ARU Control bit 0 Note: This bit is updated by each ARU read access and its value is sent to ARU by each ARU write access on bit 48 of the ARU word.
ACB1	1	rw	ARU Control bit 1 Note: This bit is updated by each ARU read access and its value is sent to ARU by each ARU write access on bit 49 of the ARU word.
ACB2	2	rw	ARU Control bit 2 Note: This bit is updated by each ARU read access and its value is sent to ARU by each ARU write access on bit 50 of the ARU word.
ACB3	3	rw	ARU Control bit 3 Note: This bit is updated by each ARU read access and its value is sent to ARU by each ARU write access on bit 51 of the ARU word.
ACB4	4	rw	ARU Control bit 4 Note: This bit is updated by each ARU read access and its value is sent to ARU by each ARU write access on bit 52 of the ARU word.
Reserved	[23:5]	rw	Reserved Read as zero, should be written as zero
n/a	[31:24]	r	Not applicable

26.13.3.5 Register CTRG
CTRG
Clear Trigger Bits Register
(A_H)
Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
n/a								Reserved							
r								rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRG 15	TRG 14	TRG 13	TRG 12	TRG 11	TRG 10	TRG 9	TRG 8	TRG 7	TRG 6	TRG 5	TRG 4	TRG 3	TRG 2	TRG 1	TRG 0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

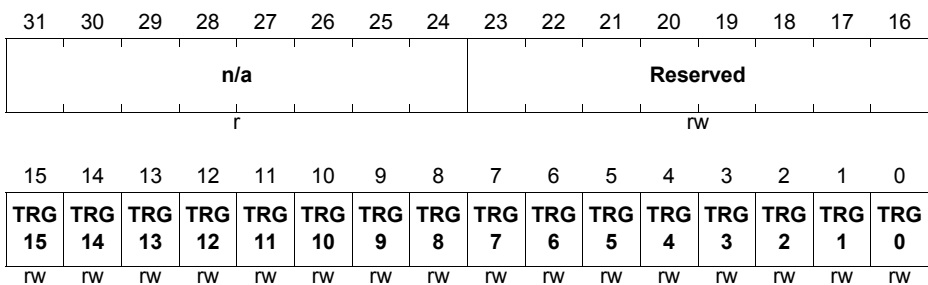
Field	Bits	Type	Description
TRG0	0	rw	Trigger bit 0 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG1	1	rw	Trigger bit 1 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG2	2	rw	Trigger bit 2 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG3	3	rw	Trigger bit 3 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG4	4	rw	Trigger bit 4 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG5	5	rw	Trigger bit 5 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG6	6	rw	Trigger bit 6 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG7	7	rw	Trigger bit 7 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG8	8	rw	Trigger bit 8 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG9	9	rw	Trigger bit 9 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG10	10	rw	Trigger bit 10 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG11	11	rw	Trigger bit 11 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG12	12	rw	Trigger bit 12 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG13	13	rw	Trigger bit 13 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG14	14	rw	Trigger bit 14 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG15	15	rw	Trigger bit 15 0_B READ: trigger bit is cleared / WRITE: do nothing 1_B READ: trigger bit is set / WRITE: clear trigger bit Note: The trigger bits TRGx (x = 0...15) are accessible by all MCS channels as well as the CPU. Setting a trigger bit can be performed with the STRG register, in the case of an MCS-channel or the MCSi_STRG register in the case of the CPU. Clearing a trigger bit can be performed with the CTRG register, in the case of an MCS-channel or the MCSi_CTRG register in the case of the CPU. Trigger bits can be used for signaling specific events to MCS-channels or the CPU. An MCS-channel suspended with a WURM instruction can be resumed by setting the appropriate trigger bit.
Reserved	[23:16]	rw	Reserved Read as zero, should be written as zero
n/a	[31:24]	r	Not applicable

26.13.3.6 Register STRG
STRG
Set Trigger Bits Register
(B_H)
Reset Value: 00000000_H


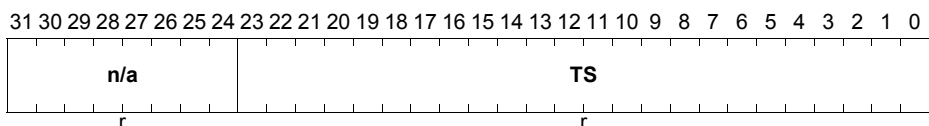
Field	Bits	Type	Description
TRG0	0	rw	Trigger bit 0 0_B READ: trigger bit is cleared / WRITE: do nothing 1_B READ: trigger bit is set / WRITE: set trigger bit

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG1	1	rw	Trigger bit 1 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG2	2	rw	Trigger bit 2 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG3	3	rw	Trigger bit 3 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG4	4	rw	Trigger bit 4 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG5	5	rw	Trigger bit 5 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG6	6	rw	Trigger bit 6 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG7	7	rw	Trigger bit 7 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG8	8	rw	Trigger bit 8 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG9	9	rw	Trigger bit 9 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG10	10	rw	Trigger bit 10 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG11	11	rw	Trigger bit 11 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG12	12	rw	Trigger bit 12 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit

Generic Timer Module (GTM)

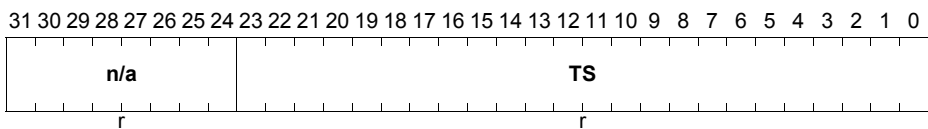
Field	Bits	Type	Description
TRG13	13	rw	Trigger bit 13 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG14	14	rw	Trigger bit 14 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG15	15	rw	Trigger bit 15 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit Note: The trigger bits TRGx (x = 0...15) are accessible by all MCS channels as well as the CPU. Setting a trigger bit can be performed with the STRG register, in the case of an MCS-channel or the MCSi_STRG register in the case of the CPU. Clearing a trigger bit can be performed with the CTRG register, in the case of an MCS-channel or the MCSi_CTRG register in the case of the CPU. Trigger bits can be used for signaling specific events to MCS-channels or the CPU. An MCS-channel suspended with a WURM instruction can be resumed by setting the appropriate trigger bit.
Reserved	[23:16]	rw	Reserved Read as zero, should be written as zero
n/a	[31:24]	r	Not applicable

26.13.3.7 Register TBU_TS0
TBU_TS0
TBU Timestamp TS0 Register
(C_H)
Reset Value: XXXXXXXX_H


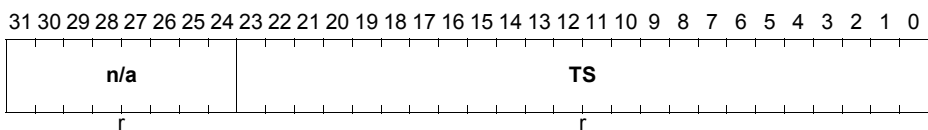
Field	Bits	Type	Description
TS	[23:0]	r	Current TBU time stamp 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
n/a	[31:24]	r	Not applicable Note: Any write access to a time base register discards the written data. A write access to a time base register may be used to destroy an unused 24-bit data word of an ARU read transfer.

26.13.3.8 Register TBU_TS1
TBU_TS1
TBU Timestamp TS1 Register
(D_H)
Reset Value: XXXXXXXX_H


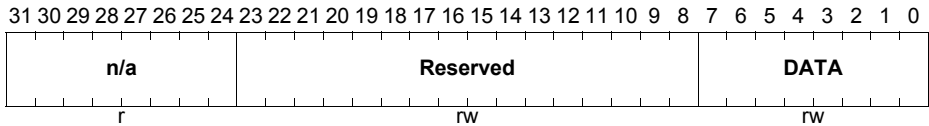
Field	Bits	Type	Description
TS	[23:0]	r	Current TBU time stamp 1
n/a	[31:24]	r	Not applicable Note: Any write access to a time base register discards the written data. A write access to a time base register may be used to destroy an unused 24-bit data word of an ARU read transfer.

26.13.3.9 Register TBU_TS2
TBU_TS2
TBU Timestamp TS2 Register
(E_H)
Reset Value: XXXXXXXX_H


Field	Bits	Type	Description
TS	[23:0]	r	Current TBU time stamp 2

Generic Timer Module (GTM)

Field	Bits	Type	Description
n/a	[31:24]	r	Not applicable Note: Any write access to a time base register discards the written data. A write access to a time base register may be used to destroy an unused 24-bit data word of an ARU read transfer.

26.13.3.10 Register MHB
MHB
Memory High Byte Register
(F_H)
Reset Value: XX000000_H


Field	Bits	Type	Description
DATA	[7:0]	rw	High Byte of a memory transfer
Reserved	[23:8]	rw	Reserved Read as zero, should be written as zero
n/a	[31:24]	r	Not applicable

26.13.4 MCS Configuration Registers

This section describes the configuration registers of the MCS submodule.

These registers can only be accessed by the CPU using AEI, but not within the MCS-channel using MCS instructions.

26.13.4.1 MCS Configuration Registers Overview

The table describes the MCS registers that are visible and accessible by the CPU:

Table 26-43 MCS Configuration Registers Overview

Register Name	Description	Details in Section
MCSi_CHx_CTRL	MCS Channel control register (x: 0...7)	Section 26.13.4.2
MCSi_CHx_PC	MCS Channel Program counter register (x: 0...7)	Section 26.13.4.3
MCSi_CHx_Ry	MCS Channel GPRz registers (x: 0...7; y: 0...7)	Section 26.13.4.4
MCSi_CHx_ACB	MCS Channel ACB register (x: 0...7)	Section 26.13.4.5
MCSi_CHx_IRQ_NOTIFY	MCS Channel x interrupt notification register (x: 0...7)	Section 26.13.4.6

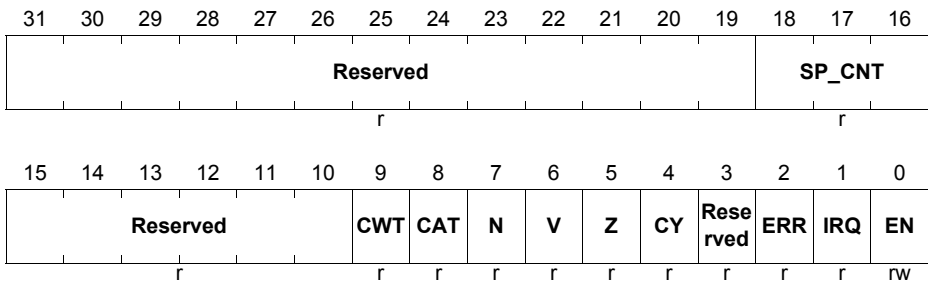
Generic Timer Module (GTM)
Table 26-43 MCS Configuration Registers Overview (cont'd)

Register Name	Description	Details in Section
MCSi_CHx_IRQ_EN	MCS Channel x interrupt enable register (x: 0...7)	Section 26.13.4.7
MCSi_CHx_IRQ_FORCINT	MCS Channel x software interrupt generation register (x: 0...7)	Section 26.13.4.8
MCSi_CHx_IRQ_MODE	IRQ mode configuration register (x=0...7)	Section 26.13.4.9
MCSi_CHx_EIRQ_EN	MCS Channel x error interrupt enable register (x=0...7)	Section 26.13.4.10
MCSi_CTRL	MCS Control register	Section 26.13.4.11
MCSi_CTRG	MCS Clear trigger control register	Section 26.13.4.12
MCSi_STRG	MCS Set trigger control register	Section 26.13.4.13
MCSi_RST	MCS Channel reset register	Section 26.13.4.14
MCSi_ERR	MCS Error register	Section 26.13.4.15

Generic Timer Module (GTM)

26.13.4.2 Register MCSi_CHx_CTRL (x:07)

All of the following registers are 32-bit only accessible.

GTM_MCS0_CHx_CTRL (x=0-7)
MCS0 Channel Control Register (30020_H+x*80_H)
Reset Value: 00000000_H
GTM_MCS1_CHx_CTRL (x=0-7)
MCS1 Channel Control Register (31020_H+x*80_H)
Reset Value: 00000000_H
GTM_MCS2_CHx_CTRL (x=0-7)
MCS2 Channel Control Register (32020_H+x*80_H)
Reset Value: 00000000_H


Field	Bits	Type	Description
EN	0	rw	<p>Enable MCS-channel</p> <p>0_B Disable current MCS-channel. 1_B Enable current MCS-channel.</p> <p>Note: Enabling or disabling of an MCS-channel is synchronized to the ending of an instruction and thus it may take several clock cycles, e.g. active memory transfers or pending WURM transfers have to be finished before disabling the MCS-channel. The internal state of a channel can be obtained by reading the bit EN.</p> <p>Note: In order to disable an MCS channel reliably the EN bit should be cleared following by setting the CAT and CWT bit in order to cancel any pending WURM or ARU instructions.</p> <p>Note: The EN bit is write protected during RAM reset phase.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
IRQ	1	r	Interrupt state 0_B No interrupt pending in MCS-channel x. 1_B Interrupt is pending in MCS-channel x. Note: This bit is read only and it mirrors the internal IRQ state.
ERR	2	r	Error state 0_B No error signal pending in MCS-channel x. 1_B Error signal is pending in MCS-channel x. Note: This bit is read only and it mirrors the internal error state.
Reserved	3	r	Reserved Read as zero, should be written as zero
CY	4	r	Carry bit state Note: This bit is read only and it mirrors the internal carry flag CY.
Z	5	r	Zero bit state Note: This bit is read only and it mirrors the internal zero flag Z.
V	6	r	Overflow bit state Note: This bit is read only and it mirrors the internal carry flag V.
N	7	r	Negative bit state Note: This bit is read only and it mirrors the internal zero flag N.
CAT	8	r	Cancel ARU transfer state Note: This bit is read only and it mirrors the internal cancel ARU transfer status flag CAT.
CWT	9	r	Cancel WURM instruction state Note: This bit is read only and it mirrors the internal cancel WURM instruction status flag CWT.
Reserved	[15:10]	r	Reserved Read as zero, should be written as zero
SP_CNT	[18:16]	r	Stack pointer counter value Actual stack depth of channel. The bit field is increment on behalf of a CALL or PUSH instruction and decremented on behalf of a RET or POP instruction. The MCS channel STK_ERR_IRQ is raised, when an overflow or underflow is detected on this bit field.

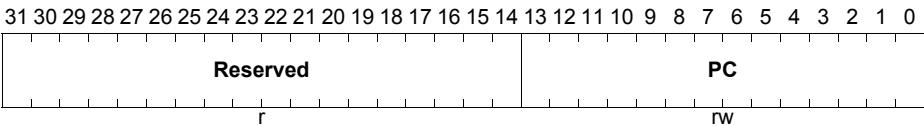
Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:19]	r	Reserved Read as zero, should be written as zero

26.13.4.3 Register MCSi_CHx_PC (x:0...7)

All of the following registers are 32-bit only accessible.

GTM_MCSi_CH0_PC (i=0-2) MCSi Channel 0 Program Counter Register (30040 _H +i*1000 _H)	Reset Value: 00000000_H
GTM_MCSi_CH1_PC (i=0-2) MCSi Channel 1 Program Counter Register (300C0 _H +i*1000 _H)	Reset Value: 00000004_H
GTM_MCSi_CH2_PC (i=0-2) MCSi Channel 2 Program Counter Register (30140 _H +i*1000 _H)	Reset Value: 00000008_H
GTM_MCSi_CH3_PC (i=0-2) MCSi Channel 3 Program Counter Register (301C0 _H +i*1000 _H)	Reset Value: 0000000C_H
GTM_MCSi_CH4_PC (i=0-2) MCSi Channel 4 Program Counter Register (30240 _H +i*1000 _H)	Reset Value: 00000010_H
GTM_MCSi_CH5_PC (i=0-2) MCSi Channel 5 Program Counter Register (302C0 _H +i*1000 _H)	Reset Value: 00000014_H
GTM_MCSi_CH6_PC (i=0-2) MCSi Channel 6 Program Counter Register (30340 _H +i*1000 _H)	Reset Value: 00000018_H
GTM_MCSi_CH7_PC (i=0-2) MCSi Channel 7 Program Counter Register (303C0 _H +i*1000 _H)	Reset Value: 0000001C_H



Field	Bits	Type	Description
PC	[13:0]	rw	Current Program Counter Note: The program counter is only writable if the corresponding MCS-channel is disabled. The bits 0 and 1 are always written as zeros.
Reserved	[31:14]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.13.4.4 Register MCSi_CHx_Ry (x:0...7, y:0...7)

All of the following registers are 32-bit only accessible.

GTM_MCS0_CH0_Ry (y=0-7)

MCS0 Channel 0 Program Counter Register y

(30000_H+y*04_H)

Reset Value: 00000000_H

GTM_MCS1_CH0_Ry (y=0-7)

MCS1 Channel 0 Program Counter Register y

(31000_H+y*04_H)

Reset Value: 00000000_H

GTM_MCS2_CH0_Ry (y=0-7)

MCS2 Channel 0 Program Counter Register y

(32000_H+y*04_H)

Reset Value: 00000000_H

GTM_MCS0_CH1_Ry (y=0-7)

MCS0 Channel 1 Program Counter Register y

(30080_H+y*04_H)

Reset Value: 00000000_H

GTM_MCS1_CH1_Ry (y=0-7)

MCS1 Channel 1 Program Counter Register y

(31080_H+y*04_H)

Reset Value: 00000000_H

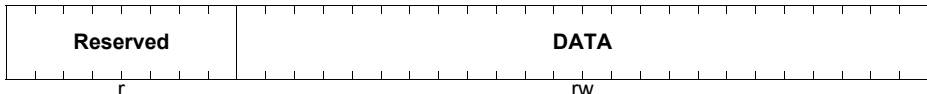
GTM_MCS2_CH1_Ry (y=0-7)

MCS2 Channel 1 Program Counter Register y

(32080_H+y*04_H)

Reset Value: 00000000_H

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

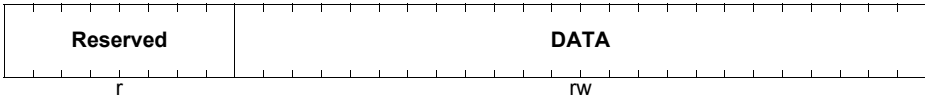


Field	Bits	Type	Description
DATA	[23:0]	rw	Data of MCS general purpose register ry
Reserved	[31:24]	r	Read as zero, should be written as zero Note: This register is the same as described in Section 26.13.2 .

Generic Timer Module (GTM)

GTM_MCS0_CH2_Ry (y=0-7)		
MCS0 Channel 2 Program Counter Register y	(30100_H+y*04_H)	Reset Value: 00000000_H
GTM_MCS1_CH2_Ry (y=0-7)		
MCS1 Channel 2 Program Counter Register y	(31100_H+y*04_H)	Reset Value: 00000000_H
GTM_MCS2_CH2_Ry (y=0-7)		
MCS2 Channel 2 Program Counter Register y	(32100_H+y*04_H)	Reset Value: 00000000_H
GTM_MCS0_CH3_Ry (y=0-7)		
MCS0 Channel 3 Program Counter Register y	(30180_H+y*04_H)	Reset Value: 00000000_H
GTM_MCS1_CH3_Ry (y=0-7)		
MCS1 Channel 3 Program Counter Register y	(31180_H+y*04_H)	Reset Value: 00000000_H
GTM_MCS2_CH3_Ry (y=0-7)		
MCS2 Channel 3 Program Counter Register y	(32180_H+y*04_H)	Reset Value: 00000000_H

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

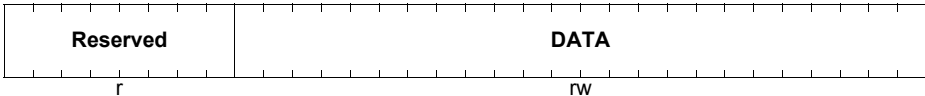


Field	Bits	Type	Description
DATA	[23:0]	rw	Data of MCS general purpose register ry
Reserved	[31:24]	r	Read as zero, should be written as zero Note: This register is the same as described in Section 26.13.2 .

Generic Timer Module (GTM)

GTM_MCS0_CH4_Ry (y=0-7)		
MCS0 Channel 4 Program Counter Register y	(30200_H+y*04_H)	Reset Value: 00000000_H
GTM_MCS1_CH4_Ry (y=0-7)		
MCS1 Channel 4 Program Counter Register y	(31200_H+y*04_H)	Reset Value: 00000000_H
GTM_MCS2_CH4_Ry (y=0-7)		
MCS2 Channel 4 Program Counter Register y	(32200_H+y*04_H)	Reset Value: 00000000_H
GTM_MCS0_CH5_Ry (y=0-7)		
MCS0 Channel 5 Program Counter Register y	(30280_H+y*04_H)	Reset Value: 00000000_H
GTM_MCS1_CH5_Ry (y=0-7)		
MCS1 Channel 5 Program Counter Register y	(31280_H+y*04_H)	Reset Value: 00000000_H
GTM_MCS2_CH5_Ry (y=0-7)		
MCS2 Channel 5 Program Counter Register y	(32280_H+y*04_H)	Reset Value: 00000000_H

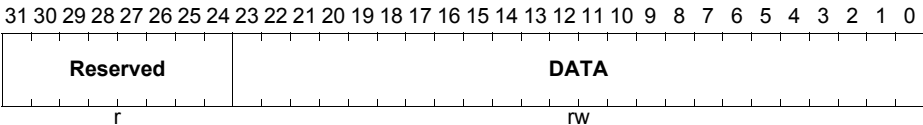
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0



Field	Bits	Type	Description
DATA	[23:0]	rw	Data of MCS general purpose register ry
Reserved	[31:24]	r	Read as zero, should be written as zero Note: This register is the same as described in Section 26.13.2 .

Generic Timer Module (GTM)

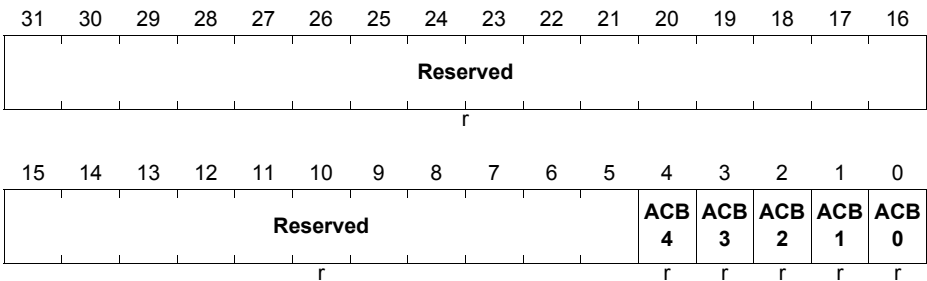
- GTM_MCS0_CH6_Ry (y=0-7)**
MCS0 Channel 6 Program Counter Register y
 (30300_H+y*04_H) **Reset Value: 00000000_H**
- GTM_MCS1_CH6_Ry (y=0-7)**
MCS1 Channel 6 Program Counter Register y
 (31300_H+y*04_H) **Reset Value: 00000000_H**
- GTM_MCS2_CH6_Ry (y=0-7)**
MCS2 Channel 6 Program Counter Register y
 (32300_H+y*04_H) **Reset Value: 00000000_H**
- GTM_MCS0_CH7_Ry (y=0-7)**
MCS0 Channel 7 Program Counter Register y
 (30380_H+y*04_H) **Reset Value: 00000000_H**
- GTM_MCS1_CH7_Ry (y=0-7)**
MCS1 Channel 7 Program Counter Register y
 (31380_H+y*04_H) **Reset Value: 00000000_H**
- GTM_MCS2_CH7_Ry (y=0-7)**
MCS2 Channel 7 Program Counter Register y
 (32380_H+y*04_H) **Reset Value: 00000000_H**



Field	Bits	Type	Description
DATA	[23:0]	rw	Data of MCS general purpose register ry
Reserved	[31:24]	r	Read as zero, should be written as zero Note: This register is the same as described in Section 26.13.2 .

26.13.4.5 Register MCSi_CHx_ACB (x:0...7)

All of the following registers are 32-bit only accessible.

GTM_MCS0_CHx_ACB (x=0-7)
MCS0 Channel ACB Register (30024_H+x*80_H) Reset Value: 00000000_H
GTM_MCS1_CHx_ACB (x=0-7)
MCS1 Channel ACB Register (31024_H+x*80_H) Reset Value: 00000000_H
GTM_MCS2_CHx_ACB (x=0-7)
MCS2 Channel ACB Register (32024_H+x*80_H) Reset Value: 00000000_H


Field	Bits	Type	Description
ACB0	0	r	ARU Control bit 0 Note: This bit is read only and it mirrors the internal state.
ACB1	1	r	ARU Control bit 1 Note: This bit is read only and it mirrors the internal state.
ACB2	2	r	ARU Control bit 2 Note: This bit is read only and it mirrors the internal state.
ACB3	3	r	ARU Control bit 3 Note: This bit is read only and it mirrors the internal state.
ACB4	4	r	ARU Control bit 4 Note: This bit is read only and it mirrors the internal state.
Reserved	[31:5]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.13.4.6 Register MCSi_CHx_IRQ_NOTIFY (x:0...7)

All of the following registers are 32-bit only accessible.

GTM_MCS0_CHx_IRQ_NOTIFY (x=0-7)

MCS0 Channel x interrupt notification register

$$(30044_H + x * 80_H)$$

Reset Value: 00000000_H

GTM_MCS1_CHx_IRQ_NOTIFY (x=0-7)

MCS1 Channel x interrupt notification register

$$(31044_H + x * 80_H)$$

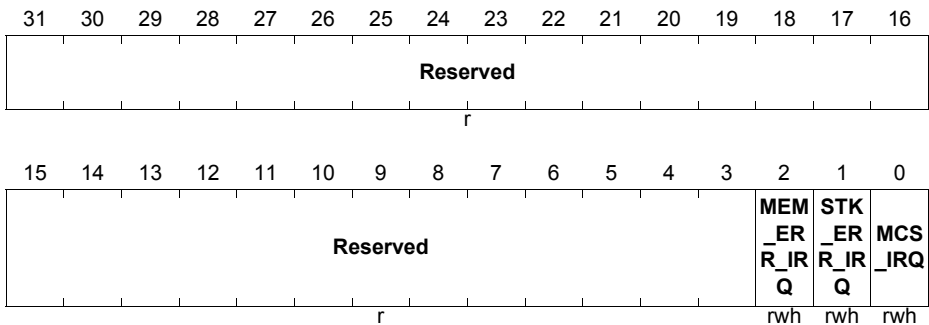
Reset Value: 00000000_H

GTM_MCS2_CHx_IRQ_NOTIFY (x=0-7)

MCS2 Channel x interrupt notification register

$$(32044_H + x * 80_H)$$

Reset Value: 00000000_H



Field	Bits	Type	Description
MCS_IRQ	0	rwh	<p>Interrupt request by MCS-channel x</p> <p>0_B No IRQ released 1_B IRQ released by MCS-channel</p> <p>Note: This bit will be cleared on a CPU write access with a value '1'. A read access leaves the bit unchanged. Note: By writing a '1' to this register, the IRQ flag in the MCS channel status register STA is cleared.</p>
STK_ERR_IRQ	1	rwh	<p>Stack counter overflow/underflow of channel x</p> <p>0_B No IRQ released 1_B A stack counter overflow or underflow occurred</p> <p>Note: This bit will be cleared on a CPU write access with a value '1'. A read access leaves the bit unchanged.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
MEM_ERR_IRQ	2	rwh	<p>Memory access out of range in channel x</p> <p>0_B No IRQ released 1_B MCS-channel request a memory location out of range 0 to MP1-4.</p> <p>Note: This bit will be cleared on a CPU write access with a value '1'. A read access leaves the bit unchanged. Note: In the case of a memory overflow, any read or write access to the RAM is always blocked. The read data is unpredictable. Note: It should be noted that in case of a memory overflow, the MSC channel will continue. Thus it is recommended that the CPU immediately stops the MSC channel.</p>
Reserved	[31:3]	r	<p>Reserved Read as zero, should be written as zero</p>

26.13.4.7 Register MCSi_CHx_IRQ_EN (x:0...7)

All of the following registers are 32-bit only accessible.

GTM_MCS0_CHx_IRQ_EN (x=0-7)

MCS0 Channel x Interrupt Enable Register

(30048_H+x*80_H)

Reset Value: 00000000_H

GTM_MCS1_CHx_IRQ_EN (x=0-7)

MCS1 Channel x Interrupt Enable Register

(31048_H+x*80_H)

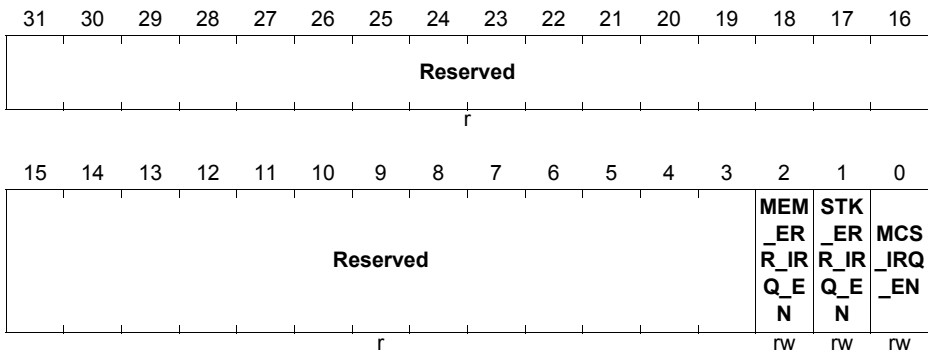
Reset Value: 00000000_H

GTM_MCS2_CHx_IRQ_EN (x=0-7)

MCS2 Channel x Interrupt Enable Register

(32048_H+x*80_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
MCS_IRQ_EN	0	rw	MCS channel x MCS_IRQ interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
STK_ERR_IRQ_EN	1	rw	MCS channel x STK_ERR_IRQ interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
MEM_ERR_IRQ_EN	2	rw	MCS channel x MEM_ERR_IRQ interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM 1 _B Enable interrupt, interrupt is visible outside GTM
Reserved	[31:3]	r	Reserved Read as zero, should be written as zero

26.13.4.8 Register MCSi_CHx_IRQ_FORCINT (x:0...7)

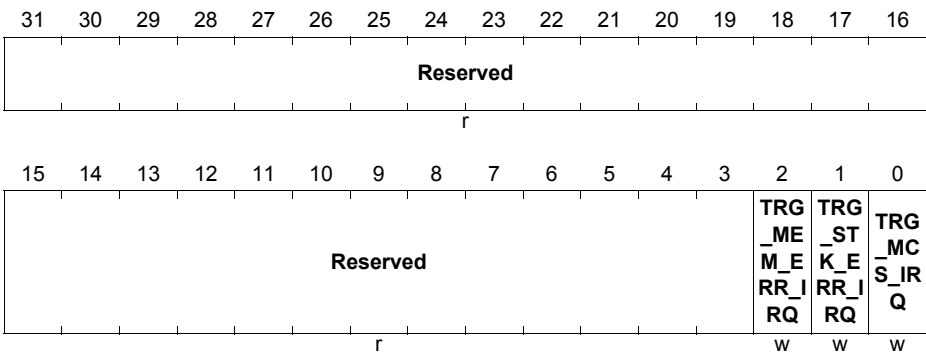
All of the following registers are 32-bit accessible.

Generic Timer Module (GTM)

GTM_MCS0_CHx_IRQ_FORCINT (x=0-7)
 MCS0 Channel x Software Interrupt Generation Register
 (3004C_H+x*80_H) Reset Value: 00000000_H

GTM_MCS1_CHx_IRQ_FORCINT (x=0-7)
 MCS1 Channel x Software Interrupt Generation Register
 (3104C_H+x*80_H) Reset Value: 00000000_H

GTM_MCS2_CHx_IRQ_FORCINT (x=0-7)
 MCS2 Channel x Software Interrupt Generation Register
 (3204C_H+x*80_H) Reset Value: 00000000_H



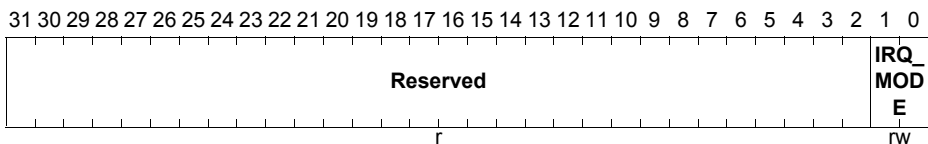
Field	Bits	Type	Description
TRG_MCS_IRQ	0	w	Trigger IRQ bit in MCS_CH_x_IRQ_NOTIFY register by software 0 _B No interrupt triggering 1 _B Assert corresponding field in MCSi_CHx_IRQ_NOTIFY register Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
TRG_STK_ERR_IRQ	1	w	Trigger IRQ bit in MCS_CH_x_IRQ_NOTIFY register by software 0 _B No interrupt triggering 1 _B Assert corresponding field in MCSi_CHx_IRQ_NOTIFY register Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG_MEM_ERR_IRQ	2	w	Trigger IRQ bit in MCS_CH_x_IRQ_NOTIFY register by software 0 _B No interrupt triggering 1 _B Assert corresponding field in MCSi_CHx_IRQ_NOTIFY register Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
Reserved	[31:3]	r	Reserved Read as zero, should be written as zero

26.13.4.9 Register MCSi_CHx_IRQ_MODE (x:0...7)

All of the following registers are 32-bit only accessible.

GTM_MCS0_CHx_IRQ_MODE (x=0-7)
MCS0 IRQ Mode Configuration Register
 $(30050_H + x * 80_H)$
Reset Value: 00000000_H
GTM_MCS1_CHx_IRQ_MODE (x=0-7)
MCS1 IRQ Mode Configuration Register
 $(31050_H + x * 80_H)$
Reset Value: 00000000_H
GTM_MCS2_CHx_IRQ_MODE (x=0-7)
MCS2 IRQ Mode Configuration Register
 $(32050_H + x * 80_H)$
Reset Value: 00000000_H


Field	Bits	Type	Description
IRQ_MODE	[1:0]	rw	IRQ mode selection 00 _B Level mode 01 _B Pulse mode 10 _B Pulse-Notify mode 11 _B Single-Pulse mode Note: The interrupt modes are described in Section 26.2.5 .

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.13.4.10 Register MCSi_CHx_EIRQ_EN (x:0...7)

GTM_MCS0_CHx_EIRQ_EN (x=0-7)

 MCS0_Channel x Error Interrupt Enable Register
 (30054_H+x*80_H)

 Reset Value: 00000000_H

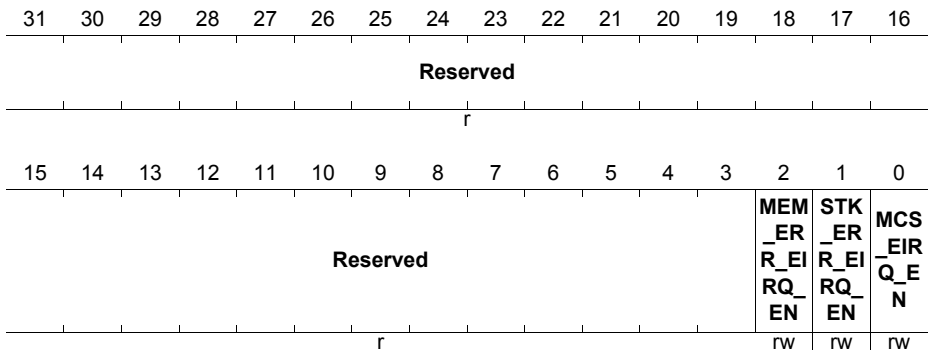
GTM_MCS1_CHx_EIRQ_EN (x=0-7)

 MCS1_Channel x Error Interrupt Enable Register
 (31054_H+x*80_H)

 Reset Value: 00000000_H

GTM_MCS2_CHx_EIRQ_EN (x=0-7)

 MCS2_Channel x Error Interrupt Enable Register
 (32054_H+x*80_H)

 Reset Value: 00000000_H


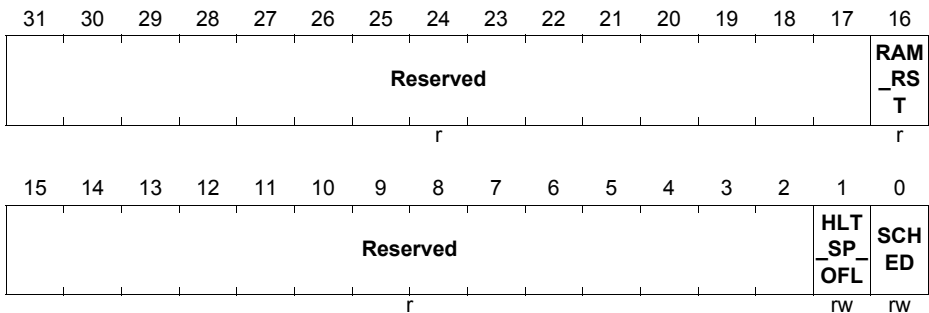
Field	Bits	Type	Description
MCS_EIRQ_EN	0	rw	MCS channel x MCS_EIRQ error interrupt enable 0 _B Disable error interrupt 1 _B Enable error interrupt
STK_ERR_IRQ_EN	1	rw	MCS channel x STK_ERR_IRQ error interrupt enable 0 _B Disable error interrupt 1 _B Enable error interrupt
MEM_ERR_IRQ_EN	2	rw	MCS channel x MEM_ERR_IRQ error interrupt enable 0 _B Disable error interrupt 1 _B Enable error interrupt
Reserved	[31:3]	r	Reserved Read as zero, should be written as zero

26.13.4.11 Register MCSi_CTRL

All of the following registers are 32-bit only accessible.

GTM_MCSi_CTRL (i=0-2)

MCSi Control Register (30074_H+i*1000_H) **Reset Value: 00010000_H**



Field	Bits	Type	Description
SCHED	0	rw	MCS submodule scheduling scheme 0 _B Accelerated scheduling scheme. 1 _B Round-Robin scheduling scheme.
HLT_SP_OFL	1	rw	Halt on stack pointer overflow 0 _B No halt on MCS-channel stack pointer counter over/underflow. 1 _B MCS-channel is disabled if a stack pointer counter over/underflow occurs.
Reserved	[15:2]	r	Reserved Read as zero, should be written as zero
RAM_RST	16	r	RAM reset bit 0 _B READ: no RAM reset is active / WRITE: do nothing. 1 _B READ: MCS currently resets RAM content / WRITE: trigger RAM reset. Note: The RAM reset initializes the memory content with zeros. RAM access and enabling of MCS channels is disabled during active RAM reset. Note: This bit is only writable if the bit RF_PROT in register GTM_CTRL is cleared and all MCS-channels are disabled.

Generic Timer Module (GTM)

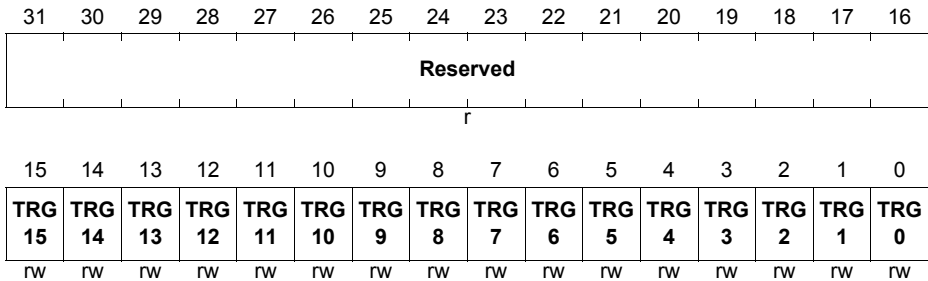
Field	Bits	Type	Description
Reserved	[31:17]	r	Reserved Read as zero, should be written as zero

26.13.4.12 Register MCSi_CTRG

All of the following registers are 32-bit only accessible.

GTM_MCSi_CTRG (i=0-2)

MCSi Clear Trigger Control Register(30028_H+i*1000_H) **Reset Value: 00000000_H**



Field	Bits	Type	Description
TRG0	0	rw	Trigger bit 0 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG1	1	rw	Trigger bit 1 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG2	2	rw	Trigger bit 2 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG3	3	rw	Trigger bit 3 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG4	4	rw	Trigger bit 4 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG5	5	rw	Trigger bit 5 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG6	6	rw	Trigger bit 6 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG7	7	rw	Trigger bit 7 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG8	8	rw	Trigger bit 8 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG9	9	rw	Trigger bit 9 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG10	10	rw	Trigger bit 10 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG11	11	rw	Trigger bit 11 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG12	12	rw	Trigger bit 12 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG13	13	rw	Trigger bit 13 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit
TRG14	14	rw	Trigger bit 14 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: clear trigger bit

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG15	15	rw	Trigger bit 15 0_B READ: trigger bit is cleared / WRITE: do nothing 1_B READ: trigger bit is set / WRITE: clear trigger bit Note: The trigger bits TRGx (x = 0...15) are accessible by all MCS channels as well as the CPU. Setting a trigger bit can be performed with the STRG register, in the case of an MCS-channel or the MCSi_STRG register in the case of the CPU. Clearing a trigger bit can be performed with the CTRG register, in the case of an MCS-channel or the MCSi_CTRG register in the case of the CPU. Trigger bits can be used for signaling specific events to MCS-channels or the CPU. An MCS-channel suspended with a WURM instruction can be resumed by setting the appropriate trigger bit.
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

Note: A write access to MCSi_CTRG may take up to 9 clock cycles, since the write access is scheduled to the next CPU time slot determined by the MCS scheduler.

26.13.4.13 Register MCSi_STRG

All of the following registers are 32-bit only accessible.

GTM_MCSi_STRG (i=0-2)

MCSi Set Trigger Control Register(3002C_H+i*1000_H) **Reset Value: 00000000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRG 15	TRG 14	TRG 13	TRG 12	TRG 11	TRG 10	TRG 9	TRG 8	TRG 7	TRG 6	TRG 5	TRG 4	TRG 3	TRG 2	TRG 1	TRG 0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG0	0	rw	Trigger bit 0 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG1	1	rw	Trigger bit 1 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG2	2	rw	Trigger bit 2 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG3	3	rw	Trigger bit 3 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG4	4	rw	Trigger bit 4 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG5	5	rw	Trigger bit 5 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG6	6	rw	Trigger bit 6 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG7	7	rw	Trigger bit 7 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG8	8	rw	trigger bit 8 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG9	9	rw	Trigger bit 9 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG10	10	rw	Trigger bit 10 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG11	11	rw	Trigger bit 11 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG12	12	rw	Trigger bit 12 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG13	13	rw	Trigger bit 13 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG14	14	rw	Trigger bit 14 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit
TRG15	15	rw	Trigger bit 15 0 _B READ: trigger bit is cleared / WRITE: do nothing 1 _B READ: trigger bit is set / WRITE: set trigger bit Note: The trigger bits TRGx (x = 0...15) are accessible by all MCS channels as well as the CPU. Setting a trigger bit can be performed with the STRG register, in the case of an MCS-channel or the MCSi_STRG register in the case of the CPU. Clearing a trigger bit can be performed with the CTRG register, in the case of an MCS-channel or the MCSi_CTRG register in the case of the CPU. Trigger bits can be used for signaling specific events to MCS-channels or the CPU. An MCS-channel suspended with a WURM instruction can be resumed by setting the appropriate trigger bit.
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

Note: A write access to MCSi_STRG may take up to 9 clock cycles, since the write access is scheduled to the next CPU time slot determined by the MCS scheduler.

26.13.4.14 Register MCSi_RST

All of the following registers are 32-bit only accessible.

GTM_MCSi_RST (i=0-2)

MCSi Channel Reset Register ($30078_H + i * 1000_H$) **Reset Value: 00000000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Reserved									CWT	CWT	CWT	CWT	CWT	CWT	CWT	
									7	6	5	4	3	2	1	0
r									rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
CAT	CAT	CAT	CAT	CAT	CAT	CAT	CAT	RST	RST	RST	RST	RST	RST	RST	RST	
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	w	w	w	w	w	w	w	w	

Field	Bits	Type	Description
RST0	0	w	Software reset of channel 0 0 _B No action 1 _B Reset channel
RST1	1	w	Software reset of channel 1 0 _B No action 1 _B Reset channel
RST2	2	w	Software reset of channel 2 0 _B No action 1 _B Reset channel
RST3	3	w	Software reset of channel 3 0 _B No action 1 _B Reset channel
RST4	4	w	Software reset of channel 4 0 _B No action 1 _B Reset channel
RST5	5	w	Software reset of channel 5 0 _B No action 1 _B Reset channel
RST6	6	w	Software reset of channel 6 0 _B No action 1 _B Reset channel

Generic Timer Module (GTM)

Field	Bits	Type	Description
RST7	7	w	Software reset of channel 7 0 _B No action 1 _B Reset channel Note: The RSTx (x = 0 ...7) bits is cleared automatically after write access of CPU. All channel related registers are set to their reset values and channel operation is stopped immediately. The reset action RSTx has a higher priority than setting the bits CWTx or CATx (x = 0 7). Note: Channel related registers are all registers MCSi_CHx_*, all MCS internal registers accessible by the corresponding channel, with exception of the common trigger register (accessed by CTRG/STRG).
CAT0	8	rwh	Cancel ARU transfer for channel 0 0 _B Do nothing. 1 _B Cancel any pending ARU read or write transfer.
CAT1	9	rwh	Cancel ARU transfer for channel 1 0 _B Do nothing. 1 _B Cancel any pending ARU read or write transfer.
CAT2	10	rwh	Cancel ARU transfer for channel 2 0 _B Do nothing. 1 _B Cancel any pending ARU read or write transfer.
CAT3	11	rwh	Cancel ARU transfer for channel 3 0 _B Do nothing. 1 _B Cancel any pending ARU read or write transfer.
CAT4	12	rwh	Cancel ARU transfer for channel 4 0 _B Do nothing. 1 _B Cancel any pending ARU read or write transfer.
CAT5	13	rwh	Cancel ARU transfer for channel 5 0 _B Do nothing. 1 _B Cancel any pending ARU read or write transfer.
CAT6	14	rwh	Cancel ARU transfer for channel 6 0 _B Do nothing. 1 _B Cancel any pending ARU read or write transfer.

Generic Timer Module (GTM)

Field	Bits	Type	Description
CAT7	15	rwh	<p>Cancel ARU transfer for channel 7</p> <p>0_B Do nothing. 1_B Cancel any pending ARU read or write transfer. This bit is cleared if no read or write is performed actually or with the next read or write after the currently performed read or write is cancelled.</p> <p><i>Note: The CATx (x = 0 ...7) bit inside the STA register of the corresponding MCS-channel is set and any pending ARU read or write request is cancelled. The MCS-channel resumes with the instruction after the ARU transfer instruction.</i></p> <p><i>Note: Note: The CATx (x = 0 ...7) bit is cleared by the corresponding MCS channel, when the channel reaches an ARU read or write instruction.</i></p>
CWT0	16	rwh	<p>Cancel WURM instruction for channel 0</p> <p>0_B Do nothing. 1_B Cancel any pending WURM instruction.</p>
CWT1	17	rwh	<p>Cancel WURM instruction for channel 1</p> <p>0_B Do nothing. 1_B Cancel any pending WURM instruction.</p>
CWT2	18	rwh	<p>Cancel WURM instruction for channel 2</p> <p>0_B Do nothing. 1_B Cancel any pending WURM instruction.</p>
CWT3	19	rwh	<p>Cancel WURM instruction for channel 3</p> <p>0_B Do nothing. 1_B Cancel any pending WURM instruction.</p>
CWT4	20	rwh	<p>Cancel WURM instruction for channel 4</p> <p>0_B Do nothing. 1_B Cancel any pending WURM instruction.</p>
CWT5	21	rwh	<p>Cancel WURM instruction for channel 5</p> <p>0_B Do nothing. 1_B Cancel any pending WURM instruction.</p>
CWT6	22	rwh	<p>Cancel WURM instruction for channel 6</p> <p>0_B Do nothing. 1_B Cancel any pending WURM instruction.</p>

Generic Timer Module (GTM)

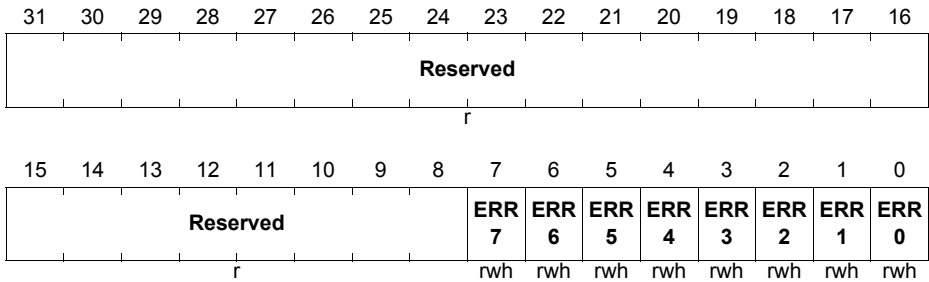
Field	Bits	Type	Description
CWT7	23	rwh	<p>Cancel WURM instruction for channel 7</p> <p>0_B Do nothing. 1_B Cancel any pending WURM instruction. This bit is cleared if no WURM is performed actually or with the next WURM after the currently performed WURM is cancelled.</p> <p><i>Note:</i> The CWT_x (x = 0 ...7) bit inside the STA register of the corresponding MCS-channel is set and any pending WURM instruction is cancelled. The MCS-channel resumes with the instruction after the WURM instruction.</p> <p><i>Note:</i> <i>Note:</i> The CWT_x (x = 0 7) bit is cleared by the corresponding MCS channel, when the channel reaches a WURM instruction.</p>
Reserved	[31:24]	r	<p>Reserved Read as zero, should be written as zero</p>

26.13.4.15 Register MCSi_ERR

All of the following registers are 32-bit only accessible.

GTM_MCSi_ERR (i=0-2)

MCSi Error Register (3007C_H+i*1000_H) Reset Value: 00000000_H



Field	Bits	Type	Description
ERR0	0	rwh	Error State of MCS-channel 0 0 _B No error signal. 1 _B Error signal is pending.
ERR1	1	rwh	Error State of MCS-channel 1 0 _B No error signal. 1 _B Error signal is pending.
ERR2	2	rwh	Error State of MCS-channel 2 0 _B No error signal. 1 _B Error signal is pending.
ERR3	3	rwh	Error State of MCS-channel 3 0 _B No error signal. 1 _B Error signal is pending.
ERR4	4	rwh	Error State of MCS-channel 4 0 _B No error signal. 1 _B Error signal is pending.
ERR5	5	rwh	Error State of MCS-channel 5 0 _B No error signal. 1 _B Error signal is pending.
ERR6	6	rwh	Error State of MCS-channel 6 0 _B No error signal. 1 _B Error signal is pending.

Generic Timer Module (GTM)

Field	Bits	Type	Description
ERR7	7	rwh	Error State of MCS-channel 7 0 _B No error signal. 1 _B Error signal is pending. Note: The CPU can read the ERRx (x = 07) bits in order to determine the current error state of the corresponding MCS-channel x. The error state is also evaluated by the module MON, if this module is available. Note: Writing a value 1 to this bit resets the corresponding error state and resets the channel internal ERR bit in the STA and channel CTRL registers.
Reserved	[31:8]	r	Reserved Read as zero, should be written as zero

26.14 Memory Configuration (MCFG)

26.14.1 Overview

The Memory Configuration submodule (MCFG) is an infrastructure module that organizes physical memory blocks and maps them to the instances of Multi Channel Sequencer (MCS) submodules.

The default configuration maps a memory of size $1K \times 32 \text{ bit} = 4\text{KB}$ to MCS memory page 0 and a memory of size $0.5K \times 32 \text{ bit} = 2 \text{KB}$ to MCS memory page 1.

In order to support different memory size for different MCS instances, the MCFG module provides two additional layout configurations for reorganization of memory pages between neighboring MCS modules. [Figure 26-63](#) shows all layout configurations.

The layout configurations SWAP is swapping the 2KB memory page of the current MCS instance with the 4KB memory page of the successive MCS instance. Thus the memory of the current MCS module is increased by 2KB but the memory of the successor is decreased by 2KB.

The layout configurations BORROW is borrowing the 4KB memory page of the successive MCS instance for the current instance. Thus the memory of the current MCS module is increased by 4KB but the memory of the successor is decreased by 4KB.

It should be noted that the successor of the last MCS instance $MCS[i]$ is the first MCS instance $MCS0$.

The actual size of the memory pages for an MCS instance depends on the layout configuration for the current instance $MCS[i]$ and the layout configuration of the preceding memory instance $MCS[i-1]$.

[Figure 26-62](#) summarizes the layout parameters $MP0$ and $MP1$ for MCS instance $MCS[i]$.

The addressing of memory page 0 ranges from 0 to $MP0-[i]$ and the addressing of memory page 1 ranges from $MP0$ to $MP1-[i]$.

Besides these software related layout configurations, the MCFG submodule has an additional input port $MCS_RAM1_EN_ADDR_MSB$ which is routed to the top level of the GTM.

The above mentioned behaviour of the MCFG submodule is applied if this port is connected to a constant logic level of zero (0).

If a constant logic level of one (1) is applied to this port, the MCFG submodule assumes that a memory of size $1K \times 32 \text{ bit} = 4\text{KB}$ is also mapped to MCS memory page 1.

In this case the memory layout configurations of the MCFG submodule change as shown in [Figure 26-64](#) and the memory layout parameters are modified according to [Figure 26-65](#).

**26.14.1.1 Memory Layout Configurations
(MSC_RAM1_EN_ADDR_MSB=0)**

	DEFAULT	SWAP	BORROW
Configuration for instance MCS[j]	<div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">4KB</div> <div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">2KB</div>	<div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">4KB</div> <div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">4KB</div>	<div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">4KB</div> <div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">4KB</div> <div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">2KB</div>
Configuration for instance MCS[j+1]	<div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">4KB</div> <div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">2KB</div>	<div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">2KB</div> <div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">2KB</div>	<div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 0 auto;">2KB</div> <p style="text-align: right; margin-top: 10px;">MCFG_2</p>

Figure 26-62 Memory Layout Configurations (MSC_RAM1_EN_ADDR_MSB=0)

26.14.1.2 Memory layout Parameters (MSC_RAM1_EN_ADDR_MSB=0)

			Memory Layout Option of preceding MCS instance MCS[i-1]		
			DEFAULT	SWAP	BORROW
Memory Layout Option of current MCS instance MCS[i]	DEFAULT	MP0	0x1000	0x800	0x0
		MP1	0x1800	0x1000	0x800
	SWAP	MP0	0x1000	0x800	0x0
		MP1	0x2000	0x1800	0x1000
	BORROW	MP0	0x1000	0x800	0x0
		MP1	0x2800	0x2000	0x1800

MCFG_1

Figure 26-63 Memory layout Parameters (MSC_RAM1_EN_ADDR_MSB=0)

**26.14.1.3 Memory Layout Configurations
(MSC_RAM1_EN_ADDR_MSB=1)**

	DEFAULT	SWAP	BORROW
Configuration for instance MCS[i]	<div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div> <div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div>	<div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div> <div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div>	<div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div> <div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div> <div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div>
Configuration for instance MCS[i+1]	<div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div> <div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div>	<div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div> <div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div>	<div style="border: 1px solid black; padding: 5px; width: 60px; margin: 5px auto;">4KB</div>

MCFG_3

Figure 26-64 Memory Layout Configurations (MSC_RAM1_EN_ADDR_MSB=1)

26.14.1.4 Memory Layout Parameters (MSC_RAM1_EN_ADDR_MSB=1)

		Memory Layout Option of preceding MCS instance MCS[i-1]			
		DEFAULT	SWAP	BORROW	
Memory Layout Option of current MCS instance MCS[i]	DEFAULT	MP0	0x1000	0x1000	0x0
		MP1	0x2000	0x2000	0x1000
	SWAP	MP0	0x1000	0x1000	0x0
		MP1	0x2000	0x2000	0x1000
	BORROW	MP0	0x1000	0x1000	0x0
		MP1	0x3000	0x3000	0x2000

MCFG_4

Figure 26-65 Memory Layout Parameters (MSC_RAM1_EN_ADDR_MSB=1)

26.14.2 MCFG Configuration Registers

This section describes the configuration registers of the MCFG submodule.

Table 26-44 MCFG Configuration Registers

Register Name	Description	Details in Section
MCFG_CTRL	Memory layout configuration register	Section 26.14.2.1

26.14.2.1 Register MCFG_CTRL

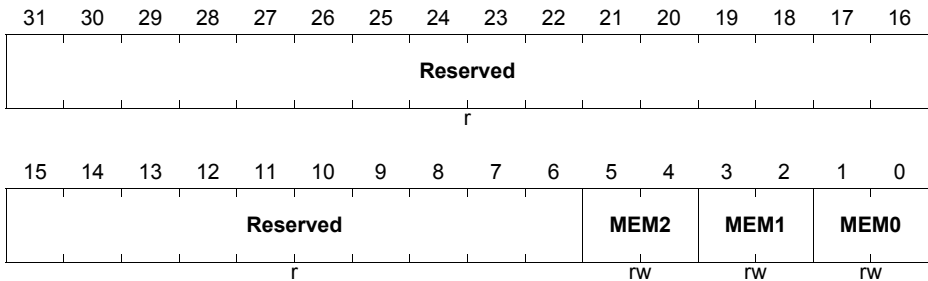
The following register is 32-bit only accessible.

GTM_MCFG_CTRL

Memory Layout Configuration Register

(F40_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
MEM0	[1:0]	rw	Configure Memory pages for MCS-instance MCS0 00 _B DEFAULT configuration 01 _B SWAP configuration 10 _B BORROW configuration 11 _B DEFAULT configuration
MEM1	[3:2]	rw	Configure Memory pages for MCS-instance MCS1 00 _B DEFAULT configuration 01 _B SWAP configuration 10 _B BORROW configuration 11 _B DEFAULT configuration
MEM2	[5:4]	rw	Configure Memory pages for MCS-instance MCS2 00 _B DEFAULT configuration 01 _B SWAP configuration 10 _B BORROW configuration 11 _B DEFAULT configuration
Reserved	[31:6]	r	Reserved Read as zero, should be written as zero

26.15 TIM0 Input Mapping Module (MAP)

26.15.1 Overview

The MAP submodule generates the two input signals TRIGGER and STATE for the submodule DPLL by evaluating the output signals of the channel 0 up to channel 5 of submodule TIM0. By using the TIM as input submodule, the filtering of the input signals can be done inside the TIM channels themselves. The MAP submodule architecture is depicted in [Figure 26-66](#).

26.15.1.1 MAP Submodule architecture

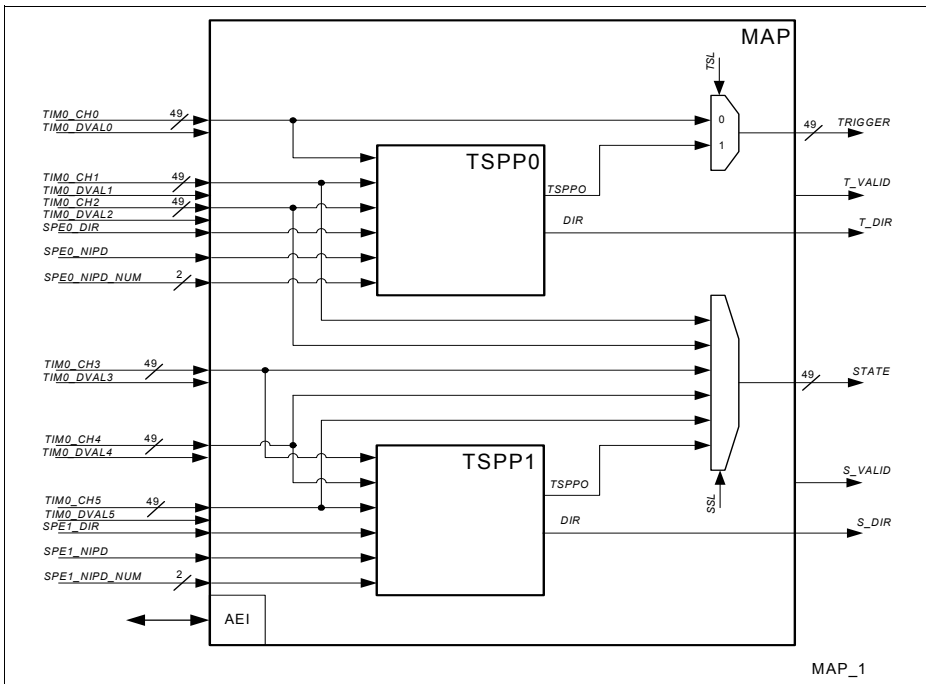


Figure 26-66 MAP Submodule architecture

Generally, the MAP submodule can route the channel signals coming from TIM0 in three ways. First, it is possible to route the whole 49 bits of data coming from channel 0 of

Generic Timer Module (GTM)

module TIM0 (TIM0_CH0) to the TRIGGER signal which is then provided to the DPLL together with the T_VALID signal.

Second, the MAP module can route one of the five signals coming from the module TIM0 (i.e. the signals coming from channel 1 up to channel 5) to the output signal STATE which is then provided to the module DPLL together with the S_VALID signal.

Third, the TRIGGER, T_VALID, STATE and S_VALID signals can be generated out of the TIM Signal Preprocessing (TSPP) subunits. This is done in combination with the Sensor Pattern Evaluation (SPE) submodule described in [Section 26.17](#).

There, the signal TRIGGER is generated in subunit TSPP0 out of the TIM0 signals coming from channel 0 up to 2.

The signal STATE is generated in subunit TSPP1 out of the TIM signals coming from channel 3 up to channel 5.

This is only be done, when the TSSPx subunits are enabled and when the SPEx_NIPD signal is raised by the SPE submodule. The SPEx_NIPD_NUM signal encodes, which of the 3 TIMx_CHy input signals has been changed. The SPEx_DIR signal is routed through the TSSPx subunit and implements the T_DIR or S_DIR signal.

A third method to provide a direction signal to DPLL is to use TIM0 channel 6 input (TIM0_IN6) and to route it instead of the DIR signal coming from TSSOP0 to the MAP output T_DIR (set TSEL=0)

26.15.2 TIM Signal Preprocessing (TSPP)

The TSPP combines the three 49 bit input streams coming from the TIM0 submodule and generates one combined 49 bit output stream TSPPO. The input stream combination is done in the unit Bit Stream Combination (BSC). The architecture of the TSPP is shown in [Figure 26-67](#).

26.15.2.1 TIM Signal Preprocessing (TSPP) subunit architecture

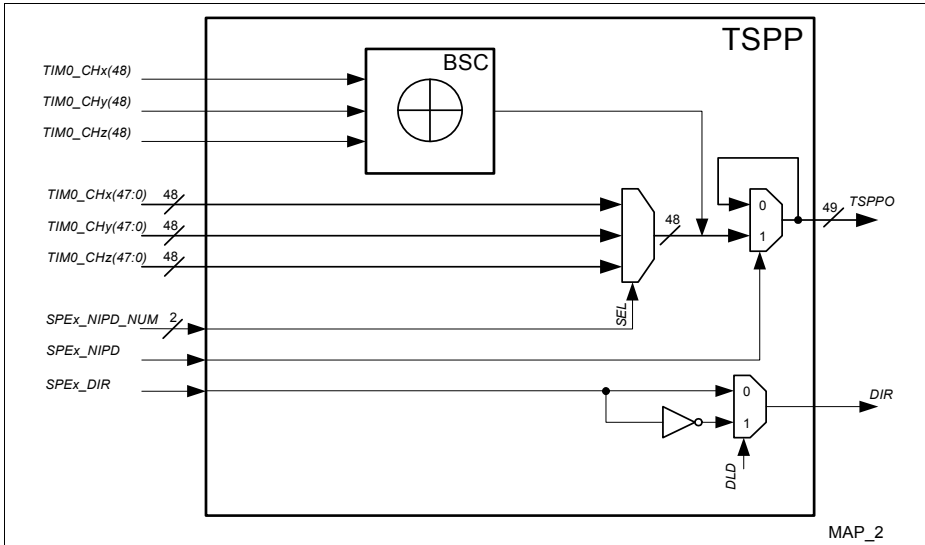


Figure 26-67 TIM Signal Preprocessing (TSPP) subunit architecture

26.15.2.2 Bit Stream Combination

The BSC subunit is used to xor-combine the three most significant bits TIM0_CHx(48), TIM0_CHy(48) and TIM0_CHz(48) of the TIM0 inputs. The xor-combined signal is merged with the remaining 48 bits of one of the three input signals TIM0_CHx(47...0), TIM0_CHy(47...0) or TIM0_CHz(47...0) the TSPPO signal. The selection is done with the SPEx_NIPD_NUM input signal coming from the SPE submodule. The action, when the 49 bits are transferred to the TSPPO and the T_VALID or S_VALID signal is raised is determined by the SPEx_NIPD signal coming from the SPE submodule. The TSPPO output signal generation is shown in the example in [Chapter 26.17.3](#).

TSPPO Signal generation for signal TSPPO

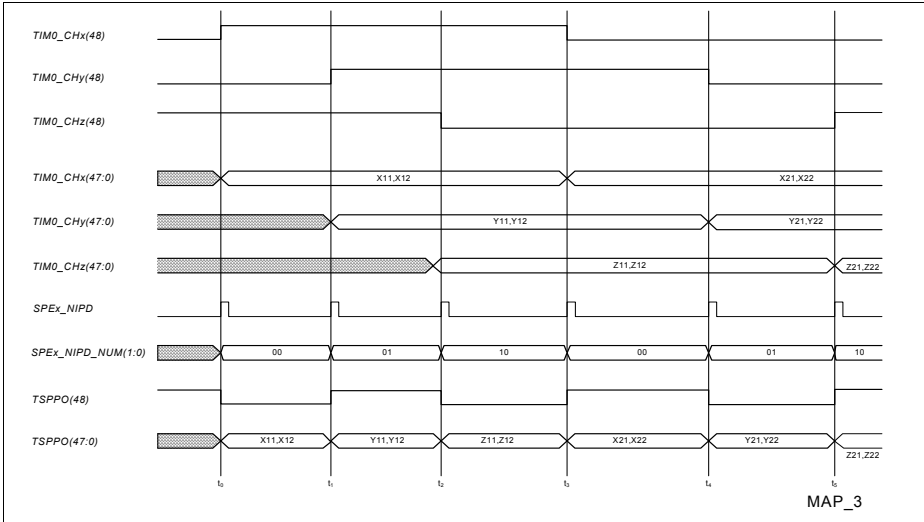


Figure 26-68 TSPPO Signal generation for signal TSPPO

The SPEX_NIPD_NUM input signal determines, which data is routed to the TSPPO signal. At the first edge of TIM0_CHx(48) the new data X11 and X12 are routed to TSPPO(47:0). The values X11 and X12 are the two 24 bit values coming from the TIM input channel TIM0_CHx. The next edge is at time t1 on signal TIM0_CHy(48). Therefore, at time t1 the TSPPO(48) signal level changes and the TSPPO(47:0) is set to Y11 and Y12 and so forth.

26.15.3 MAP Register overview

The following table gives an overview about the MAP registers

Table 26-45 MAP Register overview

Register Name	Description	Details in Section
MAP_CTRL	MAP Control register	Section 26.15.4.1

26.15.4 MAP Register description

26.15.4.1 Register MAP_CTRL

GTM_MAP_CTRL

MAP Control Register

 (F00_H)

 Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	TSP P1_I 2V	TSP P1_I 1V	TSP P1_I 0V	Reserved		TSP P1_ DLD	TSP P1_ N	Reserved	TSP P0_I 2V	TSP P0_I 1V	TSP P0_I 0V	Reserved		TSP P0_ DLD	TSP P0_ E N
r	rw	rw	rw	r		rw	rw	r	rw	rw	rw	r		rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											LSE L	SSL		TSE L	
r											rw	rw		rw	

Field	Bits	Type	Description
TSEL	0	rw	TRIGGER signal output select. 0 _B TIM0_CH0 selected as TRIGGER output signal. TIM0_IN6 (TIM0 channel 6 input) is used as direction signal T_DIR. 1 _B TSPP0_TSPPO selected as TRIGGER output signal.
SSL	[3:1]	rw	STATE signal output select. 000 _B TIM0_CH1 selected as STATE output signal. 001 _B TIM0_CH2 selected as STATE output signal. 010 _B TIM0_CH3 selected as STATE output signal. 011 _B TIM0_CH4 selected as STATE output signal. 100 _B TIM0_CH5 selected as STATE output signal. 101 _B TSPP1_TSPPO selected as STATE output signal. 110 _B same as '000' 111 _B same as '000'
LSEL	4	rw	TIM0_IN6 input level selection 0 _B TIM0_IN6 input level '0' encodes TRIGGER in forward direction 1 _B TIM0_IN6 input level '1' encodes TRIGGER in forward direction

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[15:5]	r	Reserved Read as zero, should be written as zero
TSPP0_EN	16	rw	Enable of TSPP0 subunit 0 _B TSPP0 disabled. 1 _B TSPP0 enabled.
TSPP0_DLD	17	rw	DIR level definition bit 0 _B SPEX_DIR signal is routed through as is. 1 _B SPEX_DIR signal is inverted.
Reserved	[19:18]	r	Reserved Read as zero, should be written as zero
TSPP0_I0V	20	rw	Disable of TSPP0 TIM0_CHx(48) input line. 0 _B Input line enabled. 1 _B Input line disabled; input for TSPP0 is set to zero (0).
TSPP0_I1V	21	rw	Disable of TSPP0 TIM0_CHy(48) input line. 0 _B Input line enabled. 1 _B Input line disabled; input for TSPP0 is set to zero (0).
TSPP0_I2V	22	rw	Disable of TSPP0 TIM0_CHz(48) input line. 0 _B Input line enabled. 1 _B Input line disabled; input for TSPP0 is set to zero (0).
Reserved	23	r	Reserved Read as zero, should be written as zero
TSPP1_EN	24	rw	Enable of TSPP1 subunit 0 _B TSPP1 disabled. 1 _B TSPP1 enabled.
TSPP1_DLD	25	rw	DIR level definition bit 0 _B SPEX_DIR signal is routed through as is. 1 _B SPEX_DIR signal is inverted.
Reserved	[27:26]	r	Reserved Read as zero, should be written as zero
TSPP1_I0V	28	rw	Disable of TSPP1 TIM0_CHx(48) input line 0 _B Input line enabled. 1 _B Input line disabled; input for TSPP1 is set to zero (0).

Generic Timer Module (GTM)

Field	Bits	Type	Description
TSPP1_I1 V	29	rw	Disable of TSPP1 TIM0_CHy(48) input line 0 _B Input line enabled. 1 _B Input line disabled; input for TSPP1 is set to zero (0).
TSPP1_I2 V	30	rw	Disable of TSPP1 TIM0_CHz(48) input line. 0 _B Input line enabled. 1 _B Input line disabled; input for TSPP1 is set to zero (0).
Reserved	31	r	Reserved Read as zero, should be written as zero

26.16 Digital PLL Module (DPLL)

26.16.1 Overview

The digital PLL (DPLL) submodule is used for frequency multiplication. The purpose of this module is to get a higher precision of position or value information also in the case of applications with rapidly changed input frequencies. There are two input signals TRIGGER and STATE for which periodic events are processed. The time period between two valid events is called an increment. Each increment is divided into a given number of subincrements by pulses called SUB_INC. The resolution of the generated pulses is restricted by the period of the CMU_CLK0 clock or the TS_CLK respectively (see description of the modules TBU, CMU). The input signals TRIGGER and STATE can have the meaning of position information of linear or angle motions, mass flow values, temperature, pressure or level of liquids.

By means of the DPLL the load of the CPU can be reduced essentially by relieving it from repeated or periodic standard tasks.

The DPLL has to perform the following tasks:

- prediction of the duration of the current increment in [Section 26.16.8](#)
- synchronization of the actual position (under CPU control, see [Synchronization description](#))
- possibility of seamless switch to emergency mode and back under CPU control, see configuration register DPLL_CTRL_0 at [Section 26.16.11](#)
- prediction of position and time related events in [Section 26.16.7](#)

26.16.2 Requirements and demarcation

The two input signals TRIGGER and STATE can be sensor signals from the same device or from two independent devices. When they come from the same device the TRIGGER input is typically a more frequent signal and STATE is a less frequent signal. In such a case the STATE signal can support an emergency mode, when no TRIGGER signal is available. There are also applications supported when STATE and TRIGGER are independent signals from different devices. Both input signals are combined with a validation signal T_VALID or S_VALID respectively, which shows the appearance of new data and must result in a data fetch and a start of the correspondent state machine to perform the calculations (see explanation below).

When STATE is a redundant signal of the same device only the TRIGGER input is used to generate the SUB_INC1 pulses in normal mode. There is a configuration possible, called emergency mode, for which the SUB_INC1 pulses are generated using the STATE input signal.

The decision to switch in the emergency mode and back is made outside the DPLL. The CPU must switch the configuration bit RMO (reference mode) in the DPLL_CTRL_0

Generic Timer Module (GTM)

register (see [Section 26.16.11](#)). Because a switch in emergency mode can appear suddenly, the information of the last increment durations of the STATE input up to FULL_SCALE should be stored always as a precaution.

The filtering as well as the combination or choice of the input signals is made in the TIM submodule (see [Section 26.10](#)) by use of a configurable filter algorithm for each slope and signal as well as in the MAP module (see [Section 26.15](#)) the right TRIGGER or STATE signal is selected by a multiplexer or in the SPE module (see [Section 26.18](#)) different signals are combined to a TRIGGER or STATE signal by using an antivalence operation.

The filter delay value of the signal is transmitted from the TIM module in the FT part of the corresponding signal, because the delay conditions of the signals can change during application.

The filter delays depend also on the filter algorithms used. Only the effective filter delay can be considered in the DPLL.

In order to provide the timing conditions to the DPLL the input trigger signals should have a time stamp (and optional in addition a filter value and a signal level value, as stated above) with an appropriate resolution. The resolution of the time stamps can be either the same resolution as the input time base TBU_TS0 (see [Section 26.16.4.1](#)) or 8 times higher, selected by configuration bits in the DPLL_CTRL_1 register (see [Section 26.16.11.2](#)). The time base TBU_TS0 is used to predict events in the future, called actions.

At the SUB_INCx outputs a predefined number of pulses between each active slope of the TRIGGER/STATE signal is generated, when the correspondent pulse generator is enabled by the enable bits SGEx=1 in the DPLL_CTRL_1 register (see [Section 26.16.11.2](#)).

Dependent on configuration different strategies can be used to correct a wrong pulse number.

The FULL_SCALE range is divided into a fix number of nominal increments. Nominal increments do have the same size. The number of nominal increments in HALF_SCALE is specified in the DPLL_CTRL_0 register (see [Section 26.16.11](#)).

For synchronization purposes some TRIGGER/STATE input signals can be suppressed in dependency on the current position. Therefore an increment as duration between two valid input events can be either a nominal increment or it can consist of more then one nominal increment.

While a true nominal increment starts with a valid event a virtual increment (of always nominal size) is an increment which starts with a missing event. Each increment which represents a gap (e.g. for synchronization purposes) consists of exactly one true nominal increment and at least one virtual increment, each of them having the same nominal duration (see figure below).

26.16.3 Input signal courses

Typical input signal courses are shown in the figure below.

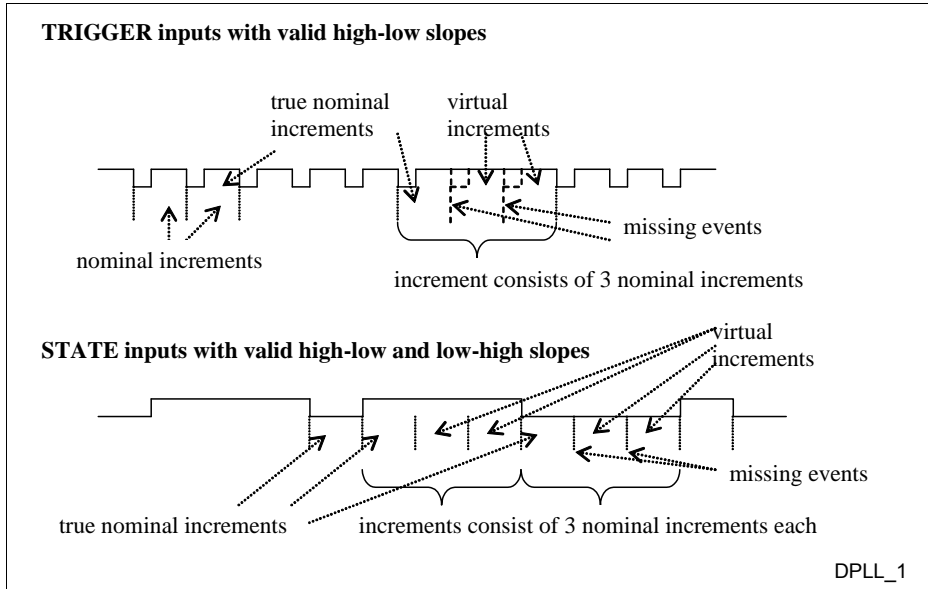


Figure 26-69 Input signal courses

26.16.4 Block and interface description

The block description of the DPLL is shown in the following figure.

26.16.4.1 DPLL Block diagram

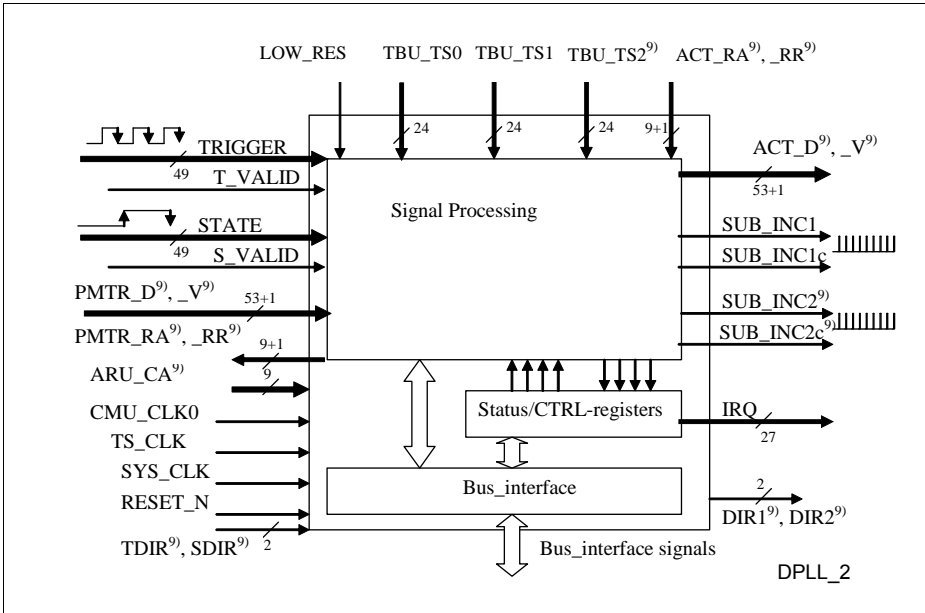


Figure 26-70 DPLL Block diagram

Table 26-46 summarizes the interface signals of the DPLL shown by the block diagram above.

26.16.4.2 Interface description of DPLL
Table 26-46 Interface description of DPLL

Name	Width	I/O	Description	Comment
TRIGGER	49	I	Normal Signal for triggering DPLL by positions/values Bit(48)= TRIGGER_S Bits(47:24)= TRIGGER_FT Bits(23:0)= TRIGGER_TS	one bit signal value (SV), 24 bits filter delay value info and 24 bits time stamp, filtered in different modes.
T_VALID	1	I	The values of TRIGGER are valid	Announces the arrival of a new TRIGGER value
STATE	49	I	Assistance signal for synchronisation STATE(48)= STATE_S STATE(47:24)= STATE_FT STATE(23:0)= STATE_TS	Replacement of signal TRIGGER for emergency situations, or signal from an independent device; bits like above, corresponding
S_VALID	1	I	The values of STATE are valid	Announces the arrival of a new STATE value
PMTR_D	53	I	Position minus time request data, delivered by ARU on request for up to 24 requests PMTR_RR; SV _i =PMTR_D(52:48): ACB bits, directly written to the correspondent DPLL_ACB _j registers PSA _i =PMTR_D(47:24): position value for action DLA _i =PMTR_D(23:0) time delay value for action	Data values for calculation of actual ACTIONS; the values are requested by AEN _i =11 ¹⁾ and CAIP=02 ²⁾ ; a served request is shown by PMTR_V which signals that valid PMTR data arrived and they are written immediately after that to the corresponding RAM regions and registers; The DLA _i values must have the same resolution as the TBU_TS0 input.

Generic Timer Module (GTM)
Table 26-46 Interface description of DPLL (cont'd)

Name	Width	I/O	Description	Comment
PMTR_V	1	I	signals a valid PMTR_D value, that means data is delivered on request	when valid: PMTR_D overwrites data in the PSAi and DLAi registers, also when the corresponding ACT_Ni ³⁾ bit =1;
ARU_CA	9	I	Channel address; for valid PMTR addresses: demand data by setting PMTR_RR=1 when enabled by AENi=11 ¹⁾ and CAIP=02 ²⁾ ;	counter value of ARU selects PMTR_RA and PMTR_RR when a valid address
PMTR_RA	9	O	read address of PMTR access	reflects ID_PMTR_i according to the selected channel address
PMTR_RR	1	O	read request of PMTR access; suppressed for CAIPi=1 (see DPLL_STATUS register)	reflects the value of the corresponding AENi ¹⁾ bit while the correspondent bit CAIPi=0 ²⁾
ACT_D	53	O	Output of a time stamp, a position and a control signal for a calculated action; SV_i=ACT_D(52:48): ACB bits, directly written from the correspondent PMTR_D signals; ACT_D(47:24) is the calculated position value PSACi for the action in relation to TBU_TS1 or 2 ⁴⁾ and ACT_D(23:0) is the time stamp value TSACi for the action in relation to TBU_TS0 ⁴⁾	Future time stamp with the resolution as TBU_TS0 input, additional position information and additional control bits;

Generic Timer Module (GTM)
Table 26-46 Interface description of DPLL (cont'd)

Name	Width	I/O	Description	Comment
ACT_V	1	O	ACT_D value is available and valid; blocking read access	for a valid action address: ACT_V reflects the shadow value of ACT_Ni ³⁾ (ACT_Ni is 1 when new PMTR values are available and the shadow register is updated, when a calculation of the actual PMTR values was done); reset after reading of the ACT_D values
ACT_RA	9	I	ACTION read address;	address bits for selection of all 24 action channels
ACT_RR	1	I	read request of selected action	the action data is demanded from an other module
IRQ	27	O	Interrupt request output	Interrupts of DPLL
SUB_INC1	1	O	Pulse output for TRIGGER input filter	sub-position increment provided continuously
SUB_INC2	1	O	Pulse output for STATE input filter (when TRIGGER and STATE are used for 2 independent devices)	sub-position increment provided continuously
SUB_INC1c	1	O	Pulse output for time base unit 1 in compensation mode (can stop in automatic end mode)	sub-position increment related to TRIGGER input
SUB_INC2c	1	O	Pulse output for time base unit 2 in compensation mode (can stop in automatic end mode)	sub-position increment related to STATE input (when TRIGGER and STATE are used for 2 independent devices)
TS_CLK	1	I	Time stamp clock	used for generation of the time stamps; this clock is used to generate the SUB_INC1,2 pulses

Generic Timer Module (GTM)
Table 26-46 Interface description of DPLL (cont'd)

Name	Width	I/O	Description	Comment
CMU_CLK0	1	I	CMU clock 0	used for rapid pulse correction of SUB_INC1,2
SYS_CLK	1	I	System clock	High frequency clock
RESET_N	1	I	Asynchronous reset signal	Low active; After Reset the DPLL is available only after performing the RAM reset procedures by the DPLL hardware.
LOW_RES	1	I	low resolution of TBU_TS0 selected; shows which of the 27 bits of TBU_TS0 are connected to the DPLL	LOW_RES=0: TBU_TS0(DPLL)= lower 24 Bits of TBU_TS0(TBU); LOW_RES=1: TBU_TS0(DPLL)= higher 24 Bits of TBU_TS0(TBU); In the case LOW_RES=1 the TS0_HRT and/or TS0_HRS bits can be set 5)
TBU_TS0	24	I	Actual time stamp from TBU; is needed to decide, if a calculated action is already in the past	24 bit time input, with a resolution of the time stamp clock
TBU_TS1	24	I	Actual position/value stamp 1; for calculation of position stamps (TRIGGER/STATE)	24 bit pos./val. input, with a resolution of the SUB_INC1 pulses
TBU_TS2	24	I	Actual position/value stamp 2; to be implemented for an additional independent position	ditto for SUB_INC2 for calculation of position stamps (STATE) for $SMC^{(6)}=RMO^{(7)}=1$

Generic Timer Module (GTM)
Table 26-46 Interface description of DPLL (cont'd)

Name	Width	I/O	Description	Comment
TDIR	1	I	Direction of TRIGGER input values (TDIR=0 does mean a forward direction and TDIR=1 a backward direction)	direction information from multiple sensors valid only for SMC ⁶⁾ =1 or IDDS ⁸⁾ = 1
SDIR	1	I	Direction of STATE input values (SDIR=0 does mean a forward direction and SDIR=1 a backward direction)	direction information from multiple sensors valid only for SMC ⁶⁾ =1
DIR1	1	O	Direction information of SUB_INC1 (count forwards for DIR1=0 and backwards for DIR1=1)	count direction of TBU_TS1; DIR1 changes always after the evaluation of the corresponding valid TRIGGER slope and after incrementing/decremented of the address pointer
DIR2	1	O	Direction information of SUB_INC2 (count forwards for DIR2=0 and backwards for DIR2=1)	count direction of TBU_TS2; DIR2 changes always after the evaluation of the corresponding valid STATE slope and after incrementing/decremented of the address pointer

1) see DPLL_CTRL_x register, x=2,3,4

2) see DPLL_STATUS register

3) see DPLL_ACT_STA register

4) see DPLL input signal description

5) see DPLL_CTRL_1 register

6) see DPLL_CTRL_1 register

7) see DPLL_CTRL_0 register

8) see DPLL_CTRL_1 register

26.16.5 DPLL Architecture

26.16.5.1 Purpose of the module

The DPLL generates a predefined number of incremental signal pulses within the period between two events of an input TRIGGER or STATE signal, when the corresponding pulse generator is enabled. The resolution of the pulses is restricted by the frequency of the time stamp clock (TS_CLK). Changes in the period length of the predicted time period of the current increment will result in a change of the pulse frequency in order to get the same number of pulses. This adoption can be performed by DPLL hardware, software or with support of DPLL hardware in different modes.

The basic part of a DPLL is to make a prediction of the current period between two TRIGGER and/or STATE signal edges. Disturbances and systematic failures must be considered as well as changes of increment durations caused by acceleration and deceleration of the supervised process. Therefore, a good estimation is to be done using some measuring values from the past. When the process to be predicted takes a steady and differentiable course not only the current increment but also some more increments for the future can be predicted. In utilisation of such calculations actions for the future can be predicted.

26.16.5.2 Explanation of the prediction methodology

As already shown in [Section 26.16.1](#) the DPLL has to perform different tasks. The basic function for all these tasks is the prediction of the current increment which is based on a relation between increments in the past. Because the relation between two succeeding intervals at a fixed position remains also valid in the case of acceleration or deceleration the prediction of the duration of the current time interval is done by a similarity transformation. Having a good estimation of the current time interval, all the other tasks can be done easily by calculations explained in [Section 26.16.6](#).

26.16.5.3 Clock topology

All registers are read using the system clock SYS_CLK. The SUB_INC1,2 pulses generated have the highest frequency not higher than CMU_CLK0 or TS_CLK respectively. All operations can be performed using the system clock.

26.16.5.4 Clock generation

The clock is generated outside the DPLL.

26.16.5.5 Typical frequencies

For system clock a reasonable clock frequency should be applied to give the DPLL module sufficient computational power to calculate all needed values (prediction of next increment, actions) in time.

26.16.5.6 Time stamps and systematic corrections

The time stamps for the input signals TRIGGER and STATE have 24 bits each. These bits represent the value of the 24 bit free running counter running with a clock frequency selected by the configuration of the TBU. Using a typical frequency of 20 MHz the time stamp represents a relative value of time with a resolution of 50 ns.

The input signals have to be filtered. The filter is not part of the DPLL. The time stamps can have a delay caused by the filter algorithm used. There are delayed and undelayed filter algorithms available and the delay value can depend on a time or a position value.

Systematic deviations of TRIGGER inputs can be corrected by a profile, which also considers systematic missing TRIGGERS. The increments containing missing TRIGGERS are divided into the corresponding number of nominal increments whereas the duration of a nominal increment is the greatest divider of all increment durations.

For each increment this number of enclosed nominal increments is stored in a profile as NT value for TRIGGER. When the increment is a nominal increment the NT value is 1.

For the TRIGGER input the value NT is stored in the ADT_T field in RAM region 2C.

In the case of $AMT^4) = 1$ the ADT_Ti values in the RAM region 2C must also contain the adapting information for the TRIGGER signal, which considers for each increment a systematic physical deviation PD from the perfect increment value with a resolution according to the chosen value of $MLT+1$, which describes the number of SUB_INC1 pulses for a nominal increment.

The value PD for the TRIGGER describes the amount of missing or surplus pulses with a sint13 value, to be added to $MLT+1$ directly. The correction value is in this way also applicable in the case of missing TRIGGER inputs for the synchronization gaps. In this case the amount of provided SUB_INC1 pulses for a nominal increment ($MLT+1$) is multiplied by NT first before the PD value is added.

The NT value of the current increment is stored in the variable SYN_T (see NUTC register in [Section 26.16.11.13](#)).

In the case of $RMO^4) = 1$ for $SMC^5) = 0$ (emergency mode) the time stamp of STATE is used to generate the output signal SUB_INC1.

More inaccuracy should be accepted in emergency mode because usually there are only fewer events available for FULL_SCALE according to the value $SNU^4)$.

For the STATE signal the systematic deviations of the increments can be corrected in the same way as for TRIGGER by profile and adaptation information as described below.

Generic Timer Module (GTM)

Systematic deviations of STATE inputs can be corrected by a profile, which also considers systematic missing STATE events. The increments containing missing STATES are divided into the corresponding number of nominal increments whereas the duration of a nominal increment is the greatest divider of all increment durations.

For each increment this number of enclosed nominal increments is stored in a profile as NS value for STATE. When the increment is a nominal increment the NS value is 1.

For the STATE input the value NS is stored in the ADT_S field in RAM region 1C3.

In the case of $AMS^4) = 1$ the ADT_Si values in the RAM region 1C3 must contain the adapting information for the STATE signal, which considers for each increment a systematic physical deviation PD_S from the perfect increment value with a resolution according to the chosen value of MLS1, which describes the number of SUB_INC1 pulses for a nominal increment (see below).

The number of pulses SUB_INC1 for a nominal STATE increment in emergency mode (for SMC=0) is given by the value of $MLS1 = (MLT + 1) * (TNU + 1) / (SNU + 1)$ in order to get the same number of pulses in FULL_SCALE for normal and emergency mode. This value has to be configured by the CPU.

The value PD_S for the STATE describes the amount of missing or surplus pulses with a sint16 value, to be added to MLS1 directly. The correction value is in this way also applicable in the case of missing STATE inputs for the synchronization gaps. In this case the amount of provided SUB_INC1 pulses for a nominal increment MLS1 is multiplied by NS first before the PD_S value is added.

The current NS value is stored in the variable SYN_S (see NUSC register in [Section 26.16.11.14](#)).

For references above the following hints are used:

- 1) see DPLL_CTRL_x register, x=2,3,4
- 2) see DPLL_STATUS register
- 3) see DPLL_ACT_STA register
- 4) see DPLL_CTRL_0 register
- 5) see DPLL_CTRL_1 register
- 6) see DPLL input signal description

26.16.5.7 DPLL Architecture overview

As shown in [Section 26.16.4.1](#) the DPLL can process different input signals. The signal TRIGGER is the normal input signal which gives the detailed information of the supervised process. It can be for instance the information of water or other liquid level representing the volume of the liquid, where each millimeter increasing results in a TRIGGER signal generation. In order to get a predefined filling level, without overflow also the inertia of the system must be taken into account. Hence, some delay for closing

Generic Timer Module (GTM)

the inlet valve and also the remaining water amount in the pipe must be considered in order to start the closing action earlier as the filling level will be reached.

A second input signal STATE sends an additional (redundant) information for instance at some centimeters and because of intervals with different distances it gives also information about the system state with the direction of the water flow (in or out), while the TRIGGER signal must not contain information concerning the flow direction. In some applications the inactive slope of TRIGGER can be utilized to transmit a direction information. In the case of faults in the TRIGGER signal the STATE signal is to be processed in order to reach the desired value nevertheless, maybe with some loss of accuracy.

The measuring scale can have some systematic failures, because not all millimeter or centimeter distances measured mean the same value. This could be due to changes in the thickness of the measuring cylinder or the inaccurate position of the marks. These systematic failures are well known by the system and for improvement of the prediction the signals ADT_T and ADT_S for the correction of the systematic failures of TRIGGER and STATE respectively are stored in the internal RAM.

The input signals TRIGGER and STATE are represented as a time stamp signal each, which is stored in the 24 bit TS-part of the corresponding signal.

Information concerning the delay of this signal by filtering of disturbances is stored in the 24 bit FT-part of the signal.

In order to establish the relation of time stamps to the actual time the TBU_TS0⁶⁾ value is also provided showing the actual time value used for prediction of actions in the future.

After reaching the desired water level the water is filled in a bottle by draining. After that the water filling is repeated. The water level at draining is observed by the same sensor signals (the same number of TRIGGER pulses), but the duration of the draining could be different from the filling time. Both times together form the FULL_SCALE region, while one of them is a HALF_SCALE region, which can differ in time but not in the number of pulses, especially for TRIGGER.

For synchronisation purposes some TRIGGER marks can be omitted in order to set the system to a proper synchronisation value (maybe before the upper filling value is reached).

In emergency situations, when the TRIGGER signals are missed the STATE signal is used instead of.

The PMTR_⁶⁾ signals announce the request for a position minus time calculation for up to 24 events.

All 24 events can be activated using the 24 AENi¹⁾ (action enable) bits. Each of these enable bits are asked by the routing engine for a read access. The corresponding read request is generated by the AENi bit while CAIPx is zero. CAIP1 and CAIP2 are two bits of the DPLL_STATUS register for 12 actions each with the meaning "calculation of

Generic Timer Module (GTM)

actions in progress”, controlled by the state machine (see [Chapter 26.16.2](#)) for scheduling the operations.

When such a request is serviced by the ARU (in the case CAIPx=0) the values for position and time are written in the corresponding RAM 1A region (0x0200... 0x025C for the position value and 0x0260... 0x02BC for the delay value), the control bits for the corresponding action are set accordingly. When a new PMTR value arrives, an old value is overwritten without notice and the shadow bit of ACT_Ni is cleared while the ACT_Ni (new action) bit in the DPLL_ACT_STA register is set. The ACT_Ni is cleared, when the currently calculated action value is in the past. Overwriting of old information is possible without data inconsistency because the read request to ARU is suppressed during action calculations by the CAIP1,2 bits. In this way always the last possible PMTR value is used consistently.

For references above the following hints are used:

- 1) see DPLL_CTRL_x register, x=2,3,4
- 2) see DPLL_STATUS register
- 3) see DPLL_ACT_STA register
- 4) see DPLL_CTRL_0 register
- 5) see DPLL_CTRL_1 register
- 6) see DPLL input signal description

26.16.5.8 DPLL Architecture description

The DPLL block diagram [Chapter 26.16.4.1](#) will now be explained in detail in combination with some example configurations of the control registers. There are different configuration bits available which can adopt the DPLL to the use case (see [Chapter 26.16.11](#)).

Let for example in HALF_SCALE the TRIGGER number TNU⁴) be 0x3B (which is for TNU+1 = 60 decimal that does mean 120 events in FULL_SCALE) and the number of SUB_INC1 pulses between two TRIGGERs MLT⁴) be 0x257 (this means 600 pulses per TRIGGER event). Then the FULL_SCALE region can be divided into 72000 parts each of them associated with its own SUB_INC1 pulse. For a run through FULL_SCALE all 72000 pulses should appear but maybe with a different pulse frequency between two TRIGGER events. For this example after each 600 pulses at the SUB_INC1 output the next TRIGGER event is to be expected with the corresponding new time stamp.

Missing SUB_INC1 pulses due to acceleration have to be taken into account within the next increment. Not one pulse has to be missed or added because of calculation inaccuracy in average for a sufficient number of FULL_SCALE periods. This means that not one pulse is sent in addition and all missing pulses are to be caught up on afterwards.

For the systematic arrangement of TRIGGER inputs the profile (as already mentioned in [Section 26.16.5.6](#) is stored in the RAM region 2C (see [Section 26.16.11.100](#)). In this

Generic Timer Module (GTM)

field the relative position of gaps can be stored in the NT value and also physical deviations in the PD value.

For the consideration of systematic missing TRIGGERS the actual NT value of the profile is stored in the SYN_T bits of the NUTC register (see [Section 26.16.11.13](#)).

In normal mode the physical deviation values PD in the ADT_T field could be used to balance the local systematic inaccuracy of the TRIGGER signal. The value of PD (see [Section 26.16.11.100](#)) is the pulse difference in the corresponding increment and does mean the number of sub pulses to be added to the nominal number of pulses. PD is a signed integer value using 13 bits: up to +/-4096 pulses can be added for each increment.

The NT value of the profile ADT_T has the value 1, when a nominal increment is assumed. An integer number greater than 1 shows the number of nominal increments to be considered for a gap. For the actual increment after synchronization the corresponding NT value is stored in SYN_T of the NUTC register.

Using the STATE input there are similar configuration bits available (see [Section 26.16.11](#)).

Let for example in HALF_SCALE the STATE number SNU⁴) be 0xB (which is for SNU+1 =12 decimal and while SYSF⁴) =0 that does mean 24 events in FULL_SCALE). In order to get the same number of SUB_INC1 pulses for FULL_SCALE as above for TRIGGERS the value (MLT+1)=600 is divided by 2*(SNU+1)=24 and multiplied with 2*(TNU+1)=120. The result 3000 must be stored in MLS1 by the CPU (see [Section 26.16.11.78](#)).

For the systematic arrangement of STATE inputs the profile (as already mentioned in [Section 26.16.5.6](#) is stored in the RAM region 1C3 (see [Section 26.16.11.93](#)). In this field the relative position of gaps can be stored in the NS value and also physical deviations in the PD_S value.

For the consideration of systematic missing TRIGGERS the actual NS value of the profile is stored in the SYN_S bits of the NUSC register (see [Section 26.16.11.14](#)).

In emergency mode the physical deviation values PD_S in the ADT_S field could be used to balance the local systematic inaccuracy of the STATE signal. The value of PD_S (see [Section 26.16.11.93](#)) is the pulse difference in the corresponding increment and does mean the number of sub pulses to be added to the nominal number of pulses. PD_S is a signed integer value using 16 bits: up to +/-32768 pulses can be added for each increment.

The NS value of the profile ADT_S has the value 1, when a nominal increment is assumed. An integer number greater than 1 shows the number of nominal increments to be considered for a gap. For the actual increment after synchronization the corresponding NS value is stored in SYN_S of the NUSC register.

For references above the following hints are used:

1) see DPLL_CTRL_x register, x=2,3,4

- 2) see DPLL_STATUS register
- 3) see DPLL_ACT_STA register
- 4) see DPLL_CTRL_0 register
- 5) see DPLL_CTRL_1 register
- 6) see DPLL input signal description

26.16.5.9 Block diagrams of time stamp processing.

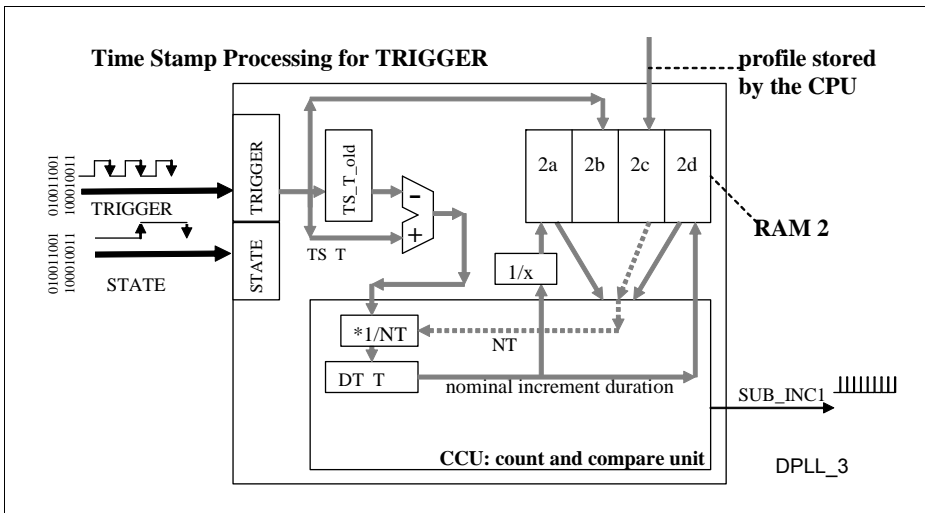


Figure 26-71 Time Stamp Processing

As shown in the block diagram above the time stamp difference of two succeeding input events is calculated. For the prediction of the current increment duration such values from the past are used. For this purpose the measured and calculated values of the last FULL_SCALE period are stored in the RAM. For the TRIGGER input there are 4 different RAM parts in the RAM region 2:

- 2A stores the reciprocals of each nominal increment duration RDT_T
- 2B stores the time stamps of each valid input event TSF_T
- 2C is used for the profile ADT_T and
- 2D for the nominal increment durations DT_T.

Because the prediction is based on the relations of increments in the past this relation can be calculated easily by the multiplication of increment duration values with the reciprocal value of an other increment. In order not to be forced to distinguish between

Generic Timer Module (GTM)

gaps and "normal" increment durations also for gaps only the nominal duration and the correspondent reciprocal values are stored in the RAM field. This is possible by consideration of the NT value in the profile: the measured increment duration is divided by NT.

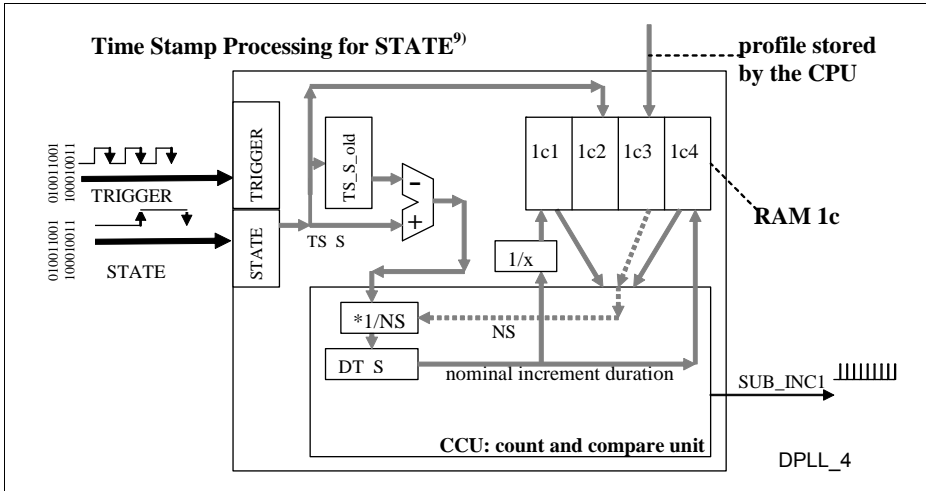


Figure 26-72 Time Stamp Processing for STATE

For the STATE input there are also 4 different RAM parts in the RAM region 1C:

- 1C1 stores the reciprocals of each nominal increment duration RDT_S
- 1C2 stores the time stamps of each valid input event TSF_S
- 1C3 is used for the profile ADT_S and
- 1C4 for the nominal increment durations DT_S.

The calculations are performed similar as for the TRIGGER input. The NS value in the profile shows the appearance of a gap.

26.16.5.10 Register and RAM address overview

The address map of the DPLL is divided into register and memory regions as defined in [Table 26-47](#). The addresses from 0x0000 to 0x00FC are reserved for registers, from 0x0100 to 0x01FC is reserved for action registers to serve the ARU at immediately read request.

The RAM is divided into 3 independent accessible parts 1A, 1B+C and 2.

The part 1A from 0x0200 to 0x037C is used for PMTR values got from ARU and intermediate calculation values; there is no write access from the CPU possible, while the DPLL is enabled.

Generic Timer Module (GTM)

The RAM 1B part from 0x0400 to 0x05FC is reserved for RAM variables and the RAM part 1C from 0x0600 to 0x09FC is used for the STATE signal values.

The RAM region 2 from 0x4000 to 0x4FFC is reserved for the TRIGGER signal values. RAM region 1A has a size of 288 bytes, Ram 1B+C uses 1,125 Kbytes while RAM region 2 3 Kbytes, depending on the number of TRIGGER events in FULL_SCALE. The AOSV_2 register is used to determine the beginning of each part.

The table [Table 26-47](#) in gives the DPLL Address map overview

Register and RAM address map

Registers are used to control the DPLL and to show its status. Also parameters are stored in registers when useful. The table below shows the addresses for status and control registers as well as values stored in additional registers. The register meaning explained in the register overview ([Section 26.16.10](#)) while the bit positions of the status and control registers are described in detail in [Section 26.16.11](#).

Time stamps for TRIGGER and STATE can have either the same resolution as the TBU_TS0 input or 8 times higher. This is configured in the DPLL_CTRL_1 register (see [Section 26.16.11.2](#)). While the TBU_TS0 is used for action predictions the higher resolution of TRIGGER and STATE inputs can be used for a more accurate pulse generation.

The time stamp fields of TRIGGER and STATE are stored in the corresponding RAM regions in such a way, that for a gap also entries for the virtual increments are provided. This is due to the necessity to calculate time differences between a given number of (real and virtual) input events independent of a gap. Therefore the gap is extended in the RAM fields 2B and 1C2.

For all other RAM regions in RAM 2 and RAM 1C the gap is considered as one increment.

For the access to the RAM fields there must be address pointers. When the device starts all address pointers have a zero value and the first measured and calculated values are stored in the beginning of the corresponding RAM field.

Because the position of the device is usually unknown at the beginning no profile information can be used. The profile regions must have their own address pointers each which are set by the CPU as soon as the position is known. By setting the appropriate value to the address pointer APT_2C of the TRIGGER profile or APT_1C3 of the STATE profile respectively the synchronization bits in the DPLL_STATUS register SYT or SYS are set respectively. In the following the gap information can be used.

Because the time stamp fields are extended at the gaps there must be additional address pointers for these regions: APT_2B for TRIGGER time stamps and APT_1C2 for STATE time stamps. These address pointers must be increment by NT or NS respectively when a gap appears.

Table 26-47 DPLL Register and RAM address map

Addr. range Start	Addr. range End	Value number	Byte #	Content	Indication	Region	RAM size
0x0000	0x0FC	64	256	Register	used/reserved	0	no RAM
0x100	0x1FC	64	192	ACTION registers	direct read from ARU	0	no RAM
0x0200	0x03FC	128	384	PMTR values RAM 1A	CPU rw access, when DPLL disabled ; ARU has highest priority	1A with own ports	RAM part 1A: 384 bytes
0x0400	0x05FC	128	384	Variables RAM 1B	r and monitored w access by the CPU	1B	RAM part 1B+C: 1,125 Kbytes
0x0600	0x09FC	256	768	STATE data	r and monitored w access by the CPU	1C	
0x0600	0x06FC	64	192	RDT_Si	STATE reciprocal values	1C1	
0x0700	0x07FC	64	192	TSF_Si	STATE TS values	1C2	

Generic Timer Module (GTM)
Table 26-47 DPLL Register and RAM address map (cont'd)

Addr. range Start	Addr. range End	Value number	Byte #	Content	Indication	Region	RAM size
0x0800	0x08FC	64	192	ADT_Si	adapt values of STATE	1C3	
0x0900	0x09FC	64	192	DT_Si	nom. STATE inc	1C4	
0x4000	0x4FFC	1024	3072	TRIGGER data	r and monitored w access of CPU	2	RAM part 2: 3Kbytes
0x4000	0x43FC	256	768	RDT_Ti	TRIGGER reciprocal values	2A	
0x4400	0x47FC	256	768	TSF_Ti	TRIGGER TS values	2B	
0x4800	0x4FFC	256	768	ADT_Ti	adapt values of TRIGGER	2C	
0x4C00	0x57FC	256	768	DT_Ti	nom. TRIGGER increments	2D	

RAM Region 1

RAM region 1 has a size of 1,5 Kbytes and is used to store variables and parameters as well as the measured and calculated values for increments of STATE. The RAM 1 region is divided into two independent accessible RAM parts (A and B+C) with own ports. The address information is shown in the table above and the detailed description is

Generic Timer Module (GTM)

performed in the following sections. The RAM 1A is used to store the PMTR values got from ARU and in addition some intermediate calculation results of actions. RAM region 1B is used for variables needed for the prediction of increments, while RAM 1C is used to store time stamps, profile and durations for all the STATE inputs of the last FULL_SCALE region. All variables and values of RAM 1B+C part use a data width of up to 24 bits.

The RAM is to be initialized by the DPLL after HW-reset. All RAM cells must have a zero value after performing the initialize procedure. This is performed when setting The Init_RAM bit in the DPLL_RAM_INI register. The DPLL is only available after finishing this procedure. The initialization progress is shown in the status bits of the same register.

- RAM Region 1A: used for storage of PMTR values got from ARU;
 - read and write access by the CPU is only possible, when the DPLL is disabled.
 - The CPU Address range: 0x0200 – 0x03FC
- RAM Region 1B: usable for intermediate calculations and auxiliary values, data width of 3 bytes used for 24 bit values;
 - A write access to this region results in an interrupt to the CPU, when enabled.
 - Address range: 0x0400 – 0x05FC
- RAM Region 1C: Values of all STATE increments in FULL_SCALE, data width of 3 bytes used for 24 bit values;
 - A write access to this region results in an interrupt to the CPU, when enabled.
 - Address range: 0x0600 – 0x09FC

In RAM region 1C there is a difference in the amount of data. While for the RAM regions 1C1, 1C3 and 1C4 there are $2^*(SNU+1-SYN_NS)$ entries for $SYSF=0$ or $2^*(SNU+1) - SYN_NS$ entries for $SYSF=1$, for the RAM region 1C2 there are $2^*(SNU+1)$ entries (see DPLL_CTRL_0 and _1 registers). For the latter also the virtual events are considered, that means the gap is divided into equidistant parts each having the same position share as increments without a gap. For that reason the CPU must extend the stored TSF_Si values in the RAM region 1C2 before the APS_1C3 is written. The write access to APS_1C3 sets the SYS bit in the DPLL_Status register in order to show the end of the synchronization process. Only when the SYS bit is set the PMTR values can consider more then the last increment duration for the action prediction by setting NUSE to a corresponding value.

Note: RAM regions 1B and 1C have a common port.

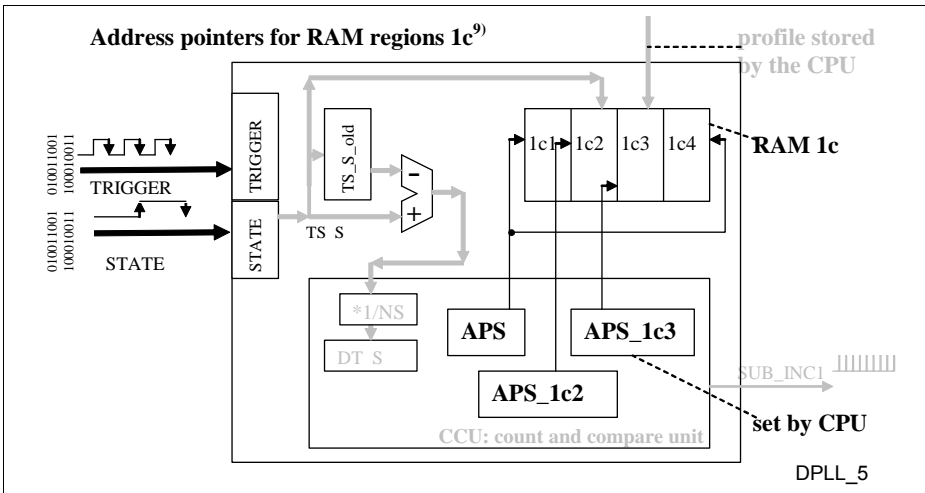


Figure 26-73 Address pointers for RAM regions 1C

The address pointers for RAM region 1C are shown in the diagram above. While the address pointer APS points to the RAM regions 1C1 and 1C4, the address pointer APS_1C2 points to the time stamp field in the region 1C2. This is necessary, because in the time stamp field the gaps are extended to the number of nominal increments (see explanation above and also to the synchronization procedure explained in [Synchronization description](#)).

The address pointer APS_1C3 is set by the CPU when the position is known and therefore the relation to the other address pointers is calculated. This setting of this profile address pointer synchronizes the RAM fields to one another. The synchronization is shown in the DPLL_STATUS register by the SYS bit.

RAM Region 2

The RAM region 2 has 3 Kbytes and is used to store measured and calculated values for increments of TRIGGER. The address information is explained in [Section 26.16.10](#) while the meaning is explained in this section.

Because of up to 128 TRIGGER events in HALF_SCALE the fields 2A, B C and D must have up to 256 storage places each. For 3 Bytes word size this does mean 3 k Byte of RAM region 2.

In order to save RAM size for configurations with less TRIGGER events the RAM is configurable by the offset switch Register OSW (0x001C) and the address offset value register of RAM region 2 AOSV_2 (0x0020). The RAM is to be initialized by the DPLL after HW-reset. All RAM cells must have a zero value after performing the initialize procedure. The DPLL is only available after finishing this procedure.

Generic Timer Module (GTM)

In RAM region 2 there is a difference in the amount of data. While for the RAM regions 2A, 2C and 2D there are $2 \cdot (TNU+1-SYN_NT)$ entries, for the RAM region 2B there are $2 \cdot (TNU+1)$ entries (see DPLL_CTRL_0 and _1 registers). For the latter also the virtual events are considered, that means the gap is divided into equidistant parts each having the same position share as increments without a gap. For that reason the CPU must extend the stored TSF_Ti values in the RAM region 2B before the APT_2C is written.

The write access to APT_2C sets the SYT bit in the DPLL_Status register in order to show the end of the synchronization process. Only when the SYT bit is set the PMTR values can consider more then the last increment duration for the action prediction by setting NUTE to a value greater then one.

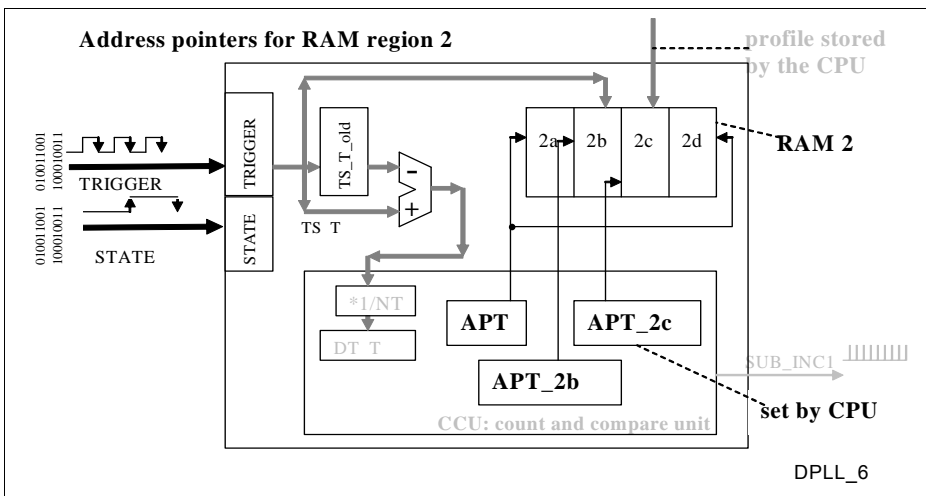


Figure 26-74 Address pointers for RAM region 2

The address pointers for RAM region 2 are shown in the diagram above. While the address pointer APT points to the RAM regions 2A and 2D, the address pointer APT_2B points to the time stamp field in the region 2B. This is necessary, because in the time stamp field the gaps are extended to the number of nominal increments (see explanation above and also to the synchronization procedure explained in [Synchronization description](#)).

The address pointer APT_2C is set by the CPU when the position is known and therefore the relation to the other address pointers is calculated. This setting of this profile address pointer synchronizes the RAM fields to one another. The synchronization is shown in the DPLL_STATUS register by the SYT bit.

26.16.6 Prediction of the current increment duration

26.16.6.1 The use of increments in the past

Past values to be considered for the prediction of TRIGGER

In order to take into account values of increments for TRIGGERS in the past, the NUTE value is configured to determine the number of past values. In addition the VTN has a value according to the number of virtual increments in the NUTE region. Because gaps come in to the NUTE region or leave it the VTN value must be updated by the CPU until NUTE is set to HALF_SCALE or FULL_SCALE. For the RAM regions 2A and 2D the value NUTE-VTN is to be considered while for the RAM region 2B only the NUTE value is to be considered. This is due to the fact that the time stamp entries in a gap are extended to the number of nominal increments, but duration entries not.

Past values to be considered for the prediction of STATE

In order to take into account values of increments for STATE in the past, the NUSE value is configured to determine the number of past values. In addition the VSN has a value according to the number of virtual increments in the NUSE region. Because gaps come in to the NUSE region or leave it the VSN value must be updated by the CPU until NUSE is set to HALF_SCALE or FULL_SCALE. For the RAM regions 1C1 and 1C4 in the past the value NUSE-VSN is to be considered while for the RAM region 1C2 only the NUSE value is to be considered. This is due to the fact that time stamp entries in a gap are extended to the number of nominal increments, but duration entries not.

26.16.6.2 Increment prediction in Normal Mode forwards (DIR1=0)

For the prediction of increments and actions in normal mode the values are calculated as described in the following equations.

Please note, that the ascending order of calculation must be hold in order not to lose results still needed. It is important for TRIGGER values to calculate and store in the RAM region 2 all values according to equations up to DPLL-14 before DPLL-1a4...7, DPLL-1b1 and DPLL-1C1, while the last one overwrites DT_Ti when NUTE (see [Section 26.16.11.13](#)) is set to the FULL_SCALE range. Because the old value of DT_Ti is also needed for equation DPLL-10 and DPLL-11 this value is stored temporarily at DT_T_ACT as shown by equation DPLL-1a or DPLL-1b respectively until all prediction calculations are done and after that equation DPLL-1a4...7, DPLL-1b1 and DPLL-1c1 updates DT_Ti: update DT_Ti after calculations of equation DPLL-14. For p=APT calculates in normal mode:

When using filter information of TRIGGER_FT, selected by IDT=1, it must be distinguished by IFP, if this filter information is time or position related:

Generic Timer Module (GTM)

In order to make possible to perform the automatic resolution corrections of equations DPLL-1a1a the filter unit in TIM module must operate using the time stamp clock.

Equations DPLL-1a to calculate TRIGGER time stamps

Equations DPLL-1a to calculate TRIGGER time stamps

For calculation of time stamps use the filter delay information

$$TS_T = TRIGGER_TS \text{ (unchanged for } IDT=0) \quad \text{(DPLL-1a0)}$$

$$TS_T = TS_T - FTV_T \text{ (for } IDT=1 \text{ and } IFP=0) \quad \text{(DPLL-1a1)}$$

with

$$FTV_Tx = FTV_T/8 \quad \text{(for } LOW_RES = 1 \text{ and } TS0_HRT = 0) \quad \text{(DPLL-1a1a)}$$

$$FTV_Tx = FTV_T \quad \text{(for } LOW_RES = 0 \text{ or } TS0_HRT = 1) \quad \text{(DPLL-1a1b)}$$

and

$$TS_T = TS_T - FTV_T * (CDT_TX/NMB_T)_{old}^{(1)} \text{ for } (IDT=1 \text{ and } IFP=1) \quad \text{(DPLL-1a2)}$$

this can be also calculated using the value of ADD_IN_CALN:

$$TS_T = TS_T - FTV_T * (1/ADD_IN_CALN_{old}^{(1)}) \text{ for } (IDT=1 \text{ and } IFP=1) \quad \text{(DPLL-1a3)}$$

- 1) Consider values, calculated for the last increment; position related filter values are only considered up to at least 1 ms time between two TRIGGER events. The reciprocal value is stored using a 32 bit fractional part, while only the 24 lower bits are used - for explanation see note 4) at DPLL_CTRL0 register. The value of 1/ADD_IN_CALN_OLD or (CDT_TX/NMB_T)_old is set to 0xFFFFF in case of an overflow.

Note: CDT_TX is the predicted duration of the last TRIGGER increment and NMB_T the calculated number of SUB_INC1 events in the last increment, because the new calculations are done by equations DPLL-5 and DPLL-21 for the current increment after that. Therefore in equation DPLL-1a3 the value ADD_IN of the last increment is used (see equation DPLL-25). SYN_T_OLD is the number of TRIGGER events including missing TRIGGERs as specified in the NUTC register for the last increment, with the initial value of 1.

For storage of time stamps in the RAM see also equations DPLL-1a4 ff. after calculation of actions

Equation DPLL-1b to calculate DT_T_ACT (nominal value)

$$DT_T_ACT = (TS_T - TS_T_OLD) / SYN_T_OLD \quad \text{(DPLL-1b)}$$

For the case SYT=0 (still no synchronization to the profile) the values SYN_T and SYN_T_OLD are still assumed as having the value 1.

Equation DPLL-1c to calculate RDT_T_ACT (nominal value)

$$RDT_T_ACT = 1 / DT_T_ACT \quad \text{(DPLL-1c)}$$

Equation DPLL-2a1 to calculate QDT_T_ACT

Relation of the recent last two increment values for APT=p in forward direction (DIR1=0)

$$QDT_T_ACT = DT_T_ACT * RDT_T[p-1] \quad \text{(DPLL-2a1)}$$

QDT_T_ACT as well as QDT_Ti have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

Note: QDT_T_ACT uses a 6 bit integer part and a 18 bit fractional part.

Equation DPLL-3 to calculate the error of last prediction

When q = NUTE-VTN considers for the error calculation only the last valid prediction values for DIR1=0:

Calculate the error of the last prediction when using only RDT_T_FS1, DT_T[p-q] and DT_T[p-1] for the prediction of DT_T[p]:

$$EDT_T = DT_T_ACT - (DT_T[p-1] * QDT_T[p-q]) \quad \text{(DPLL-3)}$$

with

$$QDT_T[p-q] = DT_T[p-q] * RDT_T[p-q-1] \text{ for } FST = 0 \quad \text{(DPLL-2b1)}$$

Generic Timer Module (GTM)

$$QDT_T[p-q] = DT_T[p-q] * RDT_T_FS1 \text{ for } FST = 1 \quad \text{(DPLL-2b2)}$$

and

FST has the meaning: NUTE = FULL_SCALE (see NUTC register)

while

QDT_T_ACT as well as QDT_Ti have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

Note: QDT_T[p-q] uses a 6 bit integer part and a 18 bit fractional part.

Equation DPLL-4 to calculate the weighted average error

for SYT=1 calculate:

$$MEDT_T := (EDT_T + MEDT_T) / 2 \quad \text{(DPLL-4)}$$

Equations DPLL-5 to calculate the current increment value

nominal increment value:

$$CDT_TX_nom = (DT_T_ACT + MEDT_T) * QDT_T[p-q+1] \quad \text{(DPLL-5a)}$$

with (for $q > 1$):

$$QDT_T[p-q+1] = DT_T[p-q+1] * RDT_T[p-q] \quad \text{(DPLL-2c)}$$

and for $q=1$ use equation DPLL-2a1.

while

QDT_T_ACT as well as QDT_Ti have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

and the expected duration to the next TRIGGER event

$$CDT_TX = CDT_TX_nom * SYN_T \quad \text{(DPLL-5b)}$$

Note: QDT_T[p-q+1] uses a 6 bit integer part and a 18 bit fractional part.

Note: In the case of an overflow in equations DPLL-5a or b set the value to 0xFFFFFFFF and the corresponding CTO bit in the DPLL_STATUS register. In the case of negative values set CDT_TX to 0x0 without any effect to the CTO bit.

26.16.6.3 Increment prediction in Emergency Mode forwards (DIR2=0)

Please note, that the ascending order of calculations for STATE and storage of the values in the RAM region 1C must be hold in order not to lose results still needed. The same considerations as done for DT_T_ACT are valid for DT_S_ACT (equation DPLL-6a4...7, DPLL-6b1 and DPLL-6c1): update TD_Si only after calculations of equation DPLL-14.

When using filter information of STATE_FT, selected by IDS=1, it must be distinguished by IFP, if this filter information is time or position related:

In order to make possible to perform the automatic resolution corrections of equations DPLL-6a1a the filter unit in TIM must operate using the time stamp clock.

Equations DPLL-6a to calculate STATE time stamps

For calculation of time stamps use the filter delay information and use p=APS while DIR2=0:

$$TS_S = STATE_TS \quad (\text{for } IDS=0, \text{ received unchanged value}) \quad (\text{DPLL-6a0})$$

$$TS_S = TS_S - FTV_Sx \quad (\text{for } IDS=1 \text{ and } IFP=0) \quad (\text{DPLL-6a1})$$

with

$$FTV_Sx = FTV_S/8 \quad (\text{for } LOW_RES = 1 \text{ and } TS0_HRS = 0) \quad (\text{DPLL-6a1a})$$

$$FTV_Sx = FTV_S \quad (\text{for } LOW_RES = 0 \text{ or } TS0_HRS = 1) \quad (\text{DPLL-6a1b})$$

and

$$TS_S = TS_S - FTV_S * (CDT_SX/NMB_S)_{old}^{(1)} \quad (\text{for } IDS=1 \text{ and } IFP=1) \quad (\text{DPLL-6a2})$$

this can be also calculated using the value of ADD_IN_CALE:

$$TS_S = TS_S - FTV_S * (ADD_IN_CALE)_{old}^{(1)} \quad (\text{for } IDS=1 \text{ and } IFP=1) \quad (\text{DPLL-6a3})$$

with

see also equations DPLL-6a4 ff. at [Chapter 26.16.6.2](#) for TRIGGER.

- 1) Consider values, calculated for the last increment; position related filter values are only considered up to at least 1 ms time between two STATE events. The reciprocal value is stored using a 32 bit fractional part, while only the 24 lower bits are used - for explanation see note 4) at DPLL_CTRL0 register. The value of 1/ADD_IN_CALE_old or (CDT_SX/NMB_S)_old is set to 0xFFFFFFFF in case of an overflow.

Generic Timer Module (GTM)

Note: CDT_SX is the predicted duration of the last STATE increment and NMB_S the calculated number of SUB_INC1 events in the last increment, because the new calculations are done by equations DPLL-10 and DPLL-22 respectively for the current increment after that. Therefore in equation DPLL-6a3 the value ADD_IN of the last increment is used (see equation DPLL-26). SYN_S_OLD is the number of increments including missing STATEs as specified in the NUSC register for the last increment with the initial value of 1. The update to the RAM region 1C4 is done after all related calculations (see equation DPLL-6d -after DPLL-14- for this reason).

Equation DPLL-6b to calculate DT_S_ACT (nominal value)

$$DT_S_ACT = (TS_S - TS_S_OLD) / SYN_S_OLD \quad \text{(DPLL-6b)}$$

For the case SYS=0 (still no synchronization to the profile) the values SYN_S and SYN_S_OLD are still assumed as having the value 1.

Equation DPLL-6c to calculate RDT_S_ACT (nominal value)

$$RDT_S_ACT = 1 / DT_S_ACT \quad \text{(DPLL-6c)}$$

Equation DPLL-7a1 to calculate QDT_S_ACT

for APS=p in forward direction (DIR2=0)

$$QDT_S_ACT = DT_S_ACT * RDT_S[p-1] \quad \text{(DPLL-7a1)}$$

QDT_S_ACT as well as QDT_Si have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

Note: QDT_S_ACT uses a 6 bit integer part and a 18 bit fractional part.

Equation DPLL-8 to calculate the error of last prediction

with q= NUSE-VSN when using QDT_S[p-q] and DT_S[p-1] for the prediction of DT_S[p]

Generic Timer Module (GTM)

$$\text{EDT_S} = \text{DT_S_ACT} - (\text{DT_S}[p-1] * \text{QDT_S}[p-q]) \quad \text{(DPLL-8)}$$

and with

$$\text{QDT_S}[p-q] = \text{DT_S}[p-q] * \text{RDT_S}[p-q-1] \text{ for FSS} = 0 \quad \text{(DPLL-7b1)}$$

$$\text{QDT_S}[p-q] = \text{DT_S}[p-q] * \text{RDT_S_FS1} \text{ for FSS} = 1 \quad \text{(DPLL-7b1)}$$

and

FSS has the meaning: NUSE = FULL_SCALE (see NUSC register)

QDT_S_ACT as well as QDT_Si have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

Note: QDT_S[p-q] uses a 6 bit integer part and a 18 bit fractional part.

Equation DPLL-9 to calculate the weighted average error

for SYS=1 calculate:

$$\text{MEDT_S} := (\text{EDT_S} + \text{MEDT_S})/2 \quad \text{(DPLL-9)}$$

Equations DPLL-10 to calculate the current increment (nominal value)

$$\text{CDT_SX_nom} = (\text{DT_S_ACT} + \text{MEDT_S}) * \text{QDT_S}[p-q+1] \quad \text{(DPLL-10a)}$$

with

$$\text{QDT_S}[p-q+1] = \text{DT_S}[p-q+1] * \text{RDT_S}[p-q] \quad (\text{for } q > 1) \quad \text{(DPLL-7c)}$$

see equation DPLL-7a1 for q=1

while

QDT_S_ACT as well as QDT_Si have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

and the expected duration to the next STATE event

$$\text{CDT_SX} = \text{CDT_SX_nom} * \text{SYN_S} \quad \text{(DPLL-10b)}$$

Generic Timer Module (GTM)

Note: QDT_S[p-q+1] uses a 6 bit integer part and a 18 bit fractional part.

Note: In the case of an overflow in equations DPLL-10a or b set the value to 0xFFFFF and the corresponding CSO bit in the DPLL_STATUS register. In the case of negative values set CDT_SX to 0x0 without any effect to the CSO bit.

All 5 steps above (DPLL-6 to DPLL-10) are only needed in emergency mode. For the normal mode the calculations of equations DPLL-6 and DPLL-7 are done solely in order to get the values needed for a sudden switch to emergency mode.

26.16.6.4 Increment prediction in Normal Mode backwards (DIR1=1)
Equations DPLL-2a2 to calculate QDT_T_ACT backwards

$$\text{QDT_T_ACT} = \text{DT_T_ACT} * \text{RDT_T}[p+1] \quad \text{(DPLL-2a2)}$$

QDT_T_ACT as well as QDT_Ti have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

Equation DPLL-3a to calculate of the error of last prediction

When q = NUTE-VTN and DIR1=1 using only RDT_T[p+q+1], DT_T[p+q] and DT_T[p+1] for the prediction of DT_T[p]

$$\text{EDT_T} = \text{DT_T_ACT} - (\text{DT_T}[p+1] * \text{QDT_T}[p+q]) \quad \text{(DPLL-3a)}$$

with

$$\text{QDT_T}[p+q] = \text{DT_T}[p+q] * \text{RDT_T}[p+q+1] \text{ for FST} = 0 \quad \text{(DPLL-2b3)}$$

$$\text{QDT_T}[p+q] = \text{DT_T}[p+q] * \text{RDT_T_FS1} \text{ for FST} = 1 \quad \text{(DPLL-2b4)}$$

and

FST has the meaning: NUTE = FULL_SCALE (see NUTC register)

QDT_T_ACT as well as QDT_Ti have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

Equation DPLL-4 to calculate the weighted average error

For SYT=1 calculate:

$$\text{MEDT_T} := (\text{EDT_T} + \text{MEDT_T})/2 \quad \text{(DPLL-4)}$$

Equation DPLL-5 to calculate the current increment value

$$\text{CDT_TX_nom} = (\text{DT_T_ACT} + \text{MEDT_T}) * \text{QDT_T}[p+q-1] \quad \text{(DPLL-5a1)}$$

with

$$\text{QDT_T}[p+q-1] = \text{DT_T}[p+q-1] * \text{RDT_T}[p+q] \quad (\text{for } q > 1) \quad \text{(DPLL-2c1)}$$

for q=1 use equation DPLL-2a1.

while

QDT_T_ACT as well as QDT_Ti have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

and the expected duration to the next TRIGGER event

$$\text{CDT_TX} = \text{CDT_TX_nom} * \text{SYN_T} \quad \text{(DPLL-5b)}$$

Note: In the case of an overflow (or negative values) in equations DPLL-5a1 or b set the value to 0xFFFFFFFF and the corresponding CTO bit in the DPLL_STATUS register. In case of negative values CDT_TX(_nom) to 0x0.

26.16.6.5 Increment prediction in Emergency Mode backwards (DIR2=1)
Equation DPLL-7a2 to calculate QDT_S_ACT backwards

$$\text{QDT_S_ACT} = \text{DT_S_ACT} * \text{RDT_S}[p+1] \quad \text{(DPLL-7a2)}$$

QDT_S_ACT as well as QDT_Si have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

Generic Timer Module (GTM)
Equation DPLL-8a to calculate the error of the last prediction

While $q = \text{NUSE-VSN}$, use only $\text{QDT_S}[p+q]$ and $\text{DT_S}[p+1]$ for the prediction of $\text{DT_S}[p]$

$$\text{EDT_S} = \text{DT_S_ACT} - (\text{DT_S}[p+1] * \text{QDT_S}[p+q]) \quad \text{(DPLL-8a)}$$

with

$$\text{QDT_S}[p-q] = \text{DT_S}[p+q] * \text{RDT_S}[p+q+1] \text{ for } \text{FSS} = 0 \quad \text{(DPLL-7b3)}$$

$$\text{QDT_S}[p-q] = \text{DT_S}[p+q] * \text{RDT_S_FS1} \text{ for } \text{FSS} = 1 \quad \text{(DPLL-7b4)}$$

and

FSS has the meaning: $\text{NUSE} = \text{FULL_SCALE}$ (see NUSC register)

QDT_S_ACT as well as QDT_Si have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

Equation DPLL-9 to calculate the weighted average error

For $\text{SYS}=1$ calculate:

$$\text{MEDT_S} := (\text{EDT_S} + \text{MEDT_S})/2 \quad \text{(DPLL-9)}$$

Equations DPLL-10 to calculate the current increment value

$$\text{CDT_SX_nom} = (\text{DT_S_ACT} + \text{MEDT_S}) * \text{QDT_S}[p+q-1] \quad \text{(DPLL-10a1)}$$

with

$$\text{QDT_S}[p+q-1] = \text{DT_S}[p+q-1] * \text{RDT_S}[p+q] \text{ (for } q > 1) \quad \text{(DPLL-7c1)}$$

for $q=1$ use equation DPLL-7a.

while

QDT_S_ACT as well as QDT_Si have a 24 bit value using a 6 bit integer part and a 18 bit fractional part.

Generic Timer Module (GTM)

and calculate the expected duration to the next STATE event

$$\text{CDT_SX} = \text{CDT_SX_nom} * \text{SYN_S} \quad \text{(DPLL-10b)}$$

Note: In the case of an overflow (or negative values) in equations DPLL-10a1 or b set the value to 0xFFFFFFFF and the corresponding CSO bit in the DPLL_STATUS register. In case of negative values CDT_TX(nom) to 0x0.

All 5 steps above (DPLL-6 to DPLL-10) are only needed in emergency mode. For the normal mode the calculations of equations DPLL-6 and DPLL-7 are done solely in order to get the values needed for a sudden switch to emergency mode.

26.16.7 Calculations for actions

As already shown for the calculation of the current interval by equations DPLL-1 to DPLL-10 for the prediction of actions a similar calculation is to be done, as shown by the equations DPLL-11. to DPLL-14. The calculation of actions is also needed when the DPLL is used for synchronous motor control applications (SMC=1, see DPLL_CTRL_1 register). For action prediction purposes the measured time periods of the past (one FULL_SCALE back, when the corresponding NUTE or NUSE values are set properly by the CPU) are used. The calculation can be explained by the following assumptions, which are considerably simple:

Take the corresponding increments for prediction in the past and put the sum of it in relation to the increment (DT_T[k], DT_S[k], with $k > 0$, which is represented by the time stamp difference) which is exactly one FULL_SCALE period in the past (DPLL-11 or DPLL-13 respectively). Make a prediction for the coming sum of increments using the current measured increment (DT_T_ACT or DT_S_ACT respectively, that means DPLL-1 or DPLL-6 respectively) and add a weighted average error (DPLL-3 and DPLL-4 or DPLL-8 and DPLL-9 respectively, calculated for one increment prediction) before multiplication with the relation of equation DPLL-11 or DPLL-13 respectively in order to get the result as described by equations DPLL-12 or DPLL-14 respectively.

In order to avoid division operations instead of the increment (DT_T[k], DT_S[k] with $k > 0$) in the past its reciprocal value (RDT_T[k], RDT_S[k] with $k > 0$) is used, which is stored also in RAM.

For the calculation of actions perform always a new refined calculation as long as the resulting time stamp is not in the past. In the other case the tsac/psac values (time/position stamp of action calculated) is set to the time/position stamp of the last input event (TRIGGER/STATE), the ACT_Ni bit in the DPLL_ACT_STA register is reset, while the corresponding ACT_Ni bit in the DPLL_ACT_STA_shadow register is set. Each new PMTR_i value will set this ACT_Ni bit again and reset the correspondent shadow bit until a new calculation is performed.

Generic Timer Module (GTM)**Action updates at highest speed**

Up to 32 action values can be calculated. For the shortest increment durations (23,4 μ s) not all of them can be updated with each valid input event. Please notice the following conditions and parameters for an estimation of possible results.

All time estimation values are given for a system clock frequency of 100 MHz and the assumption, that the calculation of the DPLL is not impeded by a remote read or write access to the DPLL RAMs. Each RAM access is to be considered by an additional delay of about 40 ns (tremote_RAM_access, to be precised later). When using a different system clock frequency the calculation duration is extended accordingly.

1.) Typical time needed for basic operations (RAM update, pointer calculation and SUB_INC generation for normal, emergency mode or one PMSM:

$$t_{basic_0} = 9,9 \mu s$$

2.) Typical time needed basic operations (RAM update, pointer calculation and SUB_INC generation for two PMSMs:

$$t_{basic_1} = 11,0 \mu s$$

3.) Typical time needed to calculate one action

$$t_{action_i} = 3,7 \mu s$$

Please notice that the above mentioned values are observed worst case values, when the two state machines of TRIGGER and STATE are both in operation.

These values allow the calculation of at least 2 action values for each input event for all specified increment durations. The complete time needed for the basic operation, n action calculations and k remote RAM access operations can be calculated as follows

$$t_{complete} = t_{basic_0}/1 + n*t_{action_i} + k*t_{remote_RAM_access}.$$

Typical applications

Normal and emergency mode

For a typical application with the shortest increment duration of 100 μ s in normal or emergency mode the calculation of up to 24 action values can be performed for each valid input event.

One PMSM

For one PMSM and a typical shortest increment time of 39 μ s there is the calculation of up to 7 action values possible for each input event.

Two PMSMs with restricted action calculations

When only one PMSM uses the action calculation service an the shortest increment duration is 39 μ s, there can up to 7 actions served for each valid input event.

Two PMSMs with unrestricted action calculations

Generic Timer Module (GTM)

When 2 PMSMs are used and both use the action calculation service at a minimal increment duration of 39 μ s there are up to 7 action calculations possible for each of the two engines - that means up to 14 action calculations per increment in average.

26.16.7.1 Action calculations for TRIGGER forwards

valid for RMO=0 or for SMC=1 with

p =APT_2B, t =APT, m =NA[i] (part w), mb =NA[i](part b)/1024, NUTE-VTN= q , NUTE= n

Equation DPLL-11a1 to calculate the time prediction for an action

For DIR1=0 and $q>m$ calculate:

$$PDT_T[i] = (TSF_T[p+m-n] - TSF_T[p-n] + mb * DT_Tx[t-q+1]) * RDT_T[t-q] \quad \text{(DPLL-11a1)}$$

with

$$DT_Tx[t-q+1] = DT_T[t-q+1] \text{ for } TS0_HRT=0 \quad \text{(DPLL-11b2)}$$

or

$$DT_Tx[t-q+1] = DT_T[t-q+1]/8 \text{ for } TS0_HRT=1 \quad \text{(DPLL-11b3)}$$

and while the multiplication with mb does mean the fractional part of NA[i].

For SMC=0 and RMO=0 calculate for DIR1=0 all 32 actions in forward direction, if requested; in the case SMC=1 calculate up to 16 actions 0 to 15 in dependence of the TRIGGER input.

Equation DPLL-11a2 to calculate the time prediction for an action

For SYT=1, NUTE= 2*(TNU+1), $q>m$ and DIR1=0 equation DPLL-11a2 is equal to

$$PDT_T[i] = (TSF_T[p+m] - TSF_T[p] + mb * DT_Tx[t-q+1]) * RDT_T[t] \quad \text{(DPLL-11a2)}$$

with

$$DT_Tx[t-q+1] = DT_T[t-q+1] \text{ for } TS0_HRT=0 \quad \text{(DPLL-11b2)}$$

or

$$DT_Tx[t-q+1] = DT_T[t-q+1]/8 \text{ for } TS0_HRT=1 \quad \text{(DPLL-11b3)}$$

Generic Timer Module (GTM)
Equation DPLL-11b to calculate the time prediction for an action

for DIR1=0, NUTE-VTN=q, q (< or =) m, n>1 and t=APT:

$$PDT_T[i] = (m+mb)*DT_Tx[t-q+1] * RDT_T[t-q] \quad \text{(DPLL-11b)}$$

with

$$DT_Tx[t-q+1] = DT_T[t-q+1] \text{ for } TS0_HRT=0 \quad \text{(DPLL-11b2)}$$

or

$$DT_Tx[t-q+1] = DT_T[t-q+1]/8 \text{ for } TS0_HRT=1 \quad \text{(DPLL-11b3)}$$

Note: Note: Make the calculations above before updating the TSF_T[i] values according to equations DPLL-1c3 ff.

Equation DPLL-11c to calculate the time prediction for an action

for n=1 (this is always valid for SYT=0)

$$PDT_T[i] = (m+mb)* DT_T_ax * RDT_T[t-1] \quad \text{(DPLL-11c)}$$

with

$$DT_T_ax = DT_T_ACT \text{ for } TS0_HRT=0 \quad \text{(DPLL-1a4a)}$$

or

$$DT_T_ax = DT_T_ACT/8 \text{ for } TS0_HRT=1 \quad \text{(DPLL-1a4b)}$$

Note: For the relevant last increment add the fractional part of DT_T_ACT as described in NA[i].

Equation DPLL-12 to calculate the duration value until action

$$DTA[i] = (DT_T_ACT + MEDT_T)* PDT_T[i] \quad \text{(DPLL-12)}$$

Note: All 5 steps in equations DPLL-11 to DPLL-12 are only calculated in normal mode.

26.16.7.2 Action calculations for TRIGGER backwards

valid for RMO=0 or for SMC=1

$p=APS_1C2$, $t=APS$, $m=Nai(part\ w)$ $mb=Nai(part\ b)/1024$, $NUSE-VSN = q \geq m$ and $NUSE=n$

For SMC=0 and RMO=0 calculate for DIR1=1 all 32 actions in backward direction for special purposes; in the case SMC=1 calculate up to 16 actions 0 to 15 in dependence of the TRIGGER input.

Equation DPLL-11a3 to calculate the time prediction for an action

For DIR1= 1 and $q>m$ calculate

$$PDT_T[i] = (TSF_T[p-m+n] - TSF_T[p+n] + mb * DT_Tx[t+q+1]) * RDT_T[t+q] \quad \text{(DPLL-11a3)}$$

with

$$DT_Tx[t+q+1] = DT_T[t+q+1] \text{ for } TS0_HRT=0 \quad \text{(DPLL-11b4)}$$

or

$$DT_Tx[t+q+1] = DT_T[t+q+1]/8 \text{ for } TS0_HRT=1 \quad \text{(DPLL-11b5)}$$

Equation DPLL-11a4 to calculate the time prediction for an action

For SYT=1 and $NUTE = 2*(TNU+1)$, $q>m$, $VTN=2*SYN_NT$ and hence $NUTE-VTN = 2*(TNU+1-SYN_NT)$ for DIR1=1 this is equal to

$$PDT_T[i] = (TSF_T[p-m] - TSF_T[p] + mb * DT_Tx[t+q+1]) * RDT_T[t] \quad \text{(DPLL-11a4)}$$

with

$$DT_Tx[t+q+1] = DT_T[t+q+1] \text{ for } TS0_HRT=0 \quad \text{(DPLL-11b4)}$$

or

$$DT_Tx[t+q+1] = DT_T[t+q+1]/8 \text{ for } TS0_HRT=1 \quad \text{(DPLL-11b5)}$$

Note: Make the calculations above before updating the $TSF_T[i]$ values according to equations DPLL-1c3 ff.

Generic Timer Module (GTM)
Equation DPLL-11b1 to calculate the time prediction for an action

For NUTE-VTN =q, q (< or =) m the following equation is valid for n>1 and t=APT:

$$PDT_T[i] = (m+mb)*DT_Tx[t+q-1] * RDT_T[t+q] \quad \text{(DPLL-11b1)}$$

with

$$DT_Tx[t+q-1] = DT_T[t+q-1] \text{ for } TS0_HRT=0 \quad \text{(DPLL-11b4)}$$

or

$$DT_Tx[t+q-1] = DT_T[t+q-1]/8 \text{ for } TS0_HRT=1 \quad \text{(DPLL-11b5)}$$

Equation DPLL-11c1 to calculate the time prediction for an action

for n=1 (this is always valid for SYT=0)

$$PDT_T[i] = (m+mb)* DT_T_ax * RDT_T[t+1] \quad \text{(DPLL-11c1)}$$

with

$$DT_T_ax = DT_T_ACT \text{ for } TS0_HRT=0 \quad \text{(DPLL-1a4a)}$$

or

$$DT_T_ax = DT_T_ACT/8 \text{ for } TS0_HRT=1 \quad \text{(DPLL-1a4b)}$$

Note: For the relevant last increment add the fractional part of DT_T_ACT as described in NA[i].

Equation DPLL-12 to calculate the duration value for an action

$$DTA[i] = (DT_T_ACT + MEDT_T)* PDT_T[i] \quad \text{(DPLL-12)}$$

Use the results of equations DPLL-1a, b, DPLL-3 and DPLL-4 for the above calculation

Note: Note: All 5 steps in equations DPLL-11 to DPLL-12 are only calculated in normal mode or when SMC=1.

26.16.7.3 Action calculations STATE forwards

valid for RMO=1

$p=APS_1C2$, $t=APS$, $m=NA[i](part\ w)$ $mb=NA[i9](part\ b)/1024$, $NUSE-VSN = q$ and $NUSE=n>m$

For SMC=0 and RMO=1 calculate for DIR2=0 all 32 actions in forward direction, if requested; in the case SMC=1 and RMO=1 calculate up to 16 actions 16 to 31 in dependence of the STATE input.

Equation DPLL-13a1 to calculate the time prediction for an action

For DIR2=0 and $q>m$ calculate:

$$PDT_S[i] = (TSF_S[p+m-n] - TSF_S[p-n] + mb * DT_Sx[t-q+1]) * RDT_S[t-q] \quad \text{(DPLL-13a1)}$$

with

$$DT_Sx[t-q+1] = DT_S[t-q+1] \text{ for } TS0_HRS=0 \quad \text{(DPLL-13b2)}$$

or

$$DT_Sx[t-q+1] = DT_S[t-q+1]/8 \text{ for } TS0_HRS=1 \quad \text{(DPLL-13b3)}$$

Equation DPLL-13a2 to calculate the time prediction for an action

For SYS=1 and $NUSE=2*(SNU+1)$, $q>m$, $SYSF=0$, $VSN=2*SYN_NS$ and hence $NUSE-VSN = 2*(SNU+1-SYN_NS)$ equation DPLL-13a1 is equal to

$$PDT_S[i] = (TSF_S[p+m] - TSF_S[p] + mb * DT_Sx[t-q+1]) * RDT_S[t] \quad \text{(DPLL-13a2)}$$

with

$$DT_Sx[t-q+1] = DT_S[t-q+1] \text{ for } TS0_HRS=0 \quad \text{(DPLL-13b2)}$$

or

$$DT_Sx[t-q+1] = DT_S[t-q+1]/8 \text{ for } TS0_HRS=1 \quad \text{(DPLL-13b3)}$$

Equation DPLL-13b to calculate the time prediction for an action

For $NUSE-VTN = q$, $q (< \text{ or } =) m$ and $n>1$:

Generic Timer Module (GTM)

$$PDT_S[i] = (m+mb)*DT_Sx[t-q+1] * RDT_S[t-q] \quad \text{(DPLL-13b)}$$

with

$$DT_Sx[t-q+1] = DT_S[t-q+1] \text{ for } TS0_HRS=0 \quad \text{(DPLL-13b2)}$$

or

$$DT_Sx[t-q+1] = DT_S[t-q+1]/8 \text{ for } TS0_HRS=1 \quad \text{(DPLL-13b3)}$$

Equation DPLL-13c to calculate the time prediction for an action

for n=1

$$PDT_S[i] = (m+mb)* DT_S_ax * RDT_S[t-1] \quad \text{(DPLL-13c)}$$

with

$$DT_S_ax = DT_S_ACT \text{ for } TS0_HRS=0 \quad \text{(DPLL-6a4a)}$$

or

$$DT_S_ax = DT_S_ACT/8 \text{ for } TS0_HRS=1 \quad \text{(DPLL-6a4b)}$$

Equation DPLL-14 to calculate the duration value for an action

$$DTA[i] = (DT_S_ACT + MEDT_S)* PDT_S[i] \quad \text{(DPLL-14)}$$

Use the results of DPLL-7, DPLL-8 and DPLL-9 for the above calculation

Note: All 5 steps of equations DPLL-13 to DPLL-14 are only calculated in emergency mode or for SMC=1 in combination with RMO=1.

26.16.7.4 Action calculations for STATE backwards

valid for RMO=1 with

Generic Timer Module (GTM)

$p=APS_1C2$, $t=APS$, $m=NA[i](part\ w)$ $mb=NA[i](part\ b)/1024$, $NUSE-VSN = q$ and $NUSE=n$

For $SMC=0$ and $RMO=1$ calculate for $DIR1=1$ all 32 actions in backwards mode for special purposes; in the case $SMC=1$ and $RMO=1$ calculate up to 16 actions 16 to 31 in dependence of the STATE input.

Equation DPLL-13a3 to calculate the time prediction for an action

For ($DIR2= 1$ ($SMC=1$) or $DIR1=1$ ($SMC=0$)) and $q>m$ calculate

$$PDT_S[i] = (TSF_S[p-m+n] - TSF_S[p+n] + mb * DT_Sx[p+q-1]) * RDT_S[t+q] \quad \text{(DPLL-13a3)}$$

with

$$DT_Sx[p+q-1] = DT_S[p+q-1] \text{ for } TS0_HRS=0 \quad \text{(DPLL-13B4)}$$

or

$$DT_Sx[p+q-1] = DT_S[p+q-1]/8 \text{ for } TS0_HRS=1 \quad \text{(DPLL-13b5)}$$

Equation DPLL-13a4 to calculate the time prediction for an action

For $SYS=1$, $NUSE=2*(SNU+1)$, $q>m$, $SYSF=0$, $VSN=2*SYN_NS$ and hence $NUSE-VSN = 2*(SNU+1-SYN_NS)$ equation DPLL-13a3 is equal to

$$PDT_S[i] = (TSF_S[p-m] - TSF_S[p] + mb * DT_Sx[p+q-1]) * RDT_S[t] \quad \text{(DPLL-13a4)}$$

with

$$DT_Sx[p+q-1] = DT_S[p+q-1] \text{ for } TS0_HRS=0 \quad \text{(DPLL-13b4)}$$

or

$$DT_Sx[p+q-1] = DT_S[p+q-1]/8 \text{ for } TS0_HRS=1 \quad \text{(DPLL-13b5)}$$

Equation DPLL-13b1 to calculate the time prediction for an action

For $NUSE-VSN =q$, $q (< \text{ or } 0)$ $NUSE=n$ and $n>1$:

$$PDT_S[i] = m * DT_Sx[t+q-1] * RDT_S[t+q] \quad \text{(DPLL-13b1)}$$

with

Generic Timer Module (GTM)

$$DT_Sx[p+q-1] = DT_S[t+q-1] \text{ for } TS0_HRS=0 \quad \text{(DPLL-13b4)}$$

or

$$DT_Sx[p+q-1] = DT_S[t+q-1]/8 \text{ for } TS0_HRS=1 \quad \text{(DPLL-13b5)}$$

Equation DPLL-13c1 to calculate the time prediction for an action

for n=1

$$PDT_S[i] = (m+mb) * DT_S_ax * RDT_S[t+1] \quad \text{(DPLL-13c1)}$$

with

$$DT_S_ax = DT_S_ACT \text{ for } TS0_HRS=0 \quad \text{(DPLL-6a4a)}$$

or

$$DT_S_ax = DT_S_ACT/8 \text{ for } TS0_HRS=1 \quad \text{(DPLL-6a4b)}$$

Equation DPLL-14 to calculate the duration value until action

$$DTA[i] = (DT_S_ACT + MEDT_S) * PDT_S[i] \quad \text{(DPLL-14)}$$

Use the results of DPLL-7, DPLL-8 and DPLL-9 for the above calculation

Note: All 5 steps of equations DPLL-13 to DPLL-14 are only calculated in emergency mode or for SMC=1 in combination with RMO=1.

26.16.7.5 Update of RAM in Normal and Emergency Mode

After considering the calculations for up to all 24 actions according to equations (DPLL-11, DPLL-12), only when going back to state 1 or 21 (because of a new TRIGGER or STATE event, that means when no further PMTR values are to be considered) set time stamp values and durations of increments in the RAM.

Equation DPLL-1a4 to update the time stamp values for TRIGGER

Generic Timer Module (GTM)

TSF_T[s]=TS_Tx using the following equations for the determination of TS_Tx	(DPLL-1a4)
For TS0_HRT=0:	
TS_Tx=TS_T	(DPLL-1a4w)
DT_T_ax = DT_T_ACT	(DPLL-1a4a)
For TS0_HRT=1:	
TS_Tx(20:0)=TS_T/8	(DPLL-1a4x)
TS_Tx(23:21)=TBU_TS0_T(23:21)	(DPLL-1a4y)
for TBU_TS0_T(20:0) > or = TS_Tx(20:0)	
TS_Tx(23:21)=TBU_TS0_T(23:21) -1	(DPLL-1a4z)
for TBU_TS0_T(20:0) < TS_Tx(20:0)	
DT_T_ax = DT_T_ACT/8	(DPLL-1a4b)

Note: The combination of values LOW_RES=0 and TS0_HRT=1 is not possible.

Store the time stamp values in the time stamp field according to the address pointer APT_2B=s, but make this update only after the calculation of actions **Section 26.16.7** because the old TSF_T[i] values are still needed for these calculations. Please note that the address pointer after a gap is still increment by SYN_T_OLD in that case (see state machine step 1 in **Section 26.16.8.6**).

Equation DPLL-1a5-7 to extend the time stamp values for TRIGGER in forward direction

when SYT=1 and SYN_T_OLD=r>1 and DIR1=0

TSF_T[s-1] = TSF_T[s] -DT_T_ax	(DPLL-1a5)
TSF_T[s-2] = TSF_T[s-1] -DT_T_ax	(DPLL-1a6)

Generic Timer Module (GTM)

until

$$TSF_T[s-r+1] = TSF_T[s-r+2] - DT_T_ax \quad (DPLL-1a7)$$

after the incrementation of the pointer APT_2B by SYN_T_OLD

Equations DPLL-1a5-7 for backward direction

when SYT=1 and SYN_T_OLD=r>1 and DIR1=1

$$TSF_T[s+1] = TSF_T[s] - DT_T_ax \quad (DPLL-1a5)$$

$$TSF_T[s+2] = TSF_T[s+1] - DT_T_ax \quad (DPLL-1a6)$$

until

$$TSF_T[s+r-1] = TSF_T[s+r-2] - DT_T_ax \quad (DPLL-1a7)$$

after the decrementation of the pointer APT_2B by SYN_T_OLD

Equations DPLL-1b1 and DPLL-1c1 to update the RAM after calculation

$$DT_T[p] = DT_T_ACT \quad (DPLL-1b1)$$

save old reciprocal value from RAM before overwriting:

$$RDT_T_FS1 = RDT_T[p] \quad (DPLL-1c1)$$

after that store new value in RAM

$$RDT_T[p] = RDT_T_ACT \quad (DPLL-1c2)$$

Store increment duration and reciprocal value in RAM region 2 in normal mode after calculation of actions only when a new valid TRIGGER slope is detected and in emergency mode directly after calculation of DT_T_ACT or RDT_T_ACT respectively.

Equation DPLL-6a4 to update the time stamp values for STATE

Generic Timer Module (GTM)

TSF_S[s]=TS_Sx using the following equations for the determination of TS_Sx	(DPLL-6a4)
For TS0_HRS=0: TS_Sx=TS_S	(DPLL-6a4)
DT_S_ax = DT_S_ACT	(DPLL-6a4a)
For TS0_HRS=1: TS_Sx(20:0)=TS_S/8	(DPLL-6a4x)
TS_Sx(23:21)=TBU_TS0_S(23:21)	(DPLL-6a4y)
for TBU_TS0_S(20:0) > or = TS_Sx(20:0)	
TS_Sx(23:21)=TBU_TS0_S(23:21) -1	(DPLL-6a4z)
for TBU_TS0_S(20:0) < TS_Sx(20:0)	
DT_S_ax = DT_S_ACT/8	(DPLL-6a4b)

Note: The combination of values LOW_RES=0 and TS0_HRS=1 is not possible.

Store the time stamp value in the time stamp field according to the address pointer APS_1C2=s, but make this update only after the calculation of actions (equations DPLL-13a2 or DPLL-13a4, if applicable) because the old TSF_S[i] values are still needed for these calculations. Please note, that the address pointer after a gap is still increment by SYN_S_OLD in that case (see state machine step 21 in [Section 26.16.8.6](#)).

Equations DPLL-6a5-7 to extend the time stamp values for STATE

When SYS=1 and SYN_S_OLD=r>1 and DIR1=0 respectively calculate

TSF_S[s-1] = TSF_S[s] - DT_S_ax	(DPLL-6a5)
TSF_S[s-2] = TSF_S[s-1] - DT_S_ax	(DPLL-6a6)
until	
TSF_S[s- r+1] = TSF_S[s- r+2] - DT_S_ax	(DPLL-6a7)

Generic Timer Module (GTM)

after incrementation of the pointer APS_2B by SYN_S_OLD

Equations DPLL-6a5-7 for backward direction

When SYS=1 and SYN_S_OLD=r>1 and DIR1=1 respectively calculate

$$\text{TSF_S}[s+1] = \text{TSF_S}[s] - \text{DT_S_ax} \quad \text{(DPLL-6a5)}$$

$$\text{TSF_S}[s+2] = \text{TSF_S}[s+1] - \text{DT_S_ax} \quad \text{(DPLL-6a6)}$$

until

$$\text{TSF_S}[s+r-1] = \text{TSF_S}[s+r-2] - \text{DT_S_ax} \quad \text{(DPLL-6a7)}$$

after the incrementation of the pointer APS_1C2 by SYN_S_OLD

Equations DPLL-6b1 and DPLL-6c1 to update the RAM after calculation

$$\text{DT_S}[p] = \text{DT_S_ACT} \quad \text{(DPLL-6b1)}$$

save old reciprocal value from RAM before overwriting:

$$\text{RDT_S_FS1} = \text{RDT_S}[p] \quad \text{(DPLL-6c1)}$$

after that store new value in RAM

$$\text{RDT_S}[p] = \text{RDT_S_ACT} \quad \text{(DPLL-6c2)}$$

when a new valid STATE slope is detected in emergency mode or in normal mode (SMC=RMO=0) directly after calculation of the values above.

Store increment duration and reciprocal value in RAM region 1C in emergency mode after calculation of actions only when a new valid STATE slope is detected and in normal mode directly after calculation of DT_S_ACT or RDT_S_ACT respectively.

26.16.7.6 Time and position stamps for actions in Normal Mode

Equation DPLL-15 to calculate the action time stamp

$$TSAC[i] = DTA[i] - DLA[i] + TS_Tx \text{ (for } DTA[i] > DLA[i] \text{ and } DTA[i] - DLA[i] < 0x800000) \quad \text{(DPLL-15a)}$$

$$TSAC[i] = TS_Tx \text{ (for } DTA[i] < DLA[i]) \quad \text{(DPLL-15b)}$$

$$TSAC[i] = 0x7FFFFFFF + TS_Tx \text{ (for } DTA[i] > DLA[i] \text{ and } DTA[i] - DLA[i] > 0x7FFFFFFF) \quad \text{(DPLL-15c)}$$

Note: For TS_Tx see equations (DPLL-1a4 and following).

The calculation is done after the calculation of the current expected duration value according to equation DPLL-12 at [Section](#). The time stamp of the action can be calculated as shown above in equation DPLL-15 using the delay value of the action and the current time stamp.

Equations DPLL-17 to calculate the position stamp forwards

for DIR1=0 and TS0_HRT=0:

$$PSAC[i] = PSA[i] - (DLA[i] * RCDT_TX_NOM) * (MLT + 1) \quad \text{(DPLL-17)}$$

with

$$RCDT_TX_NOM = RCDT_TX * SYN_T \quad \text{(DPLL-17a)}$$

and

$$RCDT_TX = 1 / CDT_TX \quad \text{(DPLL-17b)}$$

for DIR1=0 and TS0_HRT=1:

$$PSAC[i] = PSA[i] - (8 * DLA[i] * RCDT_TX_NOM) * (MLT + 1) \quad \text{(DPLL-17d)}$$

with

$$RCDT_TX_NOM = RCDT_TX * SYN_T \quad \text{(DPLL-17a)}$$

and

$$\text{RCDT_TX} = 1/\text{CDT_TX} \quad \text{(DPLL-17b)}$$

replace (MLT+1) in equations (DPLL-17) and (DPLL-17d) by MLS1 for SMC=1
 use the calculated value of (DPLL-17b) also for the generation of SUB_INCi
 and serve the action by transmission of TSAC[i] and PSAC[i] to ACT_D_i

The action is to be updated for each new TRIGGER event until the calculated time stamp is in the past. In this case use the time stamp of the last input event instead of the calculated value and the calculated position stamp of the actual increment as target position value. Set the corresponding shadow bit in the DPLL_ACT_STA register. Because of the blocking read operation the ACT_D values can be read only once.

Equations DPLL-17 to calculate the position stamp backwards

For DIR1=1 and TS0_HRT=0:

$$\text{PSAC}[i] = \text{PSA}[i] + (\text{DLA}[i] * \text{RCDT_TX_NOM}) * (\text{MLT} + 1) \quad \text{(DPLL-17c)}$$

with

$$\text{RCDT_TX_NOM} = \text{RCDT_TX} * \text{SYN_T} \quad \text{(DPLL-17a)}$$

and

$$\text{RCDT_TX} = 1/\text{CDT_TX} \quad \text{(DPLL-17b)}$$

For DIR1=1 and TS0_HRT=1:

$$\text{PSAC}[i] = \text{PSA}[i] + (8 * \text{DLA}[i] * \text{RCDT_TX_NOM}) * (\text{MLT} + 1) \quad \text{(DPLL-17e)}$$

with

$$\text{RCDT_TX_NOM} = \text{RCDT_TX} * \text{SYN_T} \quad \text{(DPLL-17a)}$$

and

$$\text{RCDT_TX} = 1/\text{CDT_TX} \quad \text{(DPLL-17b)}$$

replace (MLT+1) in equations (DPLL-17c) and (DPLL-17e) by MLS1 for SMC=1

Generic Timer Module (GTM)

use the calculated value of (DPLL-17b) also for the generation of SUB_INCI and serve the action by transmission of TSAC[i] and PSAC[i] to ACT_D_i

The action is to be updated for each new TRIGGER event until the calculated time stamp is in the past. In this case use the time stamp of the last input event instead of the calculated value and the calculated position stamp of the actual increment as target position value. Set the corresponding shadow bit in the DPLL_ACT_STA register. Because of the blocking read operation the ACT_D values can be read only once.

26.16.7.7 The use of the RAM

The RAM is used to store the data of the last FULL_SCALE period. The use of single port RAMs is recommended. The data width of the RAM is usual 3 bytes, but could be extended to 4 bytes in future applications. There are 3 different RAMs, each with separate access ports. the RAM 1A is used to store the position minus time requests, got from the ARU. No CPU access is possible to this RAM during operation (when the DPLL is enabled).

Ram 1B is used for configuration parameters and variables needed for calculations. Within RAM 1C the values of the STATE events are stored. RAM 1B and RAM 1C do have a common access port and are also marked as RAM 1BC in order to clarify this fact.

RAM 2 is used for values of the TRIGGER events.

Because of the access of the DPLL internal state machine at the one side and the CPU at the other side the access priority has to be controlled for both RAMs 1BC and 2. The access priority is defined as stated below. The CPU access procedure via AE-interface goes in a wait state (waiting for data valid) while it needs a colliding RAM access during serving a corresponding state machine RAM access. In order not to provoke unexpected behaviour of the algorithms the writing of the CPU to the RAM regions 1B, 1C or 2 will be monitored and results in interrupt requests when enabled.

CPU access is specified at follows:

1. CPU has highest priority for a single read/write access. The DPLL algorithm is stalled during external bus RAM accesses.
2. After serving the CPU access to the RAM the DPLL gets the highest RAM access priority for 8 clock cycles. Afterwards continue with 1.

The RAM address space has to be implemented in the address space of the CPU.

26.16.7.8 Time and position stamps for actions in Emergency Mode

Equation DPLL-18 to calculate the action time stamp

Generic Timer Module (GTM)

TSAC[i]= DTA[i] - DLA[i] + TS_Sx (for DTA[i] > DLA[i] and DTA[i] - DLA[i] < 0x800000) **(DPLL-18a)**

TSAC[i]= TS_Sx (for DTA[i] < DLA[i]) **(DPLL-18b)**

TSAC[i]= 0x7FFFFFF + TS_Sx (for DTA[i] > DLA[i] and DTA[i] - DLA[i] > 0x7FFFFFF) **(DPLL-18c)**

Note: For TS_Sx see equations (DPLL-6a4 and following), [Section](#).

The calculation is done after the calculation of the current expected duration value according to equation DPLL-14 at [Section](#). The time stamp of the action can be calculated as shown in equation DPLL-18 using the delay value of the action and the current time stamp.

Equations DPLL-20 to calculate the position stamp forwards

for DIR1=0 respectively and TS0_HRS=0:

PSAC[i] = PSA[i] - (DLA[i]*RCDT_SX_NOM)*MLS1 **(DPLL-20)**

with

RCDT_SX_NOM= RCDT_SX * SYN_S **(DPLL-20a)**

and

RCDT_SX= 1/CDT_SX **(DPLL-20b)**

for DIR2=0 or DIR1=0 respectively and TS0_HRS=1:

PSAC[i] = PSA[i] - (8*DLA[i]*RCDT_SX_NOM)*MLS1 **(DPLL-20d)**

with

RCDT_SX_NOM= RCDT_SX * SYN_S **(DPLL-20a)**

and

RCDT_SX= 1/CDT_SX **(DPLL-20b)**

Generic Timer Module (GTM)

replace MLS1 in equations (DPLL-20) and (DPLL-20d) by MLS2 for (SMC=1 and RMO=1)

use the calculated value of (DPLL-20b) also for the generation of SUB_INCi.

and serve the action by transmission of TSAC[i] and PSAC[i] to ACT_D.

The action is to be updated for each new STATE event until the event is in the past. In this case use the time stamp of the last input event instead of the calculated value and the calculated position stamp of the actual increment as target position value. Set the corresponding shadow bit in the DPLL_ACT_STA register. Because of the blocking read operation the ACT_D values can be read only once.

Equations DPLL-20 to calculate the position stamp backwards

For DIR2=1 or DIR1=1 respectively and TS0_HRS=0:

$$PSAC[i] = PSA[i] + (DLA[i]*RCDT_SX_NOM)*MLS1 \quad \text{(DPLL-20c)}$$

with

$$RCDT_SX_NOM = RCDT_SX * SYN_S \quad \text{(DPLL-20a)}$$

and

$$RCDT_SX = 1/CDT_SX \quad \text{(DPLL-20b)}$$

For DIR2=1 or DIR1=1 respectively and TS0_HRS=1:

$$PSAC[i] = PSA[i] + (8*DLA[i]*RCDT_SX_NOM)*MLS1 \quad \text{(DPLL-20e)}$$

with

$$RCDT_SX_NOM = RCDT_SX * SYN_S \quad \text{(DPLL-20a)}$$

and

$$RCDT_SX = 1/CDT_SX \quad \text{(DPLL-20b)}$$

replace MLS1 in equations (DPLL-20c) and (DPLL-20e) by MLS2 for (SMC=1 and RMO=1)

use the calculated value of (DPLL-20b) also for the generation of SUB_INCi.

Generic Timer Module (GTM)

and serve the action by transmission of TSAC[i] and PSAC[i] to ACT_D.

The action is to be updated for each new STATE event until the event is in the past. In this case use the time stamp of the last input event instead of the calculated value and the calculated position stamp of the actual increment as target position value. Set the corresponding shadow bit in the DPLL_ACT_STA register. Because of the blocking read operation the ACT_D values can be read only once.

26.16.8 Signal processing

26.16.8.1 Time stamp processing

Signal processing does mean the computation of the time stamps in order to calculate at which time the outputs have to appear. For such purposes the time stamp values have to be stored in the RAM and by calculating the difference between old and new values the duration of the last time interval is determined simply. This difference should be also stored in the RAM in order to see the changes between the intervals by changing the conditions and the speed of the observed process.

26.16.8.2 Count and compare unit

The count and compare unit processes all input signals taking into account the configuration values. It uses a state machine and provides the output signals as described above.

26.16.8.3 Sub pulse generation for SMC=0

Equation DPLL-21 to calculate the number of pulses to be sent in normal mode using the automatic end mode condition

For RMO=0, SMC=0 and DMO=0

$$\text{NMB_T} = (\text{MLT}+1) * \text{SYN_T} + \text{MP} + \text{PD_store} + \text{MPVAL1} \quad \text{(DPLL-21)}$$

with

$$\text{PD_store} = \text{ADT_T}[12:0], \text{ prefetched during last increment} \quad \text{(DPLL-21a)}$$

$$\text{SYN_T} = \text{ADT_T}[18:16], \text{ prefetched during last increment}$$

$$\text{MPVAL1} = \text{pulse correction value for PCM1_SHADOW_TRIGGER=1}$$

Generic Timer Module (GTM)

while the value for PD_store is zero for AMT=0

and

the value of MP is zero for COA=0

In order to get a higher resolution for higher speed a generator for the sub-pulses is chosen using an adder. All missing pulses MP are considered using equation DPLL-21 and are determined by counting the number of pulses of the last increment. The value SYN_T is stored from the last increment using NT of the ADT_T[i] value at RAM region 2C.

Equations DPLL-22-24 to calculate the number of pulses to be sent in emergency mode using the automatic end mode condition for SMC=0

For RMO=1, SMC=0 and DMO=0;

the value for PD_S_store is zero for AMS=0

$$\text{NMB_S} = \text{MLS1} * \text{SYN_S} + \text{MP} + \text{PD_S_store} \quad (\text{DPLL-22})$$

with

$$\text{MLS1} = (\text{MLT} + 1) * (\text{TNU} + 1) / (\text{SNU} + 1) \quad (\text{DPLL-23})$$

PD_S_store = ADT_S[15:0], prefetched during last increment

SYN_S = ADT_S[21:16], prefetched during last increment

MPVAL1 = pulse correction value for PCM1_SHADOW_STATE=1

while the value for PD_S_store is zero for AMS=0

and

the value of MP is zero for COA=0

Please note, that these calculations above in equations DPLL-21 and DPLL-22 are only valid for an automatic end mode (DMO = 0).

For calculation of the number of generated pulses a value of 0.5 is added as shown in equations DPLL-25 or DPLL-26 respectively in order to compensate rounding down errors at the succeeding arithmetic operations. Because in automatic end mode the number of pulses is limited by INC_CNT1 it is guaranteed, that not more pulses as needed are generated and in the same way missing pulses are caught up for the next increment.

Equation DPLL-25 to calculate ADD_IN in normal mode for SMC=0

In normal mode (for RMO=0) calculate in the case LOW_RES=TS0_HRT

Generic Timer Module (GTM)

$$\text{ADD_IN_CALN} = (\text{NMB_T} + 0.5) * \text{RCDT_TX} \quad (\text{DPLL-25})$$

with

RCDT_TX is the 2^{32} time value of the quotient in equation DPLL-17b

In normal mode (for RMO=0) calculate in the case LOW_RES=1 and TS0_HRT=0

$$\text{ADD_IN_CALN} = (\text{NMB_T} + 0.5) * (\text{RCDT_TX} / 8) \quad (\text{DPLL-25a})$$

with

RCDT_TX is the 2^{32} time value of the quotient in equation DPLL-17b

For RMO=0 and SMC=0:

$$\text{ADD_IN_CAL1} = \text{ADD_IN_CALN} \quad (\text{DPLL-25b})$$

LOW_RES=0 and TS0_HRT=1 is not possible. For such a configuration the RCT bit in the DPLL_STATUS register is set together with the ERR bit.

In the automatic end mode (DMO = 0) missing pulses should be sent to the input RPCUx (rapid pulse catch up on) in **Adder for generation of SUB_INCx by the carry Cout**, to be caught up on with CMU_CLK0 (for COA = 0).

When normal and rapid pulses are generated simultaneously, the SUB_INCx frequency is doubled at this moment in order to count two pulses at the TBU_CHx_BASE register. In order to make the frequency doubling possible, the CMU_CLK0 should be having a frequency which does not exceed half the frequency of TS_CLK. In addition the ADD_IN value should never exceed the value 0x800000. This limitation is only necessary for DMO = 0 and COA = 0 (see DPLL_CTRL_1 register).

For the normal mode replace ADD_IN of the ADDER (see **Figure 26-75**) by ADD_IN_CAL1 (when calculated, DLM=0) or ADD_IN_LD1 (when provided by the CPU, DLM=1).

The sub-pulse generation in this case is done by the following calculations using a 24 bit adder with a carry out cout and the following inputs:

- ADD_IN
- the second input is the output of the adder, stored one time stamp clock before

In order not to complicate the calculation procedure use a Multiplier with a sufficient bit width at the output and use the corresponding shifted output bits.

Enabling of the compensated output for pulses

The cout of the adder influences directly the SUB_INC1 output of the DPLL (see [Figure 26-75](#)). The compensated output SUB_INCxc is in automatic end mode only enabled by EN_Cxc when INC_CNTx >0.

Equation DPLL-26 to calculate ADD_IN in emergency mode for SMC=0

In emergency mode (RMO=1) calculate in the case LOW_RES=TS0_HRS

$$\text{ADD_IN_CALE} = (\text{NMB_S} + 0.5) * \text{RCDT_SX} \quad (\text{DPLL-26})$$

while

RCDT_SX is the 2^{32} time value of the quotient in equation DPLL-20b

In emergency mode (RMO=1) calculate in the case LOW_RES=1 and TS0_HRS=0

$$\text{ADD_IN_CALE} = (\text{NMB_S} + 0.5) * \text{RCDT_SX} / 8 \quad (\text{DPLL-26a})$$

while

RCDT_SX is the 2^{32} time value of the quotient in equation DPLL-20b

For RMO=1 and SMC=0:

$$\text{ADD_IN_CAL1} = \text{ADD_IN_CALE} \quad (\text{DPLL-26b})$$

LOW_RES=0 and TS0_HRS=1 is not possible. For such a configuration the RCS bit in the DPLL_STATUS register is set together with the ERR bit.

In the automatic end mode (DMO = 0) missing pulses should be sent to the input RPCUx (rapid pulse catch up on) in [Adder for generation of SUB_INCx by the carry Cout](#), to be caught up on with CMU_CLK0 (for COA = 0).

When normal and rapid pulses are generated simultaneously, the SUB_INCx frequency is doubled at this moment in order to count two pulses at the TBU_CHx_BASE register. In order to make the frequency doubling possible, the CMU_CLK0 should be having a frequency which does not exceed half the frequency of the system clock. In addition the ADD_IN value should never exceed the value 0x800000 when the TS_CLK frequency exceeds half the frequency of the system clock. This limitation is only necessary for DMO = 0 and COA = 0 (see DPLL_CTRL_1 register).

Generic Timer Module (GTM)

For the emergency mode replace ADD_IN of the ADDER (see [Figure 26-75](#)) by ADD_IN_CAL1 (when calculated, DLM=0) or ADD_IN_LD1 (when provided by the CPU, DLM=1).

The sub-pulse generation in this case is done by the following calculations using a 24 bit adder with a carry out cout and the following inputs:

- ADD_IN
- the second input is the output of the adder, stored one time stamp clock before.

In order not to complicate the calculation procedure use a Multiplier with a sufficient bit width at the output and use the corresponding shifted output bits.

Adder for generation of SUB_INCx by the carry C_{out}

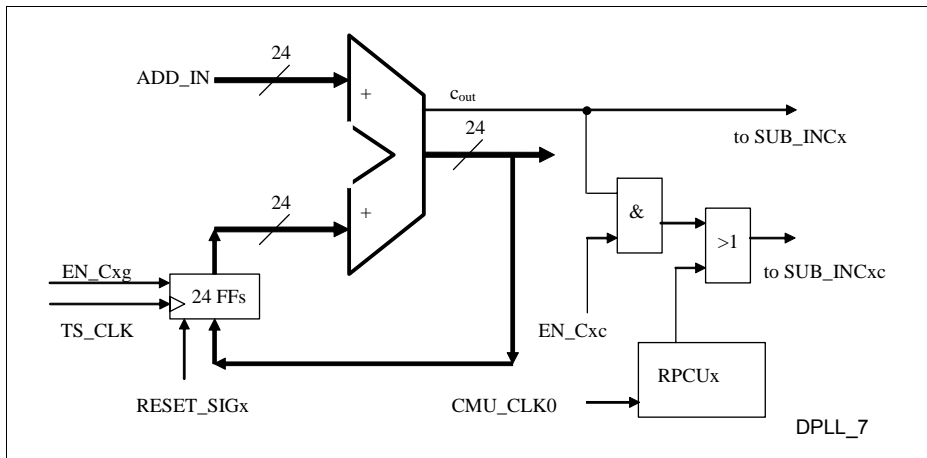


Figure 26-75 Adder for generation of SUB_INCx by the carry C_{out}

Note: The SUB_INC generation by the circuit above has the advantage, that the resolution for higher speed values is better as for a simple down counter.

After RESET and after EN_Cxg=0 the flip-flops (FFs) should have a zero value. EN_Cxg has to be zero until reliable ADD_IN values are available and the pulse generation starts. This is controlled by the configuration bits SGE1,2 in the DPLL_CONTROL_1 register. The calculated values for the increment prediction using equations DPLL-2c ([Equations DPLL-5 to calculate the current increment value](#)), DPLL-2c1 ([Equation DPLL-5 to calculate the current increment value](#)), DPLL-7c ([Equations DPLL-10 to calculate the current increment \(nominal value\)](#)) or DPLL-7c1 ([Equation DPLL-5 to calculate the current increment value](#)) respectively are valid only when at least NUTE>1 TRIGGER values or at least NUSE>1 STATE values are available. For NUTE =1 or

Generic Timer Module (GTM)

NUSE=1 respectively the equations DPLL-25 ([Equation DPLL-25 to calculate ADD_IN in normal mode for SMC=0](#)) and DPLL-26 ([Equation DPLL-26 to calculate ADD_IN in emergency mode for SMC=0](#)) use the actual increment value subtracted by the weighted average error.

The generation of SUB_INC1 pulses depends on the configuration of the DPLL.

In automatic end mode the counter INC_CNT1 resets the enable signal EN_C1 when the number of pulses desired is reached. In this case only the uncompensated output SUB_INC1 remains active in order to provide pulses for the input filter unit. A new TRIGGER input in normal mode or a new STATE input in emergency mode respectively resets the FFs and also the enable signal EN_Cxg. In the case of acceleration missing pulses can be determined at the next TRIGGER/STATE event in normal/emergency mode easily. For the correction strategy COA = 0 those missing pulses are sent out with CMU_CLK0 frequency as soon they are determined. During this time period the EN_Cxg remains cleared. After calculation or providing of a new ADD_IN value the FFs are enabled by EN_Cxg. In this way no pulse is lost. The new pulses are sent out afterwards, when INC_CNT1 is set to the desired value, maybe by adding MLT+1 or MLS1 respectively for the new TRIGGER/STATE event.

Because the used DIV procedure of the algorithms results only in integer values, a systematic failure could appear. The pulse generation at SUB_INC1 will stop in automatic end mode when the INC_CNT1 register reaches zero or all remaining pulses at a new increment will be considered in the next calculation. In this way the lost of pulses can be avoided.

When a new TRIGGER/STATE appears the value of SYN_T*(MLT+1) or SYN_S*MLS1 respectively is added to INC_CNT1, when SGE1=1. Therefore for FULL_SCALE $2*(TNU+1)*(MLT+1)$ pulses SUB_INC1 generated, when INC_CNT1 reaches the zero value. The generation of SUB_INC1 pulses has to be done as fast as possible. The calculations for the ADD_IN value must be done first. Therefore all values needed for calculation are to be fetched in a forecast.

26.16.8.4 Sub pulse generation for SMC=1

Necessity of two pulse generators

The Adder of [Figure 26-75](#) must be implemented twice in the case of SMC=1: one for SUB_INC1 controlled by the TRIGGER input and (while RMO=1) one for SUB_INC2, controlled by the STATE input. In the case described in the section above for SMC=0 only one Adder is used to generate SUB_INC1 controlled by the TRIGGER in normal mode or by STATE in emergency mode.

Equation DPLL-27 to calculate the number of pulses to be sent for the first device

using the automatic end mode condition

For SMC=1 and DMO=0

$$\text{NMB_T} = \text{MLS1} * \text{SYN_T} + \text{MP} + \text{PD_store} + \text{MPVAL1} \quad (\text{DPLL-27})$$

with

PD_store = ADT_T[12:0], prefetched during last increment

SYN_T = ADT_T[18:16], prefetched during last increment

MPVAL1 = pulse correction value for PCM1_STADOW_TRIGGER=1

while the value for PD_store is zero for AMT=0

and

for COA=0 use zero instead of the value of MP

Equation DPLL-28 to calculate the number of pulses to be sent for the second device using the automatic end mode condition

for RMO=1, SMC=1 and DMO=0

$$\text{NMB_S} = \text{MLS2} * \text{SYN_S} + \text{MP} + \text{PD_S_store} + \text{MPVAL2} \quad (\text{DPLL-28})$$

with

PD_S_store = ADT_S[15:0], prefetched during last increment

SYN_S = ADT_S[21:16], prefetched during last increment

MPVAL2 = pulse correction value for PCM2_SHADOW_TRIGGER=1

while the value for PD_store is zero for AMS=0

and

for COA=0 use zero instead of the value of MP

Please note, that these calculations above in equations DPLL-27 and DPLL-28 are only valid for an automatic end mode (DMO = 0). In addition the number of generated pulses is added by 0.5 as shown in equations DPLL-30 or DPLL-31 respectively in order to compensate rounding down errors at the succeeding division operation. Because in automatic end mode the number of pulses is limited by INC_CNTx it is guaranteed, that not more pulses as needed are generated and in the same way missing pulses are made up for the next increment.

Generic Timer Module (GTM)
Equation DPLL-30 to calculate ADD_IN for the first device for SMC=1

The sub-pulse generation in this case is done by the following calculations using a 24 bit adder with a carry out cout and the following inputs:

- ADD_IN
- the second input is the (delayed) output of the adder, stored with each time stamp clock.

Replace ADD_IN by ADD_IN_CAL1 (when calculated, DLM1=0) or ADD_IN_LD1 (when provided by the CPU, DLM1=1) respectively while:

For SMC=1 and LOW_RES=TS0_HRT

$$\text{ADD_IN_CAL1} = (\text{NMB_T} + 0.5) * \text{RCDT_TX} \quad (\text{DPLL-30})$$

When RCDT_TX is the 2^{32} time value of the quotient in equation DPLL-17b.

For SMC=1, LOW_RES= 1 and TS0_HRT=0

$$\text{ADD_IN_CAL1} = (\text{NMB_T} + 0.5) * (\text{RCDT_TX} / 8) \quad (\text{DPLL-30a})$$

When RCDT_TX is the 2^{32} time value of the quotient in equation DPLL-17b.

In order not to complicate the calculation procedure use a Multiplier with a sufficient bit width at the output and use the corresponding shifted output bits.

ADD_IN_CAL1 is a 24 bit integer value. The CDT_TX is the expected duration of current TRIGGER increment.

The cout of the adder influences directly the SUB_INC1 output of the DPLL (see [Section](#)). The SUB_INC1 output is in automatic end mode only enabled by EN_C1 when INC_CNT1 >0.

Equation DPLL-31 to calculate ADD_IN for the second device for SMC=1

Replace ADD_IN by ADD_IN_CAL2 (when calculated, DLM2=0) or ADD_IN_LD2 (when provided by the CPU, DLM2=1) respectively while:

for SMC=1, RMO=1 and LOW_RES=TS0_HRS:

$$\text{ADD_IN_CAL2} = (\text{NMB_S} + 0.5) * \text{RCDT_SX} \quad (\text{DPLL-31})$$

When RCDT_SX is the 2^{32} time value of the quotient in equation DPLL-20b.

for SMC=1, RMO=1, LOW_RES=1 and TS0_HRS=0:

Generic Timer Module (GTM)

$$\text{ADD_IN_CAL2} = (\text{NMB_S} + 0.5) * (\text{RCDT_SX} / 8) \quad (\text{DPLL.3 1a})$$

When RCDT_SX is the 2^{32} time value of the quotient in equation DPLL-20b.

In order not to complicate the calculation procedure use a Multiplier with a sufficient bit width at the output and use the corresponding shifted output bits.

The cout of the adder2 influences directly the SUB_INC2 output of the DPLL (see [Adder for generation of SUB_INCx by the carry Cout](#)).

The SUB_INC2 output is in automatic end mode only enabled by EN_C2 when INC_CNT2 > 0.

Please note, that after RESET and after EN_Cxc=0 (after stopping in automatic end mode) the flip-flops (FFs) have a zero value and also EN_Cxg has to be zero until reliable ADD_IN values are available and the pulse generation starts. The calculated values for the increment prediction using equations DPLL-2c [Equations DPLL-5 to calculate the current increment value](#), DPLL-2c1 [Equation DPLL-5 to calculate the current increment value](#), DPLL-7c [Equations DPLL-10 to calculate the current increment \(nominal value\)](#) or DPLL-7c1 [Section](#) respectively are valid only when NUTE > 1 or NUSE > 1 respectively. For NUTE=1 or NUSE=1 respectively the equations DPLL-30 (see [Equation DPLL-30 to calculate ADD_IN for the first device for SMC=1](#)) and DPLL-31 (see [Equation DPLL-31 to calculate ADD_IN for the second device for SMC=1](#)) use the actual increment value subtracted by the weighted average error.

The generation of SUB_INCx pulses depends on the configuration of the DPLL.

In automatic end mode the counter INC_CNTx resets the enable signal EN_Cxcu when the number of pulses desired is reached. In this case only the uncompensated outputs SUB_INCx remain active in order to provide pulses for the input filter units. A new TRIGGER or STATE input respectively can reset the FFs and also ADD_IN, especially when EN_Cxc was zero before. In the case of acceleration missing pulses can be determined at the next TRIGGER/STATE event easily. For the correction strategy COA = 0 those missing pulses are sent out with maximum frequency as soon they are determined. After that the pulse counter INC_CNTx should be always zero and the new pulses are sent out afterwards, when INC_CNTx is set to the desired value by adding MLS1 or MLS2 for the new TRIGGER or STATE event respectively.

Because the used DIV procedure of the algorithms results only in integer values, a systematic failure could appear. The pulse generation will stop when the INC_CNTx register reaches zero or all remaining pulses at a new increment will be considered in the next calculation. In this way the lost of pulses can be avoided.

When a new TRIGGER appears the value of SYN_T*MLS1 is added to INC_CNT1. Therefore for FULL_SCALE $2 * (\text{TNU} + 1) * \text{MLS1}$ pulses SUB_INC1 generated, when

Generic Timer Module (GTM)

INC_CNT1 reaches the zero value. The generation of SUB_INC1 pulses has to be done as fast as possible.

When a new STATE appears the value of SYN_S*MLS2 is added to INC_CNT2. Therefore for FULL_SCALE 2*(SNU+1)*MLS2 pulses SUB_INC2 generated, when INC_CNT2 reaches the zero value. The generation of SUB_INC2 pulses has to be done as fast as possible.

26.16.8.5 Calculation of the Accurate Position Values

All appearing TRIGGER and STATE signals do have a time stamp and a position stamp assigned after the input filter procedure. For the calculation of the exact time stamp the filter values are considered in the calculations of equations DPLL-1a [Equations DPLL-1a to calculate TRIGGER time stamps](#) or DPLL-6a [Equations DPLL-6a to calculate STATE time stamps](#) respectively. A corresponding calculation is to be performed for the calculation of position values.

The PSTC and PSSC values can be corrected by the CPU, when needed.

After reset, while FTD=0 and no active TRIGGER slope is detected:

$$\text{PSTC} = 0 \quad (\text{DPLL-32a})$$

Calculate the new Position value for each valid TRIGGER event:

$$\text{PSTC} = \text{PSTC_OLD} + \text{NMB_T_TAR_old} \quad (\text{DPLL-32b})$$

when FTD=1 and SGE1=1

with

PSTC_OLD is the last PSTC value and

NMB_T_old is the number of pulses which are calculated and provided for sending out in the last increment.

After reset, while FTD=0 and no active STATE slope is detected:

$$\text{PSSC} = 0 \quad (\text{DPLL-33a})$$

Calculate the new Position value for each valid STATE event:

PSSC= PSSC_OLD + NMB_S_TAR_old (DPLL-33b)

when FTD=1 and SGE1=1 (SMC=0) or SGE2=1 (SMC=1)

respectively with

PSSC_OLD is the last PSSC value and

NMB_S_old is the number of pulses which are calculated and provided for sending out in the last increment.

26.16.8.6 Scheduling of the Calculation

After enabling the DPLL with each valid TRIGGER or STATE event respectively a cycle of operations is performed to calculate all the results shown in detail in the table below DPLL-2. A state machine controls this procedure and consists of two parts, the first is triggered by a valid slope of the signal TRIGGER, begins at step 1 and ends at step 20 (in normal mode and for SMC=1). The second state machine is controlled by a valid slope of the signal STATE, begins at step 21 and ends at step 40 (in emergency mode and also for SMC=RMO=1). Depending on the mode used all 20 steps are executed or already after 2 steps the jump into the initial state is performed, as shown in the state machine descriptions below. For each new extended cycle (without this jump) all prediction values for actions in the case SMC=0 are calculated once more (with maybe improved accuracy because of better parameters) and all pending decisions are made using these new values when transmitted to the decision device.

In **State description of the State Machine**, the steps of the state machine are described. Please note, that the elaboration of the steps depends on the configuration bits described in the comments. The steps 4 to 17 are only calculated in normal mode (in the state machine explanation below marked yellow in [Section 26.16.2](#)), but steps 24 to 37 are only calculated in emergency mode (in the state machine explanation below marked cyan in [Section 26.16.2](#)) when SMC=0.

State machine partitioning for normal and emergency mode.

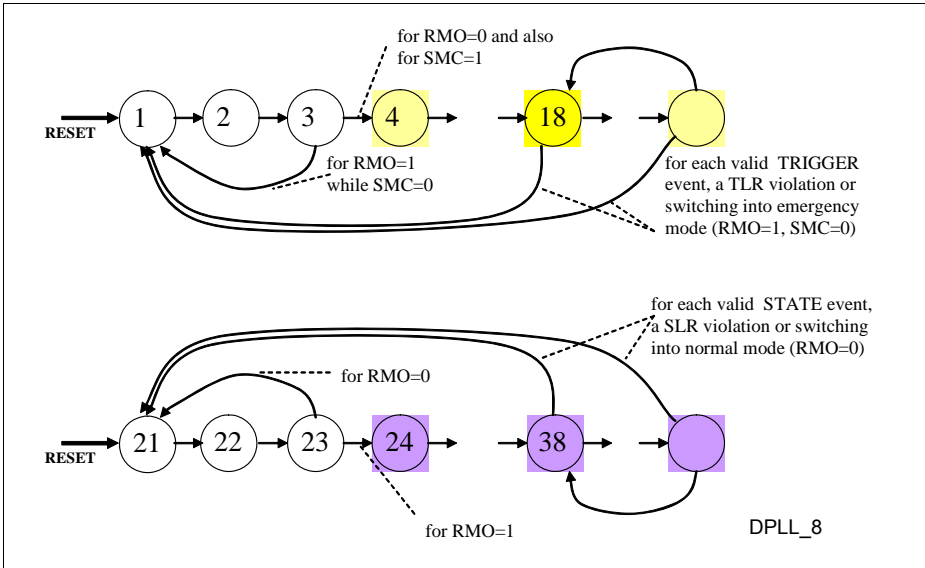


Figure 26-76 State machine partitioning for normal and emergency mode

Synchronization description

TRIGGER:

The APT (address pointer for duration and reciprocal duration values of TRIGGER increments) is initially set to zero and increment with each valid TRIGGER event. Therefore data are stored in the RAM beginning from the first available value. The actual duration of the last increment is stored at DT_T_ACT. For the prediction of the next increment it is assumed, that the same value is valid as long as NUTE is one.

A missing TRIGGER is assumed, when at least after $TOV * DT_T_ACT$ no valid TRIGGER event appears.

The data of equations DPLL-1b1 and DPLL-1c2 [Equations DPLL-1b1 and DPLL-1c1 to update the RAM after calculation](#) are written in the corresponding RAM regions and APT is increment accordingly up to $2 * TNU - 2 * SYN_NT + 1$.

The APT_2B (address pointer for the time stamp field of TRIGGER) is initially set to zero and increment with each valid TRIGGER event. When no gap is detected because of the incomplete synchronization process at the beginning, for all TRIGGER events the time stamp values are written in the RAM up to $2 * (TNU + 1)$ entries, although only $2 * (TNU + 1 - SYN_NT)$ events in FULL_SCALE appear. When the current position is detected, the synchronization procedure can be performed as described below:

Generic Timer Module (GTM)

Before the CPU sets the APT_2C address pointer in order to synchronize to the profile, it writes the corresponding increment value for the necessary extension of the RAM region 2B value APT_2B_EXT into the register APT_2B_sync and sets the status bit APT_2C_status. This value can be e.g. $2 \cdot \text{SYN_NT}$, when all gaps in FULL_SCALE already passed the input data stream of TRIGGER, or less than this value, when up to now e.g. only a single gap is to be considered in the data stream stored already in the RAM region 2B. The number of virtual increments to be considered depends on the number of inputs already got. After writing APT_2C by the CPU, with the next TRIGGER event the APT_2B address pointer is incremented (as usual) and then the additional offset value APT_2B_EXT is added to it once (while APT_2B_STATUS=1 and for forward direction). For that reason the APT_2B_STATUS bit is reset after it. The old APT_2B value before adding the offset is stored in the APT_2B_OLD register as information for the CPU where to start the extension procedure. In the following the CPU fills in the time stamp field around the APT_2B_OLD position taking into account the corresponding number of virtual entries stored in the APT_2B_EXT value and the corresponding NT values in the profile. The extension procedure ends when all gaps considered in the APT_2B_EXT value are treated once. In the consequence all storage locations of RAM region 2B up to now do have the corresponding entries. Future gaps are treated by the DPLL.

For a backward direction the APT_2C_ext value is subtracted accordingly.

When the CPU writes the APT_2C address pointer the SYT bit is set simultaneously. For SYT=1 in normal mode (SMC=0) the LOCK1 bit is set with the system clock, when the right number of increments between two synchronization gaps is detected by the DPLL. An unexpected missing TRIGGER or an additional TRIGGER between two synchronization gaps does reset the LOCK1 bit in normal mode. In that case the CPU must correct the SUB_INC pulse number and maybe correct the APT_2C pointer. For this purpose the LL1I interrupt can be used.

When SYT is set the calculations of equations DPLL-1 to DPLL-5 are performed accordingly and the values are stored in (and distributed to) the right RAM positions.

This includes the multiple time stamp storage by the DPLL for a gap according to equations DPLL-1a5 to 7 forwards [Equation DPLL-1a5-7 to extend the time stamp values for TRIGGER in forward direction](#) or backwards [Equations DPLL-1a5-7 for backward direction](#). The APT_2B pointer is for that reason incremented or decremented before this operation considering the virtual increments in addition.

Please note, that for the APT and APT_2C pointers the gap is considered as a single increment.

STATE:

The APS (address pointer for duration and reciprocal duration values of STATE) is initially set to zero and incremented with each valid STATE event. Therefore data are stored in the RAM field beginning at the first location. The ACT duration of the last

Generic Timer Module (GTM)

increment is stored at DT_S_ACT. For the prediction of the next increment it is assumed, that the same value is valid as long as NUSE is one.

A missing STATE is assumed, when at least after $TOV_S * DT_S_actual$ no valid STATE event appears.

The data of equations DPLL-6b1 and DPLL-6c2 **Equations DPLL-6b1 and DPLL-6c1 to update the RAM after calculation** is written in the corresponding RAM regions and APS is increment accordingly up to $2 * SNU - 2 * SYN_NS + 1$ (for $SYSF=0$).

The APS_1C2 (address pointer for the time stamp field of STATE) is initially set to zero and increment with each valid STATE event. When no gap is detected because of the incomplete synchronization process at the beginning, for all STATE events the time stamp values are written in the RAM up to $2 * (SNU + 1)$ entries, although (e.g. for $SYSF=0$) only $2 * (SNU + 1 - SYN_NS)$ events in FULL_SCALE appear. When the current position is detected, the synchronization procedure can be performed as described below:

Before the CPU sets the APS_1C3 address pointer in order to synchronize to the profile, it writes the corresponding increment value APS_1C2_EXT for the necessary extension of the RAM region 1C2 into the register DPLL_APS_SYNC and sets the APS_1C2_STATUS bit there. This value can be e.g. $2 * SYN_NS$ (for $SYSF=0$) or SYN_NS (for $SYSF=1$), when all gaps in FULL_SCALE already passed the input data stream of STATE. Also less than this value can be considered, when up to now only a single gap is to be considered in the data stream stored already in the RAM region 1C2. The number of increments to be considered depends on the number of inputs already got. After writing APS_1C3 by the CPU, with the next valid STATE slope the APS_1C2 address pointer is increment (as usual) and then the additional offset value APS_1C2_EXT is added to it once (while APS_1C2_STATUS=1 and forward direction). For that reason the APS_1C2_STATUS bit is reset after it. The old APS_1C2 value is stored in the APS_1C2_OLD register as information for the CPU where to start the extension procedure. In the following the CPU extends the time stamp field beginning from the APS_1C2_OLD position taking into account the corresponding number of virtual entries according to the APS_1C2_EXT value and also the correspondent NS values in the profile. The extension procedure ends when all gaps considered in the APS_1C2_EXT value are treated once. In the consequence all storage locations of RAM region 1C2 up to now do have the corresponding entries. Future gaps are treated by the DPLL.

For a backward direction the APS_1C2_EXT value is subtracted accordingly.

When the CPU writes the APS_1C3 address pointer the SYS bit is set simultaneously. For $SYS=1$ in emergency mode ($SMC=0$ and $DMO=1$) the LOCK1 bit is set with the system clock, when the right number of increments between two synchronization gaps is detected by the DPLL. An unexpected missing STATE or an additional STATE between two synchronization gaps does reset the LOCK1 bit in emergency mode. In that

Generic Timer Module (GTM)

case the CPU must correct the SUB_INC1 pulse number and maybe correct the APS_1C3 pointer. For this purpose the LL11 interrupt can be used.

When SYS is set the calculations of equations DPLL-5 to DPLL-10 are performed accordingly and the values are stored in (and distributed to) the right RAM positions.

This includes the multiple time stamp storage by the DPLL for a gap according to equations DPLL-6a5 to 7 forwards [Equations DPLL-6a5-7 to extend the time stamp values for STATE](#) or backwards [Equations DPLL-6a5-7 for backward direction](#). The APS_1C2 pointer is for that reason increment or decremented before this operation considering the virtual increments in addition.

Please note, that for the APS and APS_2C pointers the gap is considered as a single increment.

SMC=1:

For SMC=1 it is assumed, that the starting position is known by measuring the characteristic of the device. In this way the APT and APT_2C as well the APS and APS_1C3 values are set properly, maybe with an unknown repetition rate. When no gap is to be considered for TRIGGER or STATE signals the APT_2B and APS_1C2 address pointers are set equal to APT or APS respectively. It is assumed, that all missing TRIGGERS and missing STATES can be also considered from the beginning, when a valid profile with the corresponding adapt values is written in the RAM regions 1C3 and 2C respectively. In that case the TSF_T[i] and TSF_S[i] must be extended by the DPLL according to the profile. Thus the SYT and SYS bits could be set from the beginning and the LOCK1 and LOCK2 bits are set after recognition of the corresponding gaps accordingly. When no gap exists (SYN_NT=0 or SYN_NS=0), the LOCK bits are set immediately. The CPU can correct the APT_2C and APS_1C3 pointer according to the recognized repetition rate later once more without the loss of Lock1,2.

Operation in backward direction in normal and emergency mode (SMC=0)

When for SMC=0 in normal mode a backwards condition is detected for the TRIGGER input signal (e.g. when THMI is not violated), the LOCK1 bit and the FTD (as well as LOCK2 and FSD) in the DPLL_STATUS register is reset, the NUTE value in NUTC register is set to 1 (the same for NUSE in NUSC). The address pointers APT and APT_2C as described below (and after that decremented for each following valid slope of TRIGGER as long as the DIR1 bit shows the backward direction).

Please notice, that in the case of the change of the direction the ITN and ISN bit in the DPLL_STATUS register are reset.

For this transition to the backward direction no change of address pointer APT and APT_2B is necessary.

profile update for TRIGGER when changing direction

The profile address pointer APT_2C is changed step by step in order to update the profile information in SYN_T, SYN_T_old and PD_store

Generic Timer Module (GTM)

- decrement APT_2C, load SYN_T
- decrement APT_2C, load SYN_T
- decrement APT_2C, load SYN_T, PD_store, update SYN_T_OLD
- decrement APT_2C, make calculations, load SYN_T and PD_store, update SYN_T_OLD and PD_store_old and wait for a new TRIGGER event.

Note: The update of SYN_T_OLD and the loading of PD_store can be performed in all steps above. The value APT_2B needs not to be corrected. for a direction change from backwards to forwards make the same corrections by incrementing APT_2C.

Make calculations does mean: the operation of the state machine starts in normal mode at step 1 with the calculations and results in an update of the SYN_T value including the automatic update of SYN_T_OLD at state 17 using the actual APT_2C address pointer value, see [Section 26.16.2](#).

The TBU_TB1 value is to be corrected by the number of pulses sent out in the wrong direction mode during the last and current increment. This correction is done by sending out SUB_INC1 pulses for decrementing TBU_TB1 (while DIR1=1).

The amount of pulses is determined by calculation of the difference between NMB_T and INC_CNT1. This difference is multiplied by 2 and added to INC_CNT1. In addition the pulses for the new increment SYN_T_OLD*(MLT+1) are sent. All pulses are sent out by the maximum possible frequency, because no speed information is available for the first increment after changing the direction.

Save inc_cnt1 value at direction change to inc_cnt1_save.

Calculate the new inc_cnt1 value as follows:

1. Stop sending pulses and save inc_cnt1 at the moment of direction change as inc_cnt1_save.
2. Set inc_cnt1 to the target value of the last increment
nmb_t_tar_old
3. Add the number of trigger which were calculated for the current increment when this value was already added to inc_cnt1 before the direction change is detected
+ nmb_t
4. Subtract the value of still not sent pulses (remaining value at inc_cnt1_save)
- inc_cnt1_save
5. Calculate the new target pulses to be sent considering the new values of SYN_T and PD_store and add them:
+ nmb_t_tar_new

This does mean the following equation:

$$\text{inc_cnt1} = \text{nmb_t_tar_old} + \text{nmb_t} - \text{inc_cnt1_save} + \text{nmb_t_tar_new}$$

Generic Timer Module (GTM)

All pulses summarized at `inc_cnt1` are sent out by the maximum possible frequency, because no speed information is available for the first increment after changing the direction. Please notice that no pulse correction using `PCM1` of `DPLL_CTRL1` is possible during direction change.

When `PSTC` was incremented/decremented at the active slope and after that the direction change was detected at the same input event, correct `PSTC` once by

- `nmb_t_tar_old` when changed to backwards

+ `nmb_t_tar_old` when changed to forwards

in order to compensate the former operation. When the direction information is known before an intended change of `PSTC`, do not change them.

Store the new calculated value `nmb_t_tar_new` at `nmb_t_tar` for the correct calculation of `PSTC` at the next input event.

consequences for `STATE`

With the next valid `STATE` event the direction information is already given. The profile pointer `APS_1C3` is to be corrected by a two times decrement in order to point to the profile of the next following increment. In the following it is decremented with each `STATE` event while `DIR1=1`. The `SYN_S` and `PD_S_store` values must be updated accordingly, including `SYN_S_OLD` and `PD_S_store_old`.

Because the right direction is already known when an input event appears, make the following corrections:

- decrement `APS_1C3`, load `SYN_S` and `PD_S_store`, update `SYN_S_OLD` and `PD_S_store_old`
- decrement `APS_1C3`, make calculations, load `SYN_S` and `PD_S_store`, update `SYN_S_OLD` and `PD_S_store_old`, wait for a new `STATE` event

Note: The update of `SYN_S_OLD` and the loading of `PD_S_store` can be performed in all steps above. The value of `APS_1C2` needs not to be corrected.

When a new `STATE` event occurs, all address pointers are decremented accordingly as long as `DIR1=1`.

In emergency mode the pulses are corrected as follows:

Save `inc_cnt1` value at direction change to `inc_cnt1_save`.

Calculate the new `inc_cnt1` value as follows:

1. Stop sending pulses and save `inc_cnt1` at the moment of direction change as `inc_cnt1_save`.

2. Set `inc_cnt1` to the target value of the current increment

`nmb_s_tar`

Please notice, that in difference to the normal mode, `nmb_s_tar` is to be used instead of `nmb_s_tar_old`, because direction information in emergency mode is only given from the `TRIGGER` input and occurs independent of a `STATE` event.

Generic Timer Module (GTM)

That means: The calculations at the last STATE event were done for the correct former direction.

3. Do not add the calculated number of state pulses because no new STATE event occurred.

4. Subtract the value of still not sent target pulses (remaining value at `inc_cnt1_save`)
- `inc_cnt1_save`

5. Add the new calculated target pulses for the current increment

+ `nmb_s_tar_new`

when for the calculation all new conditions of `PD_S_store` and `SYN_S` are considered.

`inc_cnt1 = nmb_s_tar_old - inc_cnt1_save + nmb_s_tar_new`

All pulses summarized at `inc_cnt1` are sent out by the maximum possible frequency, because no speed information is available for the first increment after changing the direction. Please notice that no pulse correction using `PCM1` of `DPLL_CTRL1` is possible during direction change.

Do not change `PSSC` and suppress incrementing/decrementing of `PSSC` at the event directly following to the direction change information.

Store the new calculated value `nmb_s_tar_new` at `nmb_s_tar` for the correct calculation of `PSTC` at the next input event.

repeated change to forward direction for `TRIGGER`

The `DIR1` bit remains set as long as the `THMI` value remains none violated for the following `TRIGGER` events and is reset when for an invalid `TRIGGER` slope the `THMI` is violated.

Resetting the `DIR1` to 0 results (after repeated reset of `LOCK1`, `NIT`, `NIS`) in the opposite correction of the profile address pointer considered.

This does mean four increment operations of the address pointer `APS_1C3` including the update of `SYN_S` and `PD_S_store` with the automatic update of `SYN_S_OLD` and `PD_S_store_old` four increment operations of the address pointer `APT_2C` including the update of `SYN_T` and `PD_store` with the automatic update of `SYN_S_OLD` and `PD_S_store_old`.

The correction of `TBU_TS1` is done by sending out the correction pulses with the highest possible frequency at `SUB_INC1` while `DIR1=0`. The number of pulses is calculated as shown above.

consequences for `STATE`

For `DIR1=0`: `DIR2` is reset to 0 after the next valid `STATE` event and after a last decrementing of `APS`, `APS_1C3` as well as `APS_1C2`. After that the address pointers are increment again with each following valid `STATE` event as long as `DIR1=0`.

Generic Timer Module (GTM)
Operation in backward direction for TRIGGER (SMC=1)

When for SMC=1 a backwards condition is detected for the TRIGGER input signal (TDIR=1, resulting in DIR1=1), the LOCK1 bit and the FTD in the DPLL_STATUS register is reset, the NUTE value in NUTC register is set to 1. The address pointers APT and APT_2C as well as APT_2B are incremented for a last time (and after that decremented for each following valid slope of TRIGGER as long as the DIR1 bit shows the backward direction).

Please notice, that in the case of the change of the direction the ITN bit in the DPLL_STATUS register is reset.

profile update for TRIGGER

Make the same update steps for the profile address pointer as shown in **Operation in backward direction in normal and emergency mode (SMC=0)**: Decrement APT_2C for 2 times with the update of the SYN_T and PD_store values at each step with an automatic update of SYN_T_OLD and PD_store_old.

- decrement APT_2C, load SYN_T, PD_store, update SYN_T_OLD
- decrement APT_2C, make calculations, load SYN_T and PD_store, update SYN_T_OLD and PD_store_old and wait for a new TRIGGER event.

In the normal case no correction of wrong pulses sent is necessary, because the direction change is detected by the pattern immediately.

Nevertheless a correction is necessary as shown below. In the other case: see treatment of pulses TBU_TS1 in normal mode at **Operation in backward direction in normal and emergency mode (SMC=0)**.

Save inc_cntx value at direction change to inc_cnt1_save.

Calculate the new inc_cnt1 value as follows:

1. Clear inc_cnt1.
2. Set inc_cnt1 to the target value of the last increment

nmb_t_tar

Please notice, that in difference to the normal mode, nmb_t_tar is to be used instead of nmb_t_tar_old, because the direction information is known before the calculation takes place.

3. Do not add the calculated number of trigger pulses because it is not calculated yet before the direction change information is known.

4. Subtract the value of still not sent pulses (remaining value at inc_cnt1_save)

- inc_cnt1_save

5. Add the new calculated target pulses for the current increment

+ nmb_t_tar_new

when for the calculation all new conditions of PD_S_store and SYN_S are considered.

Generic Timer Module (GTM)

$inc_cnt1 = nmb_t_tar_old - inc_cnt1_save + nmb_t_tar_new$

All pulses summarized at `inc_cnt1` are sent out by the maximum possible frequency, because no speed information is available for the first increment after changing the direction. Please notice that no pulse correction using PCM1 of DPLL_CTRL1 is possible during direction change.

Suppress changing of PSTC for the TRIGGER event when a direction change is detected.

Store the new calculated value `nmb_t_tar_new` at `nmb_t_tar` for the correct calculation of PSTC at the next input event.

repeated change to forward direction for TRIGGER

The DIR1 bit remains set as long as the TDIR bit is set for the following TRIGGER events and is reset when for a valid TRIGGER slope the TDIR is zero.

Resetting the DIR1 to 0 results (after repeated reset of LOCK1 and ITN) in the opposite correction of the address pointer use.

This does mean two increment operations of the address pointer including the update of SYN_T and PD_store.

A complex correction of TBU_TS1 and INC_CNT1 is in the normal case not necessary, when all increments are equal (SYN_NT=0) and no adapt information is used. In this case only the MLS1 value is added to INC_CNT1 in order to backcount the value for the last increment. In the other case: see treatment of pulses TBU_TS1 and ICN_CNT1 in normal mode at [Operation in backward direction in normal and emergency mode \(SMC=0\)](#).

Operation in backward direction for STATE (SMC=1)

When for SMC=1 a backwards condition is detected for the STATE input signal (SDIR=1, resulting in DIR2=1), the LOCK2 bit and the FSD in the DPLL_STATUS register is reset, the NUSE value in NUSC register is set to 1 and the address pointers APS and APS_1C3_f and APS_1C2 are increment for a last time (and after that decremented for each following valid slope of STATE as long as the DIR2 bit shows the backward direction.

Please notice, that in the case of the change of the direction the ISN bit in the DPLL_STATUS register is reset.

For this transition to the backward direction no change of address pointer APS and APC_1C2 is necessary.

profile update for STATE

Make the same update steps for the profile address pointer as shown in [Operation in backward direction in normal and emergency mode \(SMC=0\)](#): Decrement APS_1C3 for 2 times with the update of the SYN_S, SYN_S_OLD and PD_S_store and PD_S_store_old values at each step:

Generic Timer Module (GTM)

A correction of TBU_TS2 and INC_CNT2 is in the normal case not necessary, because no pulses are sent out with the wrong direction information. In the other case: see treatment of pulses TBU_TS1 and ICN_CNT1 in normal mode at **Operation in backward direction in normal and emergency mode (SMC=0)**.

- decrement APT_1C3, load SYN_S, PD_S_store, update SYN_S_OLD
- decrement APT_1C3, make calculations, load SYN_S and PD_S_store, update SYN_S_OLD and PD_S_store_old and wait for a new STATE event.

A complex correction of TBU_TS2 and INC_CNT2 is in the normal case not necessary, when all increments are equal (SYN_NS=0) and no adapt information is used. In this case only the MLS2 value is added to INC_CNT2 in order to backcount the value for the last increment. In the other case: see treatment of pulses TBU_TS1 and ICN_CNT1 in normal mode at **Operation in backward direction in normal and emergency mode (SMC=0)**.

For the second PMSM the pulses are corrected as follows:

Save inc_cnt2 value at direction change to inc_cnt2_save.

Calculate the new inc_cnt2 value as follows:

1. Clear inc_cnt2.
2. Set inc_cnt2 to the target value of the last increment
nmb_s_tar

Please notice, that in difference to the normal mode, nmb_s_tar is to be used instead of nmb_s_tar_old, because no new calculation is performed so far.

3. Do not add the calculated number of state pulses because it is not calculated yet before the direction change information is known.
4. Subtract the value of still not sent pulses (remaining value at inc_cnt2_save)
- inc_cnt2_save
5. Add the new calculated target pulses for the current increment
+ nmb_s_tar_new

when for the calculation all new conditions of PD_S_store and SYN_S are considered.

$inc_cnt2 = nmb_s_tar_old - inc_cnt2_save + nmb_s_tar_new$

All pulses summarized at inc_cnt2 are sent out by the maximum possible frequency, because no speed information is available for the first increment after changing the direction. Please notice that no pulse correction using PCM2 of DPLL_CTRL1 is possible during direction change.

Do not change PSSC for a STATE event when a direction change is detected.

Store the new calculated value nmb_s_tar_new at nmb_s_tar for the correct calculation of PSTC at the next input event.

repeated change to forward direction for STATE

Generic Timer Module (GTM)

The DIR2 bit remains set as long as the SDIR bit is set for the following STATE events and is reset when for a valid STATE slope SDIR is zero.

Resetting the DIR2 to 0 results (after repeated reset of LOCK2 and FSD) in the opposite correction of the address pointer use.

After a last decrementing of all address pointers the APS_1C3 is increment 2 times with a repeated update of SYN_S, SYN_S_OLD and PD_S_store after each increment.

DPLL reaction in the case of non plausible input signals

When the DPLL is synchronized concerning the TRIGGER signal by setting the FTD, SYT and LOCK1 bits in the DPLL_STATUS register, the number of valid TRIGGER events between the gaps is to be checked continuously.

When additional events appear while a gap is expected, the LOCK1 bit is reset and the ITN bit in the DPLL_STATUS register is set.

When an unexpected gap appears (missing TRIGGERS), the NUTE value in the NUTC register is set to 1, the LOCK1 bit is reset and the ITN bit in the DPLL_STATUS register is set. The address pointers are increment with the next valid TRIGGER slope accordingly.

When the TRIGGER locking range TLR is violated 8),

the state machine 1 will remain in state 1 and the address pointer APT, APT_2B and APT_2C will remain unchanged until the CPU sets the APT_2C accordingly. In this case also the NUTE value in the NUTC register is set to 1. The DPLL stops the generation of the SUB_INC1 pulses and will perform no other actions - remaining in step1 of the first state machine (see [Section 26.16.2](#)).

8) The TOR Bit in the DPLL_STATUS register is set, when the time to the next active TRIGGER slope exceeds the value of the last nominal TRIGGER duration multiplied with the value of the TLR register (see [Section 26.16.11.75](#)). In this case also the TORI interrupt is generated, when enabled.

When in the following the direction DIR1 changes as described in the sections above the ITN bit in the DPLL_STATUS register is reset, the use of the address pointers APT_2C is switched and the pulse correction takes place as described above.

In all other cases the CPU can interact to leave the instable state. This can be done by setting the APT_2C address pointer which results in a reset of the ITN bit. In the following NUTE can also be set to higher values.

When the DPLL is synchronized concerning the STATE signal by setting the FSD, SYS and LOCK1 (for SMC=0) or LOCK2 (for SMC=1) bits in the DPLL_STATUS register, the number of valid STATE events between the gaps is to be checked continuously.

When additional events appear while a gap is expected or while an unexpected missing STATE event appears, the LOCK1,2 bit is reset and the ISN bit in the DPLL_STATUS register is set.

Generic Timer Module (GTM)

When an unexpected gap appears for $RMO=SMC=1$ (missing STATEs for synchronous motor control), the NUSE value in the NUSC register is set to 1, the LOCK2 bit is reset and the ISN bit in the DPLL_STATUS register is set. The address pointers are incremented with the next valid STATE slope accordingly.

When the STATE locking range SLR is violated 7),

the state machine 2 will remain in state 21 and the address pointer APS, APS_1C2 and APS_1C3 will remain unchanged until the CPU sets the APS_1C3 accordingly. In this case also the NUSE value in the NUSC register is set to 1. The DPLL stops the generation of the SUB_INC1,2 pulses respectively and will perform no other actions - remaining in step21 of the second state machine (see [Section 26.16.2](#)).

7) The SOR Bit in the DPLL_STATUS register is set, when the time to the next active STATE slope exceeds the value of the last nominal STATE duration multiplied with the value of the SLR register (see [Section 26.16.11.76](#)).

In this case also the SORI interrupt is generated, when enabled.

When in the following the direction DIR2 changes as described in the sections above the ISN bit in the DPLL_STATUS register is reset, the use of the address pointers APS_1C3 is switched and the pulse correction takes place as described above. In all other cases the CPU must interact to leave the instable state. This can be done by setting the APS_1C3 address pointers which results in a reset of the ISN bit. In the following NUSE can also be set to higher values.

State description of the State Machine.

Table 26-48 State description of the State Machine

Step	Description	Comments
always for DEN=1	for each inactive TRIGGER slope: generate the TISI interrupt; calculate the time stamp difference ΔT to the last valid event, store this value at THVAL; when THMI >0 is violated ($\Delta T < THMI$): generate TINI interrupt, set DIR1=0 (forwards) set BWD1=0 (see DPLL_STATUS register) else (only for THMI >0): set DIR1= 1 (backwards); set BWD1=1 (see DPLL_STATUS register) after changing the direction correct the pulses WP sent with wrong direction information and send the pulses for the actual increment in addition with highest possible frequency: WP=NMB_T-DPLL_INC_CNT1; correct INC_CNT1 by addition of 2*WP before sending the correction pulses; check THMA, when THMA is violated, generate the TAXI interrupt; go to step 1 for each inactive STATE slope: set DIR2=DIR1	for SMC=0; set DIR1 always after inc./decr. the address pointers APT, APT_x; go to step 1; stop output of SUB_INC1 and correct pulses after changing DIR1 after incr./decr. of APS_x set DIR2 always after incr./decr. the address pointers APS, APS_x; go to step 1

Generic Timer Module (GTM)

Table 26-48 State description of the State Machine (cont'd)

Step	Description	Comments
always for DEN=1	<p>set DIR1=BWD1=TDIR, set DIR2=BWD2=SDIR; for each change of TDIR go to step 1 after performing the following calculations: correct INC_CNT1 correct the pulses (WP, see above) sent with wrong direction information and send the pulses for the actual increment in addition with highest possible frequency.</p> <p>For each change of SDIR go to step 21 after performing the following calculations: update of SYN_S, PD_S_store according to Operation in backward direction in normal and emergency mode (SMC=0) correct INC_CNT1,2 correct the pulses sent with wrong direction information and send the pulses for the actual increment in addition with highest possible frequency.</p>	<p>for SMC=1; set the direction bits always after incr./decr. the corresponding address pointers;</p>
1	<p>When DEN = 0 or TEN=0: stay in step 1 until DEN=1, TEN=1 and at least one valid TRIGGER has been detected (FTD=1);</p> <p>the following steps are performed always (not necessarily in step 1, but also in steps 18 to 20 (when waiting for new PMTR values to be calculated): compare TRIGGER_S with TSL (valid slope); When no valid TRIGGER appears and when TS_T_CHECK time is reached:</p> <ul style="list-style-type: none"> send missing TRIGGER INT, also when a Gap is expected according to the profile; set MT=1 (missing TRIGGER bit) in the DPLL_STATUS register; do not leave the active step, until a valid TRIGGER appears. 	<p>Depending on TSL, TEN, DEN the leaving of step one is done with the next TRIGGER input; Note: Step 1 is also left in emergency mode when a valid TRIGGER event appears in order to make a switch back to normal mode possible; _old - values are values valid at the last but one valid TRIGGER event;</p> <p>for the whole table: use always MLS1 instead of (MLT+1) for the case SMC=1;</p>

Generic Timer Module (GTM)

Table 26-48 State description of the State Machine (cont'd)

Step	Description	Comments
1	<p>When a valid TRIGGER appears check PVT - when the PVT value is violated: generate the PWI interrupt, ignore the TRIGGER input and wait for the next valid TRIGGER slope (ignore each invalid slope); do not store any value</p> <p>- When the PVT value is fulfilled: store actual position stamp at PSTM (value at the TRIGGER event) update the RAM region 2 by equation DPLL-a-c (see Chapter 26.16.7.5) store the actual INC_CNT1 value at MP1 as missing pulses (instead of calculating in step 5) store all relevant configuration bits X of the DPLL_CTRL(0,1) Registers in shadow registers and consider them for all corresponding calculations of steps 2 to 20 accordingly; the relevant bits are explained in the registers itself generate the TASI interrupt;</p> <p>for FTD=0:</p> <ul style="list-style-type: none"> • set PSTC=PSTM • set FTD (first TRIGGER detected) • do not change PSTC,APT, APT_2B • for (RMO=0 or SMC=1) and SGE1=1: increment INC_CNT1 by (MLT+1)* + MPVAL1***) • send SUB_INC1 pulses with highest possible frequency when SGE1=1 <p>for SYT=0 and FTD =1:</p> <ul style="list-style-type: none"> • dir_crement APT and APT_2B by one; • dir_crement for SGE1_delay****)=1 and TOR=0 PSTC by NMB_T_TAR**) • for (RMO=0 or SMC=1) and SGE1=1. TOR=0: increment INC_CNT1 by (MLT+1)* + MPVAL1***) 	<p>dir_crement does mean: increment for DIR1=0 decrement for DIR1=1</p> <p>*) replace (MLT)+1 by MLS1 for SMC=1</p> <p>***) NMB_T_TAR is the target value of NMB_T of the last increment (see step 5 ff.)</p> <p>****) SGE1_delay is the value of SGE1 delay by one valid TRIGGER event</p>

Generic Timer Module (GTM)
Table 26-48 State description of the State Machine (cont'd)

Step	Description	Comments
1	for SYT=1 and TOR=0: <ul style="list-style-type: none"> dir_crement APT, APT_2C, dir_crement APT_2B by SYN_T_OLD dir_crement for SGE1_delay****)=1 PSTC by NMB_T_TAR**) for (RMO=0 or SMC=1) and SGE1=1: increment INC_CNT1 by SYN_T*(MLT+1)* + PD_store***** + MPVAL1***) PD_store is 0 for AMT=0 within the DPLL_STATUS register: <ul style="list-style-type: none"> set LOCK1 bit accordingly; 	****)PD_store=0 for AMT=0 (see DPLL_CTRL_0 register)
2	calculate TS_T according to equations DPLL-1a; calculate DT_T_ACT = TS_T - TS_T_old calculate RDT_T_ACT calculate QDT_TX according to equation DPLL-2	
3	send CDTI interrupt when NTI_CNT is zero or decrement NTI_CNT when not zero; calculate EDT_T and MEDT_T according to equations DPLL-3 and DPLL-4 for (RMO=1 and SMC=0): update SYN_T, PD_store and go back to step 1	Note: There are different behaviours of Reference Model and hardware: For the hardware the values of SYN_T and PD_store are not updated until a new valid TRIGGER slope occurs.
4	calculate CDT_TX according to equation DPLL-5a and b;	for RMO=0 or SMC=1;
5	calculate missing pulses: MP1 = INC_CNT1(at the moment of a valid TRIGGER slope) calculate target pulses: NMB_T_TAR = (MLT+1)*SYN_T + PD_store + MPVAL1 (instead of PD_store use zero in the case AMT=0)	for RMO=0 or SMC=1 *)replace (MLT+1) by MLS1 for SMC=1 add MPVAL1 only for PCM=1; add MPVAL1 once to INC_CNT1 and reset PCM1 after that;
6	sent MP with highest possible frequency and set NMB_T = NMB_T_TAR	for RMO=0 or SMC=1, DMO=0 and COA=0

Generic Timer Module (GTM)
Table 26-48 State description of the State Machine (cont'd)

Step	Description	Comments
7	calculate the number of pulses to be sent $NMB_T = NMB_T_TAR + MP$ (see equations DPLL-21 or DPLL-27 respectively)	for RMO=0 or SMC=1, DMO=0 and COA=1
8	$NMB_T = SYN_T * CNT_NUM1$	for RMO=0 or SMC=1, DMO=1
9	update SYN_T and PD_store;	Note: There are different behaviours of Reference Model and hardware: For the hardware the values of SYN_T and PD_store are not updated until a new valid TRIGGER slope occurs.
10	calculate ADD_IN_CAL1 according to equation DPLL-25 and DPLL-25b or DPLL-31 and store this value in RAM use ADD_IN_CAL1 as ADD_IN value for the case DLM=0 use ADD_IN_LD1 as ADD_IN for the case DLM=1, but do this update immediately (without waiting for this step 10); for DMO=DLM=0 and EN_C1u=0: reset the FlipFlops in the SUB_INC1 generator; start sending SUB_INC1;	for RMO=0 or SMC=1 for DLM=0 for DLM=1
11	calculate $TS_T_CHECK = TS_T + DT_T_ACT * (TOV)$;	for RMO=0 or SMC=1;
12	automatic setting of actions masking bits in the DPLL_STATUS register: for SMC=0: set CAIP1=CAIP2=1 for SMC=1: set only CAIP1=1	steps 12 to 16 are not valid for the combination: (SMC=0 and RMO=1)

Generic Timer Module (GTM)
Table 26-48 State description of the State Machine (cont'd)

Step	Description	Comments
13	for all correspondent actions with ACT_N[i]=1 calculate: $NA[i] = (PSA[i] - PSTC)/(MLT+1)^*$ for forward direction with w= integer part and b = remainder of the division (fractional part); for backward direction use $NA[i] = (PSTC - PSA[i])/(MLT+1)^*$ and consider in both cases the time base overflow in order to get a positive difference	actions 0...11 for SMC=1 actions 0...23 for SMC=0 depending on ACT_Ni in DPLL_ACT_STA register; replace MLT+1 by MLS1 for SMC=1
14	calculate PDT_T[i] and DTA[i] for up to 24 action values according to equations DPLL-11 and DPLL-12;	actions 0...11 for SMC=1 actions 0...23 for SMC=0
15	calculate TSAC[i] according to equation DPLL-15 and PSAC[i] according to equation DPLL-17	actions 0...11 for SMC=1 actions 0...23 for SMC=0
16	automatic resetting of actions masking bits in the DPLL_STATUS register: for SMC=0: set CAIP1=CAIP2=0 for SMC=1: set only CAIP1=0; set the corresponding ACT_Ni bits in the DPLL_ACT_STA register	Set ACT_Ni for all enabled actions concerned: 0...11 for SMC=1 0...23 for SMC=0
17	check the relation of the last increment to its predecessor according to the profile and taking into account TOV: set the ITN status bit and reset the corresponding LOCK bit, when not plausible; go to step 18, when no valid TRIGGER appears for all following steps 18 to 20: go immediately back to step 1, when a valid TRIGGER event occurs, interrupt all calculations there and reset all CAIP in that case; when going back to step 1: store TS_T in RAM 2B according to APT_2B; update RAM 2A and RAM 2D	for all conditions

Generic Timer Module (GTM)

Table 26-48 State description of the State Machine (cont'd)

Step	Description	Comments
18	wait for a new PMTR value; set the corresponding CAIPx values and go to step 19 in that case	go immediately to step 1 and update the RAM according to step 17 when a valid TRIGGER event occurs
19	make the requested action calculation according to new PMTR values	go immediately to step 1 and update the RAM according to step 17 when a valid TRIGGER event occurs
20	reset CAIPx and go back to step 18	go immediately to step 1 and update the RAM according to step 17 when a valid TRIGGER event occurs

Generic Timer Module (GTM)
Table 26-48 State description of the State Machine (cont'd)

Step	Description	Comments
21	<p>When DEN = 0 or SEN=0: stay in step 1 until DEN=1, SEN=1 and at least one valid STATE has been detected (FSD=1);</p> <p>the following steps are performed always (not necessarily in step 21, but also in steps 38 to 40 (when waiting for new PMTR values to be calculated): compare STATE_S with SSL (valid slope); for each invalid slope: generate a SISI interrupt; send missing STATE INT when TS_S_CHECK time is reached and set MS=1 (missing STATE bits) in that case; do not leave step 21 while no valid STATE appears.</p> <p>When a valid STATE appears: store the actual position stamp at PSSM (value at the STATE event) update RAM by equation DPLL-7c (see Chapter 26.16.7.5); store the actual INC_CNT1/2 at MP1/MP2 respectively as missing pulses (instead of calculations in step 25)</p> <p>store all relevant configuration bits X of the DPLL_CTRL(0,1) Registers in shadow registers and consider them for all corresponding calculations of steps 22 to 37 accordingly; the relevant bits are explained in the registers itself for FSD=0: set PSSC=PSSM set FSD (first STATE detected) do not increment PSSC for (RMO=1 and SMC=0) and SGE1=1: increment INC_CNT1 by MLS1+MPVAL1**) for (RMO=1 and SMC=1) and SGE2=1: increment INC_CNT2 by MLS2+MPVAL2**)</p>	<p>Depending on SSL, SEN, DEN the leaving of step 21 one is done with the next STATE input;</p> <p>for the steps 22-37: for SMC=1 replace: MLS1 by MLS2, LOCK1 by LOCK2; SUB_INC1 by SUB_INC2; CNT_NUM1 by CNT_NUM2; MPVAL1 by MPVAL2; EN_C1u by EN_C2u;</p> <p>dir_crement does mean: increment for DIR2=0 decrement for DIR2=1 or DIR1 respectively</p> <p>*) target number of pulses of the last increment (see step 25ff)</p> <p>** add MPVAL1 or MPVAL2 only once, that means as long as PCM1 or PCM2 is set respectively</p>

Generic Timer Module (GTM)
Table 26-48 State description of the State Machine (cont'd)

Step	Description	Comments
21	<p>for SYS=0, FSD =1 and SOR=0: dir_crement PSSC by NMB_S_TAR*) for (SMC=0 and SGE1=1) or (SMC=1 and SGE2=1) increment INC_CNT1 by MLS1+MPVAL1**) (for SMC=0, SGE1=1 and RMO=1); increment INC_CNT2 by MLS2+MPVAL2**) (for SMC=1, SGE2=1 and RMO=1); dir_crement APS and APS_1C3</p> <p>for SYS=1 and SOR=0: dir_crement APS and APS_1C3 dir_crement APS_1C2 by SYN_S_OLD for RMO=1 and SMC=0: for SGE1_delay***)=1 dir_crement PSSC by NMB_S_TAR*); for SGE1=1 increment INC_CNT1 by SYN_S*MLS1 + PD_S_store + MPVAL1**) for RMO=1 and SMC=1: for SGE2_delay****)=1 dir_crement PSSC by NMB_S_TAR*); for SGE2=1 increment INC_CNT2 by SYN_S*MLS2 + PD_S_store***** +MPVAL2**) within the DPLL_STATUS register: set LOCK1 or 2 bit accordingly;</p>	<p>***) SGE1_delay is the value of SGE1 delayed by one valid STATE event</p> <p>****) SGE2_delay is the value of SGE2 delayed by one valid STATE event</p> <p>*****) PD_S_store=0 for AMS=0 (see DPLL_CTRL_0 register)</p>
22	<p>calculate TS_S according to equations DPLL-6a; calculate DT_S_ACT = TS_S - TS_S_old calculate RDT_S_ACT calculate QDT_SX</p>	
23	<p>send CDSI interrupt; calculate EDT_S and MEDT_S according to equations DPLL-8 and DPLL-9 for RMO=0: go back to step 21 for RMO=0 and update SYN_S and PD_S_store using the current ADT_S[i] values in that case;</p>	<p>Note: There are different behaviours of Reference Model and hardware: For the hardware the values of SYN_T and PD_store are not updated until a new valid TRIGGER slope occurs.</p>
24	<p>calculate CDT_SX according to equation DPLL-10a and b;</p>	<p>only for RMO=1</p>

Generic Timer Module (GTM)
Table 26-48 State description of the State Machine (cont'd)

Step	Description	Comments
25	calculate missing pulses: - for TBU_CH1: $MP1 = INC_CNT1(\text{valid STATE slope})$ - for TBU_CH2: $MP2 = INC_CNT2(\text{valid STATE slope})$ calculate target number of pulses: $NMB_S_TAR = MLS1 * SYN_S + PD_S_store + MPVAL1$ (for SMC=0) $NMB_S_TAR = MLS2 * SYN_S + PD_S_store + MPVAL2$ (for SMC=1) (instead of PD_S_store use zero in the case AMS=0)	only for RMO=1 for SMC=0 instead of MPVAL1=0 use zero for PCM1=0 for SMC=1 instead of MPVAL2=0 use zero for PCM2=0; add MPVAL1/2 once to INC_CNT1/2 and reset PCM1/2 after that
26	sent MPx with highest possible frequency and set $NMB_S = NMB_S_TAR$	only for RMO=1, DMO=0 and COA=0
27	calculate number of pulses to be sent according to DPLL-22 or $NMB_S = NMB_S_TAR + MPx$	only for RMO=1, DMO=0 and COA=1
28	$NMB_S = SYN_S * CNT_NUM1$ (SMC=0) $NMB_S = SYN_S * CNT_NUM2$ (SMC=1)	only for RMO=1, DMO=1
29	update SYN_S and PD_S_store;	Note: There are different behaviours of Reference Model and hardware: For the hardware the values of SYN_T and PD_store are not updated until a new valid TRIGGER slope occurs.

Generic Timer Module (GTM)
Table 26-48 State description of the State Machine (cont'd)

Step	Description	Comments
30	calculate ADD_IN_CAL2 according to equation DPLL-26 and DPLL-26b or DPLL-31 respectively and store this value in RAM use ADD_IN_CAL2 as ADD_IN value for the case DLM=0 use ADD_IN_LD2 as ADD_IN for the case DLM=1, but do this update immediately (without waiting for this step 30); for RMO=1, DMO=DLM=0 and EN_C1u=0 (EN_C1u=0): reset the FlipFlops in the SUB_INC1 or SUB_INC2 generator respectively; start sending SUB_INC1 / SUB_INC2;	only for RMO=1 for DLM=0 for DLM=1
31	calculate $TS_S_CHECK = TS_S + DT_S_ACT * (TOV_S);$	only for RMO=1;
32	automatic setting of actions masking bits in the DPLL_STATUS register: CAIP1 and CAIP2 for SMC=0 only CAIP2 for SMC=1	for RMO=1
33	for all actions with ACT_N[i]=0 calculate: $NA[i] = (PSA[i] - PSSC)/MLS1$ for forward direction with w = integer part and b = remainder of the division (fractional part) for backward direction use $NA[i] = (PSSC - PSA[i])/(MLS1)$ and consider in both cases the time base overflow in order to get a positive difference use MLS2 as divider in the case of SMC=1	for SMC=0: 24 actions, for SMC=1: 12 actions; depending on ACT_Ni in DPLL_ACT_STA register
34	calculate PDT_S[i] and DTA[i] for up to 24 action values according to equations DPLL-13 and DPLL-14;	only for RMO=1; for SMC=0 actions 0...23 for SMC=1 actions 1223
35	calculate TSAC[i] according to equation DPLL-18 and PSAC[i] according to equation DPLL-20	for the relevant actions (see above) and RMO=1

Generic Timer Module (GTM)

Table 26-48 State description of the State Machine (cont'd)

Step	Description	Comments
36	automatic reset of the actions masking bit CAIP in the DPLL_STATUS register: CAIP1=CAIP2=0 for SMC=0 and only CAIP2=0 for SMC=1 set the corresponding ACT_Ni bits in the DPLL_ACT_STA register	for the relevant actions (see above) and RMO=1 Set ACT_Ni and reset ACT_WRi for all enabled actions
37	check the duration of the last increment to its predecessor according to the profile and taking into account TOV_S: set the ISN status bit and reset the corresponding LOCK bit, when not plausible; go to step 38, when no valid STATE appears for all following steps 38 to 40: go immediately back to step 21, when a valid STATE event occurs, interrupt all calculations there and reset all CAIPx in that case; when going back to step 21: store TS_S in RAM 1C2 according to APS_1C2; update RAM 1C1 and RAM 1C4	for all conditions
38	wait for a new PMTR value; set the corresponding CAIPx values and go to step 39 in that case	go immediately to step 21 and update the RAM according to step 37 when a valid STATE event occurs
39	make the requested action calculation according to new PMTR values	go immediately to step 21 and update the RAM according to step 37 when a valid STATE event occurs
40	reset CAIP and go back to step 38	go immediately to step 21 and update the RAM according to step 37 when a valid STATE event occurs

26.16.9 DPLL Interrupt signals

The DPLL provides 27 interrupt lines. These interrupts are shown below.

26.16.9.1 DPLL Interrupt signals

interrupt signals of table

Table 26-49 DPLL Interrupt signals

Signal	Description
DPLL_SORI_IRQ	STATE is out of range
DPLL_TORI_IRQ	TRIGGER is out of range
DPLL_CDSI_IRQ	STATE duration calculated for last increment
DPLL_CDTI_IRQ	TRIGGER duration calculated for last increment
DPLL_TE4_IRQ	TRIGGER event interrupt 4 request ¹⁾
DPLL_TE3_IRQ	TRIGGER event interrupt 3 request ¹⁾
DPLL_TE2_IRQ	TRIGGER event interrupt 2 request ¹⁾
DPLL_TE1_IRQ	TRIGGER event interrupt 1 request ¹⁾
DPLL_TE0_IRQ	TRIGGER event interrupt 0 request ¹⁾
DPLL_LL2_IRQ	Lost of lock interrupt for SUB_INC2 request
DPLL_GL2_IRQ	Get of lock interrupt for SUB_INC2 request
DPLL_E_IRQ	Error interrupt request
DPLL_LL1_IRQ	Lost of lock interrupt for SUB_INC1 request
DPLL_GL1_IRQ	Get of lock interrupt for SUB_INC1 request
DPLL_W1_IRQ	Write access to RAM region 1B or 1C interrupt request
DPLL_W2_IRQ	Write access to RAM region 2 interrupt request
DPLL_PW_IRQ	Plausibility window violation interrupt of TRIGGER request
DPLL_TAS_IRQ	TRIGGER active slope while NTI_CNT is zero interrupt request
DPLL_SAS_IRQ	STATE active slope interrupt request
DPLL_MT_IRQ	Missing TRIGGER interrupt request
DPLL_MS_IRQ	Missing STATE interrupt request
DPLL_TIS_IRQ	TRIGGER inactive slope interrupt request
DPLL_SIS_IRQ	STATE inactive slope interrupt request
DPLL_TAX_IRQ	TRIGGER maximum hold time violation interrupt request
DPLL_TIN_IRQ	TRIGGER minimum hold time violation interrupt request

Generic Timer Module (GTM)

Table 26-49 DPLL Interrupt signals (cont'd)

Signal	Description
DPLL_PE_IRQ	DPLL enable interrupt request
DPLL_PD_IRQ	DPLL disable interrupt request

1) see TINT value in the corresponding ADT_T[i] section of RAM region 2

Note: TEi_IRQ depends on the TINT value in ADT_T[i]¹⁾ and is only active when SYT²⁾

1) see RAM region 2 explanations

2) see DPLL STATUS register

26.16.10 DPLL Register overview
Table 26-50 DPLL register overview

Address offset	Name	Description	Init value	Page
0x0000	DPLL_CTRL_0	DPLL Control Register 0	0x003B_BA57	26-501
0x0004	DPLL_CTRL_1	DPLL Control Register 1	0xB000_0000	26-505
0x0008	DPLL_CTRL_2	DPLL Control Register 2 (actions 0-7 enable)	0x0000_0000	26-513
0x000C	DPLL_CTRL_3	DPLL Control Register 3 (actions 8-15 enable)	0x0000_0000	26-515
0x0010	DPLL_CTRL_4	DPLL Control Register 4 (actions 16-23 enable)	0x0000_0000	26-517
0x0018	DPLL_ACT_STA	DPLL ACTION Status Register with connected shadow register	0x0000_0000	26-519
0x001C	DPLL_OSW	DPLL Offset and switch old/new address register	0x0000_0200	26-521
0x0020	DPLL_AOSV_2	DPLL Address offset register for APT in RAM region 2	0x1810_0800	26-523
0x0024	DPLL_APT	DPLL Actual RAM pointer to RAM regions 2A, b and d	0x0000_0000	26-524
0x0028	DPLL_APS	DPLL Actual RAM pointer to RAM regions 1C1, 1C2 and 1C4	0x0000_0000	26-527
0x002C	DPLL_APT_2C	DPLL Actual RAM pointer to RAM region 2C	0x0000_0000	26-530
0x0030	DPLL_APS_1C3	DPLL Actual RAM pointer to RAM region 1C3	0x0000_0000	26-532
0x0034	DPLL_NUTC	DPLL Number of recent TRIGGER events used for calculations (mod 2*(TNU +1-SYN_NT))	0x0001_2001	26-533

Generic Timer Module (GTM)
Table 26-50 DPLL register overview (cont'd)

Address offset	Name	Description	Init value	Page
0x0038	DPLL_NUSC	Number of recent STATE events used for calculations (e.g. $\text{mod } 2 * (\text{SNU} + 1 - \text{SYN_NS})$ for $\text{SYSF}=0$)	0x0001_2081	26-536
0x003C	DPLL_NTI_CNT	DPLL Number of active TRIGGER events to interrupt	0x0000_0000	26-539
0x0040	DPLL_IRQ_NOTIFY	DPLL Interrupt notification register	0x0000_0000	26-540
0x0044	DPLL_IRQ_EN	DPLL Interrupt enable register	0x0000_0000	26-543
0x0048	DPLL_IRQ_FORCE_CINT	DPLL Interrupt force register	0x0000_0000	26-547
0x004C	DPLL_IRQ_MODE	DPLL Interrupt mode register	0x0000_0000	26-549
0x0050	DPLL_EIRQ_EN	DPLL Error Interrupt enable register	0x0000_0000	
0x00B0	INC_CNT1	DPLL Counter for pulses for TBU_TS1 to be sent in automatic end mode	0x0000_0000	26-554
0x00B4	INC_CNT2	DPLL Counter for pulses for TBU_TS2 to be sent in automatic end mode when $\text{SMC}=\text{RMO}=1$	0x0000_0000	26-555
0x00B8	DPLL_APT_SYNC	DPLL old RAM pointer and offset value for TRIGGER	0x0000_0000	26-556
0x00BC	DPLL_APS_SYNC	DPLL old RAM pointer and offset value for STATE	0x0000_0000	26-559
0x00C0	TBU_TS0_T	DPLL TBU_TS0 value at last TRIGGER event	0x0000_0000	26-562
0x00C4	TBU_TS0_S	DPLL TBU_TS0 value at last STATE event	0x0000_0000	26-562
0x00C8	ADD_IN_LD1	DPLL direct load input value for SUB_INC1	0x0000_0000	26-564

Table 26-50 DPLL register overview (cont'd)

Address offset	Name	Description	Init value	Page
0x00CC	ADD_IN_LD2	DPLL direct load input value for SUB_INC2	0x0000_0000	26-565
0x00FC	DPLL_STATUS	Status register	0x0000_0000	
0x0100 + x * 0x04	DPLL_ID_PMTR_x	DPLL 9-bit ID Information For Input Signal PMT_x[8:0] Register (x:023)	0x0000_01FE	26-573
0x01E0	DPLL_CTRL_0_SHADOW_TRIGGER	shadow register of DPLL_CTRL_0	0x0000_0000	26-574
0x01E4	DPLL_CTRL_0_SHADOW_STATE	shadow register of DPLL_CTRL_0	0x0000_0000	26-576
0x01E8	DPLL_CTRL_1_SHADOW_TRIGGER	shadow register of DPLL_CTRL_1	0x0000_0000	26-577
0x01EC	DPLL_CTRL_1_SHADOW_STATE	shadow register of DPLL_CTRL_1	0x0000_0000	26-578
0x01FC	DPLL_RAM_INI	initialization control and status for RAMs	0x0000_0000	26-580

RAM Region 1 map description.

Table 26-51 RAM Region 1 map description

Name	Description	Address offset in relation to DPLL start address	Page
RAM Region 1A 288 bytes for 96 words of 24 Bits; this RAM is only accessible for DEN = 0		0x0200-0x037C	
PSA[i]	ACTION_i Position/Value action request register (i:023)	0x0200 + i * 0x04	26-581
DLA[i]	ACTION_i time to react before PSAi register (i:023)	0x0280 + i * 0x04	26-581

Generic Timer Module (GTM)
Table 26-51 RAM Region 1 map description (cont'd)

Name	Description	Address offset in relation to DPLL start address	Page
NA[i]	Calculated number of TRIGGER/STATE increments to ACTION_i (i:023)	0x0300 + i * 0x04	26-582
DTA[i]	Calculated Relative Time To ACTION_i Register (i:023)	0x0380 + i * 0x04	26-584
RAM Region 1B		0x0400-0x05FC	
Note: the following registers for variables are located in RAM Region 1B, read access by AEI via bus interface possible, writing results in an interrupt; data width of 3 Bytes used for 24 bit values		288 bytes for 96 words of 24 Bits	
TRIGGER signal information stored			
TS_T	Actual signal TRIGGER time stamp register	0x0400/ 0x0404	26-584
TS_T_OLD	Previous signal TRIGGER time stamp register	0x0404/ 0x0400	26-586
Note: the switch of the LSB address bits is performed using the SWON register at 0x0020		0x0400 ... 0x0404	
FTV_T	Actual signal TRIGGER filter value register	0x0408	26-587
do not use		0x040C	
Note: the switch of the LSB address bits is performed using the SWON register at 0x0020		0x0400 ... 0x040C	
STATE signal information stored			
TS_S	Actual signal TRIGGER time stamp register	0x0410/ 0x0414	26-588
TS_S_OLD	Previous signal STATE time stamp register	0x0414/ 0x0410	26-589

Generic Timer Module (GTM)
Table 26-51 RAM Region 1 map description (cont'd)

Name	Description	Address offset in relation to DPLL start address	Page
	Note: The switch of the LSB address bits is performed using the SWON register at 0x0020.	0x0410 ... 0x0414	
FTV_S	Actual signal STATE filter value register do not use	0x0418 0x041C	26-590
THMI	TRIGGER hold time min value	0x0420	26-591
THMA	TRIGGER hold time max value	0x0424	26-592
THVAL	measured last pulse time from valid to invalid TRIGGER slope do not use	0x0428 0x042C	26-592
TOV	Time out value of TRIGGER, according to the last nominal increment for a missing TRIGGER	0x0430	26-594
TOV_S	Time out value of STATE, according to the last nominal increment for a missing STATE	0x0434	26-595
ADD_IN_C AL1	calculated ADD_IN value for SUB_INC1 generation	0x0438	26-596
ADD_IN_C AL2	calculated ADD_IN value for SUB_INC2 generation	0x043C	26-596
MPVAL1	missing pulses to be added/subtracted directly to SUB_INC1 and INC_CNT1 once	0x0440	26-598
MPVAL2	missing pulses to be added/subtracted directly to SUB_INC2 and INC_CNT2 once	0x0444	26-599
NMB_T_TA R	target number of TRIGGER pulses	0x0448	26-599
NMB_T_TA R_OLD	target number of TRIGGER pulses	0x044C	26-599
NMB_S_TA R	target number of STATE pulses	0x0450	26-599
NMB_S_TA R_OLD	target number of STATE pulses	0x0454	26-599

Generic Timer Module (GTM)
Table 26-51 RAM Region 1 map description (cont'd)

Name	Description	Address offset in relation to DPLL start address	Page
	do not use	0x0458 0x045C	
RCDT_TX	reciprocal value of expected increment duration (T)	0x0460	26-604
RCDT_SX	reciprocal value of expected increment duration (S)	0x0464	26-604
RCDT_TX_NOM	reciprocal value of the expected nominal increment duration (T)	0x0468	26-606
RCDT_SX_NOM	reciprocal value of the expected nominal increment duration (S)	0x046C	26-606
RDT_T_ACT	actual reciprocal value of TRIGGER	0x0470	26-608
RDT_S_ACT	actual reciprocal value of STATE	0x0474	26-608
DT_T_ACT	Duration of last TRIGGER increment	0x0478	26-610
DT_S_ACT	Duration of last STATE increment	0x047C	26-610
	Calculated immediate values (eq. DPLL-1 to DPLL-10)		
EDT_T	Absolute error of prediction for last TRIGGER increment	0x0480	26-612
MEDT_T	Average absolute error of prediction up to the last TRIGGER increment	0x0484	26-612
EDT_S	absolute error of prediction for last STATE increment	0x0488	26-614
MEDT_S	Average absolute error of prediction up to the last STATE increment	0x048C	26-614
CDT_TX	Expected duration of current TRIGGER increment	0x0490	26-616
CDT_SX	Expected duration of current STATE increment	0x0494	26-616
CDT_TX_NOM	Expected nominal duration of current TRIGGER increment (without consideration of missing events)	0x0498	26-617
CDT_SX_NOM	Expected nominal duration of current STATE increment (without consideration of missing events)	0x049C	26-617

Generic Timer Module (GTM)
Table 26-51 RAM Region 1 map description (cont'd)

Name	Description	Address offset in relation to DPLL start address	Page
TLR	TRIGGER locking range value; the TOR bit in the DPLL_STATUS register is set when violated	0x04A0	26-618
SLR	STATE locking range value; the SOR bit in the DPLL_STATUS register is set when violated	0x04A4	26-618
	Relations of the sum of prediction increments to the reference increment in the past (see equations DPLL-11 or DPLL-13 for calculation)		
PDT_Ti	predicted time to Action_i (i:023)	0x0500 + i * 0x04	26-620
MLS1	Calculated number of sub-pulses between two STATE events (to be set by CPU)	0x05C0	26-621
MLS2	Calculated number of sub-pulses between two STATE events (to be set by CPU) for the use when SMC=RMO=1	0x05C4	26-622
CNT_NUM 1	Number of sub-pulses of SUB_INC1 in continuous mode, updated by the host only	0x05C8	26-623
CNT_NUM 2	Number of sub-pulses of SUB_INC2 in continuous mode, updated by the host only	0x05CC	26-623
PVT	Plausibility value of next active TRIGGER slope	0x05D0	26-624
	do not use	0x05D4 0x05DC	
PSTC	Accurate calculated position stamp of last TRIGGER input	0x05E0	26-625
PSSC	Accurate calculated position stamp of last STATE input	0x05E4	26-626
PSTM	Measured position stamp of last TRIGGER input	0x05E8/ 0x05EC	26-627
PSTM_OLD	Measured position stamp of last but one TRIGGER input	0x05EC/ 0x05E8	26-628

Generic Timer Module (GTM)
Table 26-51 RAM Region 1 map description (cont'd)

Name	Description	Address offset in relation to DPLL start address	Page
PSSM	Measured position stamp of last STATE input	0x05F0/ 0x05F4	26-629
PSSM_OL D	Measured position stamp of last but one STATE input	0x05F4/ 0x05F0	26-630
NMB_T	Number of pulses of current increment in normal mode for SUB_INC1(see equation DPLL-21 or for SMC=1 equation DPLL-27 respectively)	0x05F8	26-631
NMB_S	Number of pulses of current increment in emergency mod for SUB_INC1 (see equation DPLL-22) or in the case SMC=1 for SUB_INC2 (see equation DPLL-28)	0x05FC	26-631
RAM Region 1C		0x0600 – 0x09FC	
<p>Note: the following registers for the signal STATE are located in RAM Region 1C, read access by AEI via bus interface possible, writing results in an interrupt; data width of 3 Bytes used for 24 bit values</p>		0,75 Kbytes for 256 words of 24 Bits	
1C1	Reciprocal values of the corresponding successive increments RDT_S[i] (see equations DPLL-6a,b); the values are calculated using the recent NUSE increments (see NUSC register at address 0x0038)		
RDT_Sx	RDT_Sx (x:063)	0x0600 + x* 0x04	26-632
1C2	Time stamp field for STATE events		
TSF_Sx	TSF_Sx (x:063)	0x0700 + x* 0x04	26-633
1C3	Adapt values for the current STATE increment; time stamp values bits 24-27 in addition to the corresponding 24 bit value of TSF_Sx above, stored in bits 23:20 of ADT_S[i];		

Generic Timer Module (GTM)
Table 26-51 RAM Region 1 map description (cont'd)

Name	Description	Address offset in relation to DPLL start address	Page
ADT_Sx	ADT_Sx (x:063)	0x0800 + x* 0x04	26-633
1C4	Uncorrected last increment value of STATE (DT_S) for FULL_SCALE; measuring data of increments without corrections used for the CPU to generate ADT_S values		
DT_Sx	DT_Sx (x:063)	0x0900 + x* 0x04	26-634

Table 26-52 Register Region EXT map description

Address offset in relation to DPLL start address	Name	Description	Init value
0x0E00 - 0x0F1C	Register Region EXT	extension of register region in order to allow up to 24 action calculations	
0x0E00 + x * 0x04	DPLL_TSACx	calculated action time stamps for actions (x:023)	0x007F_FFFF
0x0E80 + x * 0x04	DPLL_PSACx	calculated action position stamps for actions (x:023)	0x007F_FFFF
0x0F00	DPLL_ACB_0	control bits for actions 03	0x0000_0000
0x0F04	DPLL_ACB_1	control bits for actions 47	0x0000_0000
0x0F08	DPLL_ACB_2	control bits for actions 811	0x0000_0000
0x0F0C	DPLL_ACB_3	control bits for actions 1215	0x0000_0000

Generic Timer Module (GTM)

Table 26-52 Register Region EXT map description (cont'd)

Address offset in relation to DPLL start address	Name	Description	Init value
0x0F10	DPLL_ACB_4	control bits for actions 1619	0x0000_0000
0x0F14	DPLL_ACB_5	control bits for actions 2023	0x0000_0000

Generic Timer Module (GTM)

RAM Region 2 map description.

Table 26-53 RAM Region 2 Map

Name	Description	Address offset in relation to DPLL start address	Page
RAM Region 2		0x4000-0x4FFC	
	NOTE: the following registers for the signal TRIGGER are located in RAM region 2, read access by AEI via bus interface possible, write access results in an interrupt, when enabled; data width of 3 bytes used for 24 bit values The RAM field part contains 256 values	size 3 Kbytes for word sizes of 24 Bits	
Region 2A	Reciprocal values of the corresponding successive increments RDT_T[i] (see equations DPLL-2a,b); address offsets are given by the AOSV_2A	AOSV_2A values are addresses after shift left by 8	
RDT_Tx	RDT_Tx (x:0255)	AOSV_2A + x*0x04	26-639
Region 2B	Time stamp field for all TRIGGER events in FULL_SCALE; 24 bit time stamp values		
TSF_Tx	TSF_Tx (x:0255)	AOSV_2B + x*0x04	26-639
Region 2C	ADT_T values to correct the measured TRIGGER signal values; time stamp values bits 24-27 in addition to the 24 bit value of TSF_Tx, stored in the bits 23:20 of ADT_T[i]		
ADT_Tx	ADT_Tx (x:0255)	AOSV_2C + x*0x04	26-641

Generic Timer Module (GTM)

Table 26-53 RAM Region 2 Map (cont'd)

Name	Description	Address offset in relation to DPLL start address	Page
Region 2D	Uncorrected last increment values of TRIGGER (DT_T); measuring raw data of increments		
DT_Tx	DT_Tx (x:0255)	AOSV_2 D + x* 0x04	26-642

26.16.11 DPLL Register and Memory description

All of the following registers are 32-bit only accessible.

Description of registers beginning from register DPLL_CTRL_0.

Description of RAM regions beginning from RAM 1A (see below):

Bits 31 down to 24 in each RAM region are not implemented and therefore always read as zero (reserved). Other bits which are declared as reserved are not protected against writing. Reserved address regions are not protected against writing.

Description of memory region RAM 1A beginning from memory PSAi

The RAM region 1A is writable only for DEN=0 (see DPLL_CTRL_1 register).

Description of memory region RAM 1B beginning from memory TS_T.

Description of memory region RAM 1C beginning from memory RDT_S.

Description of memory region RAM 2 beginning from memory RDT_T.

26.16.11.1 Register DPLL_CTRL_0

Control register 0.

GTM_DPLL_CTRL_0

DPLL Control Register 0

(28000_H)

Reset Value: 003BBA57_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
RMO	TEN	SEN	IDT	IDS	AMT	AMS					TNU					
rw	rw	rw	rw	rw	rw	rw					rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	SNU				IFP					MLT						
	rw				rw					rw						

Generic Timer Module (GTM)

Field	Bits	Type	Description
MLT¹⁾	[9:0]	rw	<p>Multiplier for TRIGGER</p> <p>MLT+1 is number of SUB_INC1 pulses between two TRIGGER events in normal mode (1...1024); For emergency mode the number of SUB_INC1 pulses between two STATE events is calculated by the CPU using the formula $MLS1=(MLT+1) * (TNU+1) / (SNU+1)$ in order to get the same number of SUB_INC1 pulses for FULL_SCALE. This value is stored in RAM at 0x05C0. Change of MLT by the CPU must result in the corresponding change of MLS1 by the CPU for SMC=0. Note: The number of MLT events is the binary value plus 1. The value MLT+1 is replaced by MLS1 in the case of SMC=1 (see DPLL_CTRL_1 register) for all relevant calculations.</p>
IFP¹⁾²⁾³⁾	10	rw	<p>Input filter position</p> <p>Value contains position or time related information.</p> <p>0_B TRIGGER_FT and STATE_FT mean time related values, that means the number of time stamp clocks</p> <p>1_B TRIGGER_FT and STATE_FT mean position related values, that means the number of SUB_INC1 (or SUB_INC2 in the case SMC=1) pulses respectively</p>
SNU⁴⁾	[15:11]	rw	<p>STATE number</p> <p>SNU+1 is number of STATE events in HALF_SCALE (1...32). Note: The number of nominal STATE events is the binary value plus 1. Set by DEN=0 only. This value can only be written when the DPLL is disabled.</p>
TNU⁴⁾	[24:16]	rw	<p>TRIGGER number</p> <p>TNU+1 is number of TRIGGER events in HALF_SCALE (1...512). Note: The number of nominal TRIGGER events is the binary value plus 1. Set by DEN=0 only. This value can only be written when the DPLL is disabled.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
AMS²⁾	25	rw	Adapt mode STATE Use of adaptation information of STATE. 0 _B No adaptation information is used for STATE 1 _B Immediate adapting mode; the values ADT_S[i] are considered to calculate SUB_INC1 pulses in emergency mode (SMC=0) or SUB_INC2 pulses for SMC=1
AMT¹⁾	26	rw	Adapt mode TRIGGER Use of adaptation information of TRIGGER. 0 _B No adaptation information for TRIGGER is used 1 _B Immediate adapting mode; the values ADT_T[i] are considered to calculate the SUB_INC1 pulses in normal mode and for SMC=1
IDS²⁾	27	rw	Input delay STATE Use of input delay information transmitted in FT part of the STATE signal. 0 _B Delay information is not used 1 _B Up to 24 bits of the FT part contain the delay value of the input signal, concerning the corresponding edge
IDT¹⁾	28	rw	Input delay TRIGGER use of input delay information transmitted in FT part of the TRIGGER signal. 0 _B Delay information is not used 1 _B Up to 24 bits of the FT part contain the delay value of the input signal, concerning the corresponding edge
SEN	29	rw	STATE enable 0 _B STATE signal is not enabled (no signal considered) 1 _B STATE signal is enabled
TEN	30	rw	TRIGGER enable 0 _B TRIGGER signal is not enabled (no signal considered) 1 _B TRIGGER signal is enabled

Generic Timer Module (GTM)

Field	Bits	Type	Description
RMO ¹⁾²⁾	31	rw	<p>Reference mode</p> <p>Selection of the relevant the input signal for generation of SUB_INC1.</p> <p>0_B Normal mode; the signal TRIGGER is used to generate the SUB_INC1 signals</p> <p>1_B Emergency mode for SMC=0; signal STATE is used to generate the SUB_INC1 signals; Double synchronous mode for SMC=1: signal TRIGGER is used to generate the SUB_INC1 signals and STATE is used to generate the SUB_INC2 signals</p> <p>Note: for SMC=0: TRIGGER and STATE are prepared to calculate SUB_INC1. The RMO bit gives a decision only, which of them is used.</p> <p>For changing from normal mode to emergency mode at the following TRIGGER slope (according to the RMO value in the shadow register)1) the PSSC value is calculated by $PSSC = PSSM + \text{correction_value}$ (forward direction) or $PSSC = PSSM - \text{correction_value}$ (backward direction) with the $\text{correction_value} = \text{inc_cnt1} - \text{nmb_t}$.</p> <p>For changing from emergency mode to normal mode at the following STATE slope (according to the RMO value in the shadow register)2) the PSTC value is calculated by $PSTC = PSTM + \text{correction_value}$ (forward direction) or $PSTC = PSTM - \text{correction_value}$ (backward direction) with the $\text{correction_value} = \text{inc_cnt1} - \text{nmb_s}$.</p> <p>In the case of no further TRIGGER or STATE events appearing the CPU has to perform the corrections above accordingly.</p>

- 1) stored in an independent shadow register for a valid TRIGGER event and for DEN = 1.
- 2) stored in an independent shadow register for a valid STATE event and for DEN = 1.
- 3) for IFP=1 the time between two valid TRIGGER or STATE events must be always greater then 2,34 μs and the value x of MLT, MLS1 or MLS2 must be chosen such that the number of time stamps pules between two SUB_INC events must be less then 65536. This is fulfilled when x is greater then 256.
- 4) the time between two valid STATE or TRIGGER events must be always greater then 23,4 ms; in addition the TS_CLK and the resolution must be chosen such that for each nominal increment the time stamps differ at least in the value of 257.

26.16.11.2 Register DPLL_CTRL_1

Control register 1.

GTM_DPLL_CTRL_1
DPLL Control Register 1
(28004_H)
Reset Value: B0000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSL		SSL		SMC	TS0_HRT	TS0_HRS	SYS_F	SWR	LCD	Reserved	SYN_NT				
rw		rw		rw	rw	rw	rw	w	rw	r	rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SYN_NS				PCM_2	DLM_2	SGE_2	PCM_1	DLM_1	SGE_1	PIT	COA	IDDS	DEN	DMO	
rw				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
DMO ^{1) 2)}	0	rw	DPLL mode select 0 _B Automatic end mode; if the number of pulses for a increment is reached, no further pulse is generated until the next valid TRIGGER/STATE is received; in the case of getting a new valid TRIGGER/STATE before the defined number of pulses is reached, the pulse frequency is changed according to the conditions described below (COA) 1 _B Continuous mode; in this mode a difference between the predefined number of pulses and the actual number of generated pulses can influence the pulse frequency by writing a corresponding pulse number into CNT_NUM1 or CNT_NUM2 respectively in RAM region 1B.

Generic Timer Module (GTM)

Field	Bits	Type	Description
DEN	1	rw	<p>DPLL enable</p> <p>0_B The DPLL is not enabled; Disabling the DPLL will result in a reset state of the DPLL_STATUS register which remains in this state until DEN=1. No DPLL related interrupt will be generated in that case.</p> <p>1_B The DPLL is enabled;</p> <p><i>Note:</i> The bits 31 down to 0 of the DPLL STATUS register are cleared, when the DPLL is disabled. Some bits of the control registers can be set only when DEN=0. The protected bits in the DPLL_CTRL_1 register can not be written when simultaneously DEN is set to 1.</p>
IDDS	2	rw	<p>Input direction detection strategy in case of SMC = 0</p> <p>0_B The input direction is detected comparing the THMI value with the duration between valid and invalid slope of TRIGGER</p> <p>1_B The input direction is detected using TDIR input signal also in the case SMC=0</p> <p><i>Note:</i> This bit can only be written when the DPLL is disabled and be fixed to zero, when not needed for an implementation. Independent of the value of IDDS is the direction information for TRIGGER in the case SMC=0 always considered at the moment when the invalid slope appears.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
COA ¹⁾²⁾	3	rw	<p>Correction strategy in automatic end mode (DMO=0)</p> <p>0_B The maximum pulse frequency of the system clock will be used to make up for missing pulses from last increment; the output of the calculated new pulses will be delayed accordingly and the FFs in the pulse generation unit will be reset before sending new pulses</p> <p>1_B missing pulses of the last increment are distributed evenly to the next increment, calculations are done when the next valid input event appears. The number of missing sub-pulses will be determined by the pulse counter difference between the last two valid TRIGGER/STATE events respectively; the FFs in the pulse generation unit are not reset before sending new pulses.</p> <p>Note: For SMC=RMO=1: COA is used for SUB_INC1 and SUB_INC2.</p>
PIT ¹⁾	4	rw	<p>Plausibility value PVT to next valid TRIGGER is time related</p> <p>0_B the plausibility value is position related (PVT contains the number of SUB_INC1 pulses)</p> <p>1_B the plausibility value is time related (the PVT value is to be multiplied with the duration of the last increment DT_T_ACT and divided by 1024)</p>
SGE1 ¹⁾²⁾	5	rw	<p>SUB_INC1 generator enable</p> <p>0_B The SUB_INC1 generator is not enabled</p> <p>1_B The SUB_INC1 generator is enabled</p>
DLM1 ¹⁾²⁾	6	rw	<p>Direct Load Mode for SUB_INC1 generation</p> <p>0_B the DPLL uses the calculated ADD_IN_CAL value for the SUB_INC1 generation</p> <p>1_B the ADD_IN_LD value is used for the SUB_INC1 generation and is provided by the CPU; the value remains valid until the CPU writes a new one; the calculated ADD_IN values are stored as ADD_IN_CAL in the RAM at different locations for normal and emergency mode</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
PCM1 ¹⁾²⁾³⁾	7	rw	Pulse Correction Mode for SUB_INC1 generation 0 _B the DPLL does not use the correction value stored in MPVAL1 1 _B the DPLL uses the correction value stored in MPVAL1 in normal and emergency mode
SGE2 ²⁾	8	rw	SUB_INC2 generator enable 0 _B The SUB_INC2 generator is not enabled 1 _B The SUB_INC2 generator is enabled
DLM2 ²⁾	9	rw	Direct Load Mode for SUB_INC2 generation 0 _B the DPLL uses the calculated ADD_IN_CAL value for the SUB_INC2 generation 1 _B the ADD_IN_LD value is used for the SUB_INC2 generation and is provided by the CPU; the value remains valid until the CPU writes a new one; the calculated ADD_IN values are stored as ADD_IN_CAL in the RAM at different locations for normal and emergency mode
PCM2 ²⁾³⁾	10	rw	Pulse Correction Mode for SUB_INC2 generation 0 _B the DPLL does not use the correction value stored in MPVAL2 1 _B the DPLL uses the correction value stored in MPVAL2

Generic Timer Module (GTM)

Field	Bits	Type	Description
SYN_NS	[15:11]	rw	<p>Synchronization number of STATE</p> <p>Summarized number of virtual increments in HALF_SCALE</p> <p>sum of all systematic missing STATE events in HALF_SCALE (for SYSF=0) or FULL_SCALE (for SYSF=1); the SYN_NS missing STATES can be divided up to an arbitrary number of blocks. The pattern of events and missing events in FULL_SCALE is shown in RAM region 1C3 as value NS in addition to the adapt values. The number of stored increments in FULL_SCALE must be equal to $2*(SNU+1-SYN_NS)$ for SYSF=0 or $2*(SNU+1)-SYN_NS$ for SYSF=1. This pattern is written by the CPU beginning from a fixed reference point (maybe beginning of the FULL_SCALE region). The relation to the actual increment is established by setting of the profile RAM pointer APS_1C3 in an appropriate relation to the RAM pointer APS of the actual increment by the CPU.</p> <p>Note: This value can only be written when the DPLL is disabled.</p>
SYN_NT	[20:16]	rw	<p>Synchronization number of TRIGGER</p> <p>Summarized number of virtual increments in HALF_SCALE.</p> <p>Sum of all systematic missing TRIGGER events in HALF_SCALE; the SYN_NT missing TRIGGER can be divided up to an arbitrary number of blocks. The pattern of events and missing events in FULL_SCALE is shown in RAM region 2C as value NT in addition to the adapt values. The number of stored increments in FULL_SCALE must be equal to $2*(TNU-SYN_NT)$. This pattern is written by the CPU beginning from a fixed reference point (maybe beginning of the FULL_SCALE region). The relation to the actual increment is established by setting of the profile RAM pointer APT_2C in an appropriate relation to the RAM pointer APT of the actual increment by the CPU.</p> <p>Note: This value can only be written when the DPLL is disabled.</p>
Reserved	21	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
LCD	22	rw	<p>Locking condition definition</p> <p>0_B locking condition definition is one times missing TRIGGERS as expected by the profile in HALF_SCALE (one gap)</p> <p>1_B locking condition definition is n-1 times missing TRIGGERS as expected by the profile in HALF_SCALE (one additional tooth)</p> <p><i>Note: This bit can only be written when the DPLL is disabled and be fixed to zero, when not needed for an implementation.</i></p>
SWR	23	w	<p>Software Reset</p> <p>resets all register and internal states of the DPLL</p> <p>0_B no software reset enabled</p> <p>1_B software reset enabled</p> <p>Setting the SWR bit results only in a software reset when the DPLL is not enabled (DEN=0). Read as zero.</p>
SYSF	24	rw	<p>SYN_NS for FULL_SCALE</p> <p>The value SYN_NS does mean the sum of all systematic missing STATE events in HALF_SCALE (for SYSF=0) or FULL_SCALE (for SYSF=1)</p> <p>0_B the SYN_NS value is valid for HALF_SCALE</p> <p>1_B the SYN_NS value is valid for FULL_SCALE</p> <p><i>Note: This value can only be written when the DPLL is disabled.</i></p>
TS0_HRS	25	rw	<p>TS0_HRS</p> <p>0_B the resolution of the used DPLL input TBU_TS0 bits is equal to the STATE input time stamp resolution</p> <p>1_B the STATE input time stamps have a 8 times higher resolution as the TBU_TS0 DPLL input</p> <p><i>Note: This bit can only be written when the DPLL is disabled.</i></p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
TS0_HRT	26	rw	TS0_HRT 0 _B the resolution of the used DPLL input TBU_TS0 bits is equal to the TRIGGER input time stamp resolution 1 _B the TRIGGER input time stamps have a 8 times higher resolution as the TBU_TS0 input Note: This bit can only be written when the DPLL is disabled.
SMC	27	rw	Synchronous Motor Control 0 _B the TRIGGER and STATE inputs are used for a control to SMC 1 _B the TRIGGER input reflects a combined sensor signal for SMC and in the case of RMO=1 also STATE reflects a different combined sensor signal Note: This bit can only be written when the DPLL is disabled.
SSL	[29:28]	rw	STATE slope select Definition of active slope for signal STATE each active slope is an event defined by SNU. Set by DEN=0 only. 00 _B No slope of STATE will be used; this value makes only sense in normal mode 01 _B Low high slope will be used as active slope, only inputs with a signal value of "1" will be considered 10 _B High low slope will be used as active slope, only inputs with a signal value of "0" will be considered 11 _B Both slopes will be used as active slopes Note: This bits can only be written when the DPLL is disabled.
TSL	[31:30]	rw	Definition of active slope for signal TRIGGER each active slope is an event defined by TNU Set by DEN=0 only. 00 _B No slope of TRIGGER will be used; this value makes only sense in emergency mode 01 _B Low high slope will be used as active slope, only inputs with a signal value of "1" will be considered 10 _B High low slope will be used as active slope, only inputs with a signal value of "0" will be considered 11 _B Both slopes will be used as active slopes Note: This bits can only be written when the DPLL is disabled.

Generic Timer Module (GTM)

- 1) stored in an independent shadow register for a valid TRIGGER event and for DEN = 1.
- 2) stored in an independent shadow register for a valid STATE event and for DEN = 1.
- 3) Bit is cleared, when transmitted to shadow register

26.16.11.3 Register DPLL_CTRL_2

Action enable register.

GTM_DPLL_CTRL_2
DPLL Control Register 2
(28008_H)
Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								WAD 7	WAD 6	WAD 5	WAD 4	WAD 3	WAD 2	WAD 1	WAD 0
r								w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AEN 7	AEN 6	AEN 5	AEN 4	AEN 3	AEN 2	AEN 1	AEN 0	Reserved							
rw	rw	rw	rw	rw	rw	rw	rw	r							

Field	Bits	Type	Description
Reserved	[7:0]	r	Reserved Read as zero, should be written as zero.
AEN0	8	rw	ACTION_0 enable 0 _B the corresponding action is not enabled 1 _B the corresponding action is enabled This bit can only be written when the correspondent WAD _i bit is set. It can be set for debug purpose by CPU also, when the DPLL is disabled. The enable bit becomes active only when the DPLL is in operation (DEN = 1).
AEN1	9	rw	ACTION_1 enable see bit 8
AEN2	10	rw	ACTION_2 enable see bit 8
AEN3	11	rw	ACTION_3 enable see bit 8
AEN4	12	rw	ACTION_4 enable see bit 8
AEN5	13	rw	ACTION_5 enable see bit 8

Generic Timer Module (GTM)

Field	Bits	Type	Description
AEN6	14	rw	ACTION_6 enable see bit 8
AEN7	15	rw	ACTION_7 enable see bit 8
WAD0	16	w	Write control bit of Action_0 0_B the corresponding AENi bit is not writable 1_B the corresponding AENi bit is writable Note: For writing WADi =1 only the corresponding the AENx bits are written. The AENi bits remain unchanged when the corresponding WADi=0.
WAD1	17	w	Write control bit of Action_1 see bit 16
WAD2	18	w	Write control bit of Action_2 see bit 16
WAD3	19	w	Write control bit of Action_3 see bit 16
WAD4	20	w	Write control bit of Action_4 see bit 16
WAD5	21	w	Write control bit of Action_5 see bit 16
WAD6	22	w	Write control bit of Action_6 see bit 16
WAD7	23	w	Write control bit of Action_7 see bit 16
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.4 Register DPLL_CTRL_3

Action enable register.

GTM_DPLL_CTRL_3
DPLL Control Register 3
(2800C_H)
Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								WAD 15	WAD 14	WAD 13	WAD 12	WAD 11	WAD 10	WAD 9	WAD 8
r								w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AEN 15	AEN 14	AEN 13	AEN 12	AEN 11	AEN 10	AEN 9	AEN 8	Reserved							
rw	rw	rw	rw	rw	rw	rw	rw	r							

Field	Bits	Type	Description
Reserved	[7:0]	r	Reserved Read as zero, should be written as zero.
AEN8	8	rw	ACTION_8 enable 0 _B the corresponding action is not enabled 1 _B the corresponding action is enabled This bit can only be written when the correspondent WAD _i bit is set. It can be set for debug purpose by CPU also, when the DPLL is disabled. The enable bit becomes active only when the DPLL is in operation (DEN = 1).
AEN9	9	rw	ACTION_9 enable see bit 8
AEN10	10	rw	ACTION_10 enable see bit 8
AEN11	11	rw	ACTION_11 enable see bit 8
AEN12	12	rw	ACTION_12 enable see bit 8
AEN13	13	rw	ACTION_13 enable see bit 8

Generic Timer Module (GTM)

Field	Bits	Type	Description
AEN14	14	rw	ACTION_14 enable see bit 8
AEN15	15	rw	ACTION_15 enable see bit 8
WAD8	16	w	Write control bit of Action_8 0 _B the corresponding AENi bit is not writable 1 _B the corresponding AENi bit is writable Note: For writing WADx =1 only the corresponding the AENx bits are written. The AENx bits remain unchanged when the corresponding WADx=0.
WAD9	17	w	Write control bit of Action_9 see bit 16
WAD10	18	w	Write control bit of Action_10 see bit 16
WAD11	19	w	Write control bit of Action_11 see bit 16
WAD12	20	w	Write control bit of Action_12 see bit 16
WAD13	21	w	Write control bit of Action_13 see bit 16
WAD14	22	w	Write control bit of Action_14 see bit 16
WAD15	23	w	Write control bit of Action_15 see bit 16
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.5 Register DPLL_CTRL_4

Action enable register.

GTM_DPLL_CTRL_4

DPLL Control Register 4

(28010_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								WAD 23	WAD 22	WAD 21	WAD 20	WAD 19	WAD 18	WAD 17	WAD 16
r								w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AEN 23	AEN 22	AEN 21	AEN 20	AEN 19	AEN 18	AEN 17	AEN 16	Reserved							
rw	rw	rw	rw	rw	rw	rw	rw	r							

Field	Bits	Type	Description
Reserved	[7:0]	r	Reserved Read as zero, should be written as zero.
AEN16	8	rw	ACTION_16 enable 0 _B the corresponding action is not enabled 1 _B the corresponding action is enabled This bit can only be written when the correspondent WAD _i bit is set. It can be set for debug purpose by CPU also, when the DPLL is disabled. The enable bit becomes active only when the DPLL is in operation (DEN = 1).
AEN17	9	rw	ACTION_17 enable see bit 8
AEN18	10	rw	ACTION_18 enable see bit 8
AEN19	11	rw	ACTION_19 enable see bit 8
AEN20	12	rw	ACTION_20 enable see bit 8
AEN21	13	rw	ACTION_21 enable see bit 8

Generic Timer Module (GTM)

Field	Bits	Type	Description
AEN22	14	rw	ACTION_22 enable see bit 8
AEN23	15	rw	ACTION_23 enable see bit 8
WAD16	16	w	Write control bit of Action_16 0= the corresponding AENi bit is not writable 1= the corresponding AENi bit is writable Note: For writing WADx =1 only the corresponding the AENx bits are written. The AENx bits remain unchanged when the corresponding WADx=0.
WAD17	17	w	Write control bit of Action_17 see bit 16
WAD18	18	w	Write control bit of Action_18 see bit 16
WAD19	19	w	Write control bit of Action_19 see bit 16
WAD20	20	w	Write control bit of Action_20 see bit 16
WAD21	21	w	Write control bit of Action_21 see bit 16
WAD22	22	w	Write control bit of Action_22 see bit 16
WAD23	23	w	Write control bit of Action_23 see bit 16
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.6 Register DPLL_ACT_STA

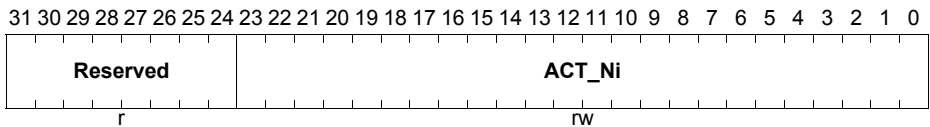
Action status register including shadow register.

GTM_DPLL_ACT_STA

DPLL ACTION Status Register With Shadow Register

(28018_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
ACT_Ni	[23:0]	rw	<p>New output data values concerning to action i provided</p> <p>0_B no new output data available after a recent PMT request or actual event value is in the past or invalid</p> <p>1_B new PMTR data received or calculation is to be precise by taking into account new TRIGGER or STATE values</p> <p>Note: ACT_Ni is</p> <ul style="list-style-type: none"> • set (for AENi=1 and a new valid PMTR), that means when new action data are to be calculated for the correspondent action. After each calculation of the new actions values the ACT_Ni bit updates the corresponding bit in the connected shadow register. The status of the ACT_Ni bits in the shadow register is reflected by the corresponding DPLL output signal ACT_V (valid bit). • reset together with the corresponding shadow register bit for AENi=0; • reset without the corresponding shadow register bit when the calculated event is in the past • the corresponding shadow register bit is reset, when new PMTR data are written or when the provided action data are read (blocking read) • writable for debugging purposes together with the corresponding shadow register when DEN=0 <p>Note: This bit can only be written when the DPLL is disabled.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.7 Register DPLL_OSW

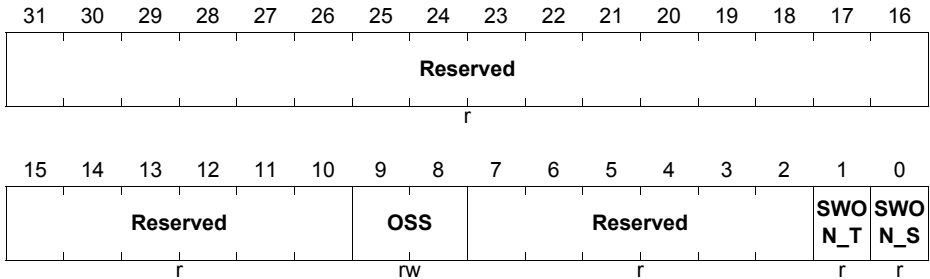
Offset and switch old/new address register.

GTM_DPLL_OSW

DPLL Offset And Switch Old/New Address Register

(2801C_H)

Reset Value: 00000200_H



Field	Bits	Type	Description
SWON_S	0	r	<p>Switch of new STATE</p> <p>Switch bit for LSB address of STATE.</p> <p>This bit is changed for each write access to TS_S/TS_S_OLD. Using this unchanged address bit SWON_S for any access to TS_S results always in an access to TS_S_OLD. For writing to this address the former old (TS_S_OLD_OLD) value is overwritten by the new one while the SWON_S bit changes. Thus the former new one is now the old one and the next access is after changing SWON_S directed to this place. Therefore write to TS_S first and after that immediately to FTV_S and PSSM, always before a new TS_S value is to be written.</p> <p>Note: After writing TS_S, FTV_S and PSSM in this order the address pointer AP with LSB(AP)=SWON_S shows for the corresponding address to TS_S_OLD, FTV_S and PSSM while LSB(AP)≠SWON_S results in an access to TS_S, FTV_S_OLD and PSSM_OLD respectively. The value can be read only. This bit is reset when disabling the DPLL (DEN = 0).</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
SWON_T	1	r	<p>Switch of new TRIGGER</p> <p>Switch bit for LSB address of TRIGGER. This bit is changed for each write access to TS_T/TS_T_OLD. Using this unchanged address bit SWON_T for any access to TS_T results always in an access to TS_T_OLD. For writing to this address the former old (TS_T_OLD_OLD) value is overwritten by the new one while the SWON_T bit changes. Thus the former new one is now the old one and the next access is after changing SWON_T directed to this place. Therefore write to TS_T first and after that immediately to FTV_T and PSTM, always before a new TS_T value is to be written.</p> <p>Note: After writing TS_T, FTV_T and PSTM in this order the address pointer AP with LSB(AP)=SWON_T shows for the corresponding address to TS_T_OLD, FTV_T and PSTM while LSB(AP)=/SWON_T results in an access to TS_T, FTV_T_OLD and PSTM_OLD respectively. The value can be read only. This bit is reset when disabling the DPLL (DEN = 0).</p>
Reserved	[7:2]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>
OSS	[9:8]	rw	<p>Offset size of RAM region 2</p> <p>00_B Offset size 128 of RAM region 2. 01_B Offset size 256 of RAM region 2. 10_B Offset size 512 of RAM region 2. 11_B Offset size 1024 of RAM region 2.</p> <p>Note: At least 128 and at most 1024 values can be stored in each of the RAM 2 regions a to d accordingly. The value can be set only for DEN=0. The change of the OSS value results in an automatic change of the offset values in the DPLL_AOSV_2 register.</p> <p>Note: This bits can only be written when the DPLL is disabled.</p>
Reserved	[31:10]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

26.16.11.8 Register DPLL_AOSV_2

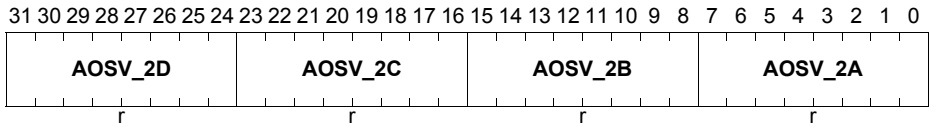
Address offset register of RAM 2 regions

GTM_DPLL_AOSV_2

DPLL Address Offset Register For APT In RAM Region 2

(28020_H)

Reset Value: 18100800_H



Field	Bits	Type	Description
AOSV_2A	[7:0]	r	Address offset value of the RAM 2A region The value in this field is to be multiplied by 256 (shift left 8 Bits) and added with the start address of the RAM in order to get the start address of RAM region 2A. When the APT value is added to this start address, the current RAM cell RDT_Tx is addressed.
AOSV_2B	[15:8]	r	Address offset value of the RAM 2B region The value in this field is to be multiplied by 256 (shift left 8 Bits) and added with the start address of the RAM in order to get the start address of RAM region 2B. When the APT value is added to this start address, the current RAM cell TSF_Tx is addressed.
AOSV_2C	[23:16]	r	Address offset value of the RAM 2C region The value in this field is to be multiplied by 256 (shift left 8 Bits) and added with the start address of the RAM in order to get the start address of RAM region 2C. When the APT value is added to this start address, the current RAM cell ADT_Tx is addressed.

Generic Timer Module (GTM)

Field	Bits	Type	Description
AOSV_2D	[31:24]	r	<p>Address offset value of the RAM 2D region</p> <p>The value in this field is to be multiplied by 256 (shift left 8 Bits) and added with the start address of the RAM in order to get the start address of RAM region 2D. When the APT value is added to this start address, the current RAM cell DT_Tx is addressed.</p> <p>Note: The offset values are needed to support a scalable RAM size of region 2 from 1,5 Kbytes to 12 Kbytes. The values above must be in correlation with the offset size defined in the OSW register. All offset values are set automatically in accordance to the OSS value in the DPLL_OSW register. This value can be set only for DEN=0.</p>

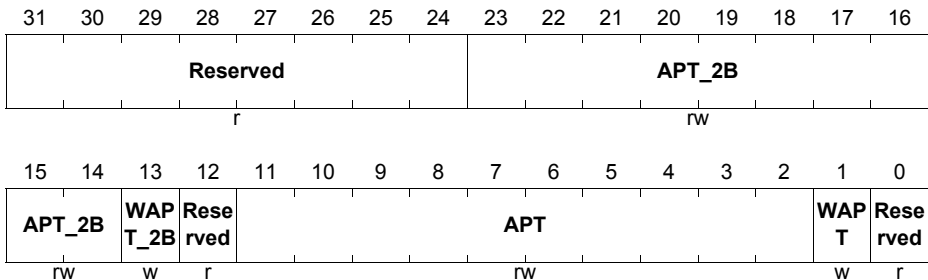
26.16.11.9 Register DPLL_APT

Actual RAM pointer address for TRIGGER.

GTM_DPLL_APT

DPLL Actual RAM Pointer to RAM Regions 2A, B and D (28024_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
Reserved	0	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
WAPT	1	w	<p>Write bit for address pointer APT read as zero</p> <p>0_B the APT is not writable 1_B the APT is writable</p> <p>Note: This bit is cleared automatically after write.</p>
APT	[11:2]	rw	<p>Address pointer TRIGGER Actual RAM pointer address value offset for DT_T[i] and RDT_T[i] in FULL_SCALE for 2*(TNU+1-SYN_NT) TRIGGER events</p> <p>This pointer is used for the RAM region 2 subsections 2A and 2D. The pointer APT is increment for each valid TRIGGER event (simultaneously with APT_2B, APT_2C) for DIR1=0. For DIR1=1 the APT is decremented.</p> <p>The APT offset value is added in the above shown bit position with the subsection address offset of the corresponding RAM region</p> <p>Note: The APT pointer value is directed to the RAM position, in which the data values are to be written, which corresponds to the last increment. The APT value is not to be changed, when the direction (shown by DIR1) changes, because it points always to a storage place after the considered increment. Changing of DIR1 takes place always after a valid TRIGGER event and the resulting increment/decrement.</p> <p>Note: This value can only be written when the WAPT bit is set.</p>
Reserved	12	r	<p>Reserved Read as zero, should be written as zero.</p>
WAPT_2B	13	w	<p>Write bit for address pointer APT_2B read as zero</p> <p>0_B the APT_2B is not writable 1_B the APT_2B is writable</p> <p>Note: This bit is cleared automatically after write.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
APT_2B	[23:14]	rw	<p>Address pointer TRIGGER for RAM region 2B Actual RAM pointer address value for TSF_T[i] Actual RAM pointer address of TRIGGER events in FULL_SCALE for $2*(TNU+1)$ TRIGGER periods; this pointer is used for the RAM region 2B. The RAM pointer is initially set to zero. For SYT = 1: The pointer APT_2B is increment by SYN_T_OLD for each valid TRIGGER event (simultaneously with APT and APT_2C) for DIR1=0 when a valid TRIGGER input appears. For DIR1=1 (backwards) the APT is decremented by SYN_T_OLD. For SYT = 0: APT_2B is incremented or decremented by 1. In addition when the APT_2C value is written by the CPU - in order to synchronize the DPLL- with the next valid TRIGGER event the APT_2B_EXT value is added/subtracted (while APT_2B_STATUS is one; see DPLL_APT_SYNC register at Chapter 26.16.11.23). Note: This value can only be written when the WAPT_2B bit is set.</p>
Reserved	[31:24]	r	<p>Reserved Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

26.16.11.10 Register DPLL_APS

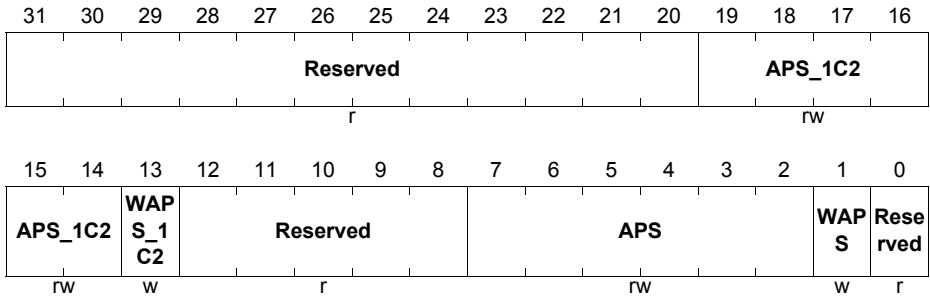
Actual RAM pointer address for STATE.

GTM_DPLL_APS

DPLL Actual RAM Pointer to RAM Regions 1C1, 1C2 and 1C4

(28028_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
Reserved	0	r	Reserved Read as zero, should be written as zero.
WAPS	1	w	Write bit for address pointer APS read as zero 0 _B the APS is not writable 1 _B the APS is writable Note: This bit is cleared automatically after write.

Generic Timer Module (GTM)

Field	Bits	Type	Description
APS	[7:2]	rw	<p>Address pointer STATE Actual RAM pointer address value for DT_S[i] and RDT_S[i] Actual RAM pointer and synchronization position/value of STATE events in FULL_SCALE for up to 64 STATE events but limited to $2^*(SNU+1-SYN_NS)$ in normal and emergency mode for SYSF=0 or to $2^*(SNU+1)-SYN_NS$ for SYSF=1 respectively; this pointer is used for the RAM region 1C1 and 1C4. APS is incremented (decremented) by one for each valid STATE event and DIR2=0 DIR2=1). The APS offset value is added in the above shown bit position with the subsection offset of the RAM region. Note: The APS pointer value is directed to the RAM position, in which the data values are to be written, which correspond to the last increment. The APS value is not to be changed, when the direction (shown by DIR2) changes, because it points always to a storage place after the considered increment. Changing of DIR2 takes place always after a valid STATE event and the resulting increment/decrement. Note: This value can only be written when the WAPS bit is set.</p>
Reserved	[12:8]	r	<p>Reserved Read as zero, should be written as zero.</p>
WAPS_1C2	13	w	<p>Write bit for address pointer APS_1C2 read as zero 0_B the APS_1C2 is not writable 1_B the APS_1C2 is writable Note: This bit is cleared automatically after write.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
APS_1C2	[19:14]	rw	<p>Address pointer STATE for RAM region 1C2 Actual RAM pointer address value for TSF_S[i] Initial value: zero (0x00). Actual RAM pointer and synchronization position/value of STATE events in FULL_SCALE for up to 64 STATE events but limited to $2*(SNU+1)$ in normal and emergency mode; this pointer is used for the RAM region 1C2. For SYS=1: APS_1C2 is increment (decremented) by SYN_S_OLD for each valid STATE event and DIR2=0 (DIR2=1). The APS_1C2 offset value is added in the above shown bit position with the subsection offset of the RAM region. For SYS=0: APT_1C2 is incremented or decremented by 1 respectively. In addition when the APS_1C3 value is written by the CPU - in order to synchronize the DPLL- with the next valid STATE event the APS_1C2_EXT value is added/subtracted (while APS_1C2_STATUS is one; see DPLL_APT_SYNC register at Section 26.16.11.24). Note: This value can only be written when the WAPS_1C2 bit is set.</p>
Reserved	[31:20]	r	<p>Reserved Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

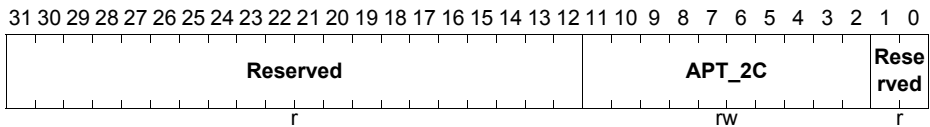
26.16.11.11 Register DPLL_APT_2C

Actual RAM pointer address for region 2C.

GTM_DPLL_APT_2C

DPLL Actual RAM Pointer to RAM Region 2C
(2802C_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
Reserved	[1:0]	r	Reserved Read as zero, should be written as zero.
APT_2C	[11:2]	rw	Address pointer TRIGGER for RAM region 2C Actual RAM pointer address value for ADT_T[i] Actual RAM pointer address value of TRIGGER adapt events in FULL_SCALE for 2*(TNU+1-SYN_NT) TRIGGER periods depending on the size of the used RAM 2; this pointer is used for the RAM region 2 for the subsection 2C only. The RAM pointer is initially set to zero. The APT_2C value is set by the CPU when the synchronization condition was detected. Within the RAM region 2C initially the conditions for synchronization gaps and adapt values are stored by the CPU.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:12]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p> <p>Note: The APT_2C pointer values are directed to the RAM position of the profile element in RAM region 2C, which correspond to the current increment. For DIR1=0 (DIR1=1) the pointers APT_2C_x are increment (decremented) by one simultaneously with APT. For SMC=0 the change of DIR1 takes place always after a valid TRIGGER event (by evaluation of the invalid slope) and the resulting increment/decrement. In the case SMC=1 the direction change is known before the input event is processed.</p> <p>The correction of the APT_2C pointer differs: for SMC=0 correct 4 times and for SMC=1 correct only 2 times.</p> <p>The APT_2C_x offset value is added in the above shown bit position with the subsection address offset of the corresponding RAM region.</p>

Generic Timer Module (GTM)

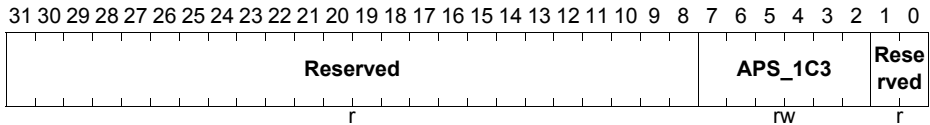
26.16.11.12 Register DPLL_APS_1C3

Actual RAM pointer address for RAM region 1C3.

GTM_DPLL_APS_1C3

DPLL Actual RAM Pointer to RAM Region 1C3
(28030_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
Reserved	[1:0]	r	Reserved Read as zero, should be written as zero.
APS_1C3	[7:2]	rw	Address pointer STATE for RAM region 1C3 Initial value: zero (0x00). Actual RAM pointer and synchronization position/value of STATE events in FULL_SCALE for up to 64 STATE events but limited to 2*(SNU+1)-SYN_NS in normal and emergency mode for SYSF=0 or to 2*(SNU+1)-SYN_NS for SYSF=1 respectively; this pointer is used for the RAM region 1C3. The RAM pointer is set by the CPU accordingly, when the synchronization condition was detected.
Reserved	[31:8]	r	Reserved Read as zero, should be written as zero. Note: The APS_1C3 pointer value is directed to the RAM position of the profile element in RAM region 1C2, which corresponds to the current increment. When changing the direction DIR1 or DIR2 respectively, this is always known before a valid STATE event is processed. This is because of the pattern recognition in SPE (for PMSM) or because of the direction change recognition by TRIGGER. This direction change results in an automatic increment (forwards) or decrement (backwards) when the input event occurs in addition with a 2 times correction. The APS_1C3_x offset value is added in the above shown bit position with the subsection address offset of the corresponding RAM region.

Generic Timer Module (GTM)

26.16.11.13 Register DPLL_NUTC

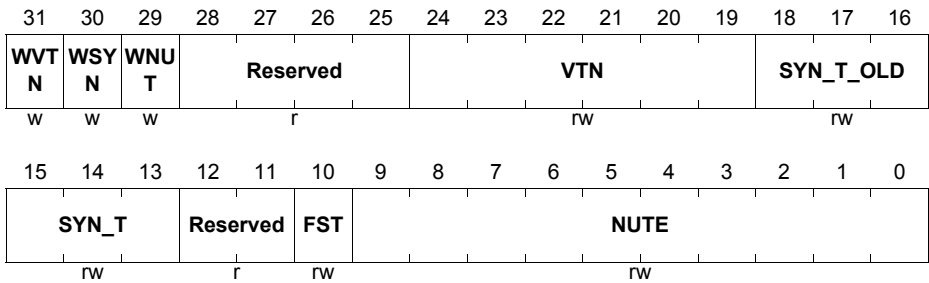
number of recent TRIGGER events used for calculations.

GTM_DPLL_NUTC

DPLL Number of Recent TRIGGER Events Used for Calculations

(28034_H)

Reset Value: 00012001_H



Field	Bits	Type	Description
NUTE	[9:0]	rw	<p>Number of recent TRIGGER events used for SUB_INC1 and action calculations modulo 2*(TNUMax+1)</p> <p>NUTE: number of last nominal increments to be considered for the calculations.</p> <p>No gap is considered in that case for this value, but in the VTN value (see below):</p> <p>This register is set by the CPU, but reset automatically to “1” by a change of direction or lost of LOCK. Each other value can be set by the CPU, maybe FULL_SCALE, HALF_SCALE or parts of them. For FULL_SCALE set NUTE=2*(TNU+1) and fir HALF_SCALE NUTE=TNU+1. The relation values QDT_Tx are calculated using NUTE values in the past with its maximum value of 2*(TNU +1). The value zero (in combination with the value FST=1) does mean 211 values in the past.</p> <p>Note: This value can only be written when the WNUT bit is set.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
FST	10	rw	<p>this value is to be set, when NUTE is set to FULL_SCALE</p> <p>0= the NUTE value is less then FULL_SCALE 1= the NUTE value is equal to FULL_SCALE Note: This value can only be written when the WNUT bit is set.</p>
Reserved	[12:11]	r	<p>Reserved Read as zero, should be written as zero.</p>
SYN_T	[15:13]	rw	<p>Number of real and virtual events to be considered for the current increment</p> <p>This value reflects the NT value of the last valid increment, stored in ADT_T[i]; to be updated after all calculations in step 17 of Table 26-48. Note: This value can only be written when the WSYN bit is set.</p>
SYN_T_OL LD	[18:16]	rw	<p>Number of real and virtual events to be considered for the last increment</p> <p>This value reflects the NT value of the last but one valid increment, stored in ADT_T[i]; is updated automatically when writing SYN_T. Note: This value is updated by the SYN_T value when the WSYN bit is set.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
VTN	[24:19]	rw	<p>Virtual TRIGGER number</p> <p>Number of virtual increments in the current NUTE region This value reflects the number of virtual increments in the current NUTE region; for NUTE=1 this value is zero, when the CPU sets NUTE to a value > 1, it must also set VTN to the correspondent value; for NUTE is set to FULL_SCALE including NUTE=zero (211 modulo 211) the VTN is to be set to 2* SYN_NT. the VTN value is subtracted from the NUTE value in order to get the corresponding APT value for the past; the VTN value is not used for the APT_2B pointer. VTN is to be updated by the CPU when a new gap is to be considered for NUTE or a gap is leaving the NUTE region; for this purpose the TINT values in the profile can be used to generate an interrupt for the CPU at the corresponding positions; no further update of VTN is necessary when NUTE is set to FULL_SCALE Note: This value can only be written when the WVTN bit is set.</p>
Reserved	[28:25]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>
WNUT	29	w	<p>Write control bit for NUTE and FST</p> <p>read as zero</p> <p>0_B the NUTE value is not writable 1_B the NUTE value is writable Note: This bit is cleared automatically after write.</p>
WSYN	30	w	<p>Write control bit for SYN_T and SYN_T_OLD</p> <p>read as zero</p> <p>0_B the SYN_T value is not writable 1_B the SYN_T value is writable Note: This bit is cleared automatically after write.</p>
WVTN	31	w	<p>Write control bit for VTN</p> <p>read as zero</p> <p>0_B the VTN value is not writable 1_B the VTN value is writable Note: This bit is cleared automatically after write.</p>

26.16.11.14 Register DPLL_NUSC

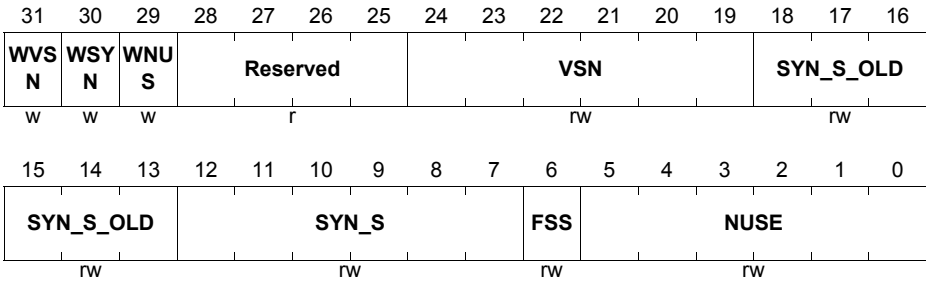
Number of recent STATE events used for calculations.

GTM_DPLL_NUSC

DPLL Number of Recent STATE Events Used for Calculations

(28038_H)

Reset Value: 00002081_H



Field	Bits	Type	Description
NUSE	[5:0]	rw	<p>Number of recent STATE events used for SUB_INCx calculations modulo 2*(SNUmax+1)</p> <p>No gap is considered in that case for this value, but in the VSN value (see below):</p> <p>This register is set by the CPU but reset automatically to “1” by a change of direction or lost of LOCK. Each other value can be set by the CPU, maybe FULL_SCALE, HALF_SCALE or parts of them. The relation values QDT_Sx are calculated using NUSE values in the past with its maximum value of 2*SNU+1.</p> <p>Note: This value can only be written when the WNUS bit is set.</p>
FSS	6	rw	<p>this value is to be set, when NUSE is set to FULL_SCALE</p> <p>0= the NUSE value is less then FULL_SCALE 1= the NUSE value is equal to FULL_SCALE</p> <p>Note: This value can only be written when the WNUT bit is set.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
SYN_S	[12:7]	rw	<p>Number of real and virtual events to be considered for the current increment</p> <p>This value reflects the NS value of the last valid increment, stored in ADT_S[i]; to be updated after all calculations in step 37 of Table 26-48.</p> <p>Note: This value can only be written when the WNUS bit is set.</p>
SYN_S_0 LD	[18:13]	rw	<p>Number of real and virtual events to be considered for the last increment</p> <p>This value reflects the NS value of the last but one valid increment, stored in ADT_S[i]; is updated automatically when writing SYN_S.</p> <p>Note: This value is updated by the SYN_S value when the WSYN bit is set.</p>
VSN	[24:19]	rw	<p>Virtual STATE number</p> <p>Number of virtual state increments in the current NUSE region</p> <p>This value reflects the number of virtual increments in the current NUSE region; for NUSE=1 this value is zero, when the CPU sets NUSE to a value > 1 or zero(27 modulo 27), it must also set VSN to the correspondent value;</p> <p>the VSN value is subtracted from the NUSE value in order to get the corresponding APS value for the past; the VSN value is not used for the APS_1C2 pointer.</p> <p>VSN is to be updated by the CPU when a new gap is to be considered for NUSE or a gap is leaving the NUSE region; for this purpose the SASI interrupt can be used; no further update of VSN is necessary when NUSE is set to FULL_SCALE.</p> <p>Note: This value can only be written when the WVSN bit is set.</p>
Reserved	[28:25]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>
WNUS	29	w	<p>Write control bit for NUSE; read as zero</p> <p>0_B the NUSE value is not writable</p> <p>1_B the NUSE value is writable</p> <p>Note: This bit is cleared automatically after write.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
WSYN	30	w	<p>Write control bit for SYN_S and SYN_S_OLD; read as zero</p> <p>0_B the SYN_S value is not writable 1_B the SYN_S value is writable Note: This bit is cleared automatically after write.</p>
WVSN	31	w	<p>Write control bit for VSN; read as zero</p> <p>0_B the VSN value is not writable 1_B the VSN value is writable Note: This bit is cleared automatically after write.</p>

Generic Timer Module (GTM)

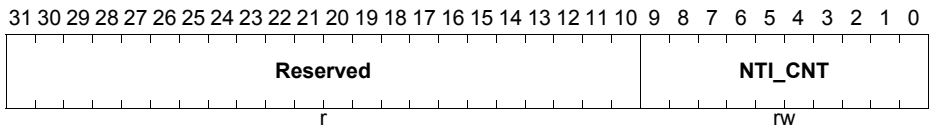
26.16.11.15 Register DPLL_NTICNT

Number of active TRIGGER events to interrupt.

GTM_DPLL_NTICNT

DPLL Number of Active TRIGGER Events to Interrupt
(2803C_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
NTI_CNT	[9:0]	rw	Number of TRIGGERs to interrupt Number of active TRIGGER events to the next DPLL_CDTI interrupt. This value shows the remaining TRIGGER events until an active TRIGGER slope results in a DPLL_CDTI interrupt; the value is to be count down for each valid TRIGGER event.
Reserved	[31:10]	r	Reserved Read as zero, should be written as zero.

26.16.11.16 Register DPLL_IRQ_NOTIFY

Interrupt register.

GTM_DPLL_IRQ_NOTIFY
DPLL Interrupt Notification Register (28040_H)
Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			DCG I	SORI	TORI	CDSI	CDTI	TE4I	TE3I	TE2I	TE1I	TE0I	LL2I	GL2I	
r			rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EI	LL1I	GL1I	W1I	W2I	PWI	TASI	SASI	MTI	MSI	TISI	SISI	TAXI	TINI	PEI	PDI
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
PDI	0	rwh	DPLL disable interrupt announces the switch off of the DEN bit 0 _B The DPLL disable interrupt is not requested 1 _B The DPLL disable interrupt is requested
PEI	1	rwh	DPLL enable interrupt announces the switch on of the DEN bit 0 _B The DPLL enable interrupt is not requested 1 _B The DPLL enable interrupt is requested
TINI	2	rwh	TRIGGER minimum hold time violation interrupt (ΔT) 0 _B No violation of minimum hold time of TRIGGER is detected 1 _B A violation of minimum hold time of TRIGGER is detected
TAXI	3	rwh	TRIGGER maximum hold time violation interrupt ($\Delta T > THMA > 0$) 0 _B No violation of maximum hold time of TRIGGER is detected 1 _B A violation of maximum hold time of TRIGGER is detected
SISI	4	rwh	STATE inactive slope interrupt 0 _B No inactive slope of STATE is detected 1 _B An inactive slope of STATE is detected

Generic Timer Module (GTM)

Field	Bits	Type	Description
TISI	5	rwh	TRIGGER inactive slope interrupt 0_B No inactive slope of TRIGGER is detected 1_B An inactive slope of TRIGGER is detected
MSI	6	rwh	Missing STATE interrupt 0_B The missing STATE interrupt is not requested 1_B The missing STATE interrupt is requested
MTI	7	rwh	Missing TRIGGER interrupt 0_B The missing TRIGGER interrupt is not requested 1_B The missing TRIGGER interrupt is requested
SASI	8	rwh	STATE active slope interrupt 0_B No active slope of STATE is detected 1_B An active slope of STATE is detected
TASI	9	rwh	TRIGGER active slope interrupt 0_B No active slope of TRIGGER is detected 1_B An active slope of TRIGGER is detected
PWI	10	rwh	Plausibility window (PVT) violation interrupt of TRIGGER 0_B The plausibility window is not violated 1_B The plausibility window is violated
W2I	11	rwh	RAM write access to RAM region 2 interrupt 0_B The RAM write access interrupt is not requested 1_B The RAM write access interrupt is requested
W1I	12	rwh	Write access to RAM region 1B or 1C interrupt 0_B The RAM write access interrupt is not requested 1_B The RAM write access interrupt is requested
GL1I	13	rwh	Get of lock interrupt, for SUB_INC1 0_B The lock getting interrupt is not requested 1_B The lock getting interrupt is requested
LL1I	14	rwh	Loss of lock interrupt for SUB_INC1 0_B The lock loss interrupt is not requested 1_B The lock loss interrupt is requested
EI	15	rwh	Error interrupt (see status register bit 31) 0_B The error interrupt is not requested 1_B The error interrupt is requested
GL2I	16	rwh	Get of lock interrupt, for SUB_INC2 0_B The lock getting interrupt is not requested 1_B The lock getting interrupt is requested

Generic Timer Module (GTM)

Field	Bits	Type	Description
LL2I	17	rwh	Loss of lock interrupt for SUB_INC2 0_B The lock loss interrupt is not requested 1_B The lock loss interrupt is requested
TE0I	18	rwh	TRIGGER event interrupt 0 0_B No Interrupt on TRIGGER event 0 requested 1_B Interrupt on TRIGGER event 0 requested
TE1I	19	rwh	TRIGGER event interrupt 1 0_B No Interrupt on TRIGGER event 1 requested 1_B Interrupt on TRIGGER event 1 requested
TE2I	20	rwh	TRIGGER event interrupt 2 0_B No Interrupt on TRIGGER event 2 requested 1_B Interrupt on TRIGGER event 2 requested
TE3I	21	rwh	TRIGGER event interrupt 3 0_B No Interrupt on TRIGGER event 3 requested 1_B Interrupt on TRIGGER event 3 requested
TE4I	22	rwh	TRIGGER event interrupt 4 0_B No Interrupt on TRIGGER event 4 requested 1_B Interrupt on TRIGGER event 4 requested
CDTI	23	rwh	Calculation of TRIGGER duration done, only while NTI_CNT is zero 0_B No Interrupt on calculated TRIGGER duration requested or NTI_CNT is not zero 1_B Interrupt on calculated TRIGGER duration requested while NTI_CNT is zero
CDSI	24	rwh	Calculation of STATE duration done 0_B No Interrupt on calculated STATE duration requested 1_B Interrupt on calculated STATE duration requested
TORI	25	rwh	TRIGGER out of range interrupt 0_B TRIGGER is not out of range 1_B TRIGGER is out of range, the TOR bit in the DPLL_STATUS register is set to 1
SORI	26	rwh	STATE out of range 0_B STATE is not out of range 1_B STATE is out of range, the SOR bit in the DPLL_STATUS register is set to 1

Generic Timer Module (GTM)

Field	Bits	Type	Description
DCGI	27	rwh	Direction change interrupt 0 _B No direction change of TRIGGER is detected 1 _B Direction change of TRIGGER is detected Note: The interrupt occurs at line number 0.
Reserved	[31:28]	r	Reserved Read as zero, should be written as zero.

Note: All bits in the DPLL_IRQ_NOTIFY register are set permanently until writing a one bit value is performed to the corresponding bit.

26.16.11.17 Register DPLL_IRQ_EN

Interrupt enable register.

GTM_DPLL_IRQ_EN

DPLL Interrupt Enable Register (28044_H) Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				DCGI	SORI	TORI	CDSI _IRQ _EN	CDTI _IRQ _EN	TE4I _IRQ _EN	TE3I _IRQ _EN	TE2I _IRQ _EN	TE1I _IRQ _EN	TE0I _IRQ _EN	LL2I _IRQ _EN	GL2I _IRQ _EN
r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EI RQ EN	LL1I _IRQ _EN	GL1I _IRQ _EN	W1I _IRQ _EN	W2I _IRQ _EN	PWI _IRQ _EN	TASI _IRQ _EN	SASI _IRQ _EN	MTI _IRQ _EN	MSI _IRQ _EN	TISI _IRQ _EN	SISI _IRQ _EN	TAXI _IRQ _EN	TINI _IRQ _EN	PEI RQ EN	PDI _IRQ _EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PDI_IRQ_EN	0	rw	DPLL disable interrupt enable, when switch off of the DEN bit 0 _B The DPLL disable interrupt is not enabled 1 _B The DPLL disable interrupt is enabled
PEI_IRQ_EN	1	rw	DPLL enable interrupt enable, when switch on of the DEN bit 0 _B The DPLL enable interrupt is not enabled 1 _B The DPLL enable interrupt is enabled

Generic Timer Module (GTM)

Field	Bits	Type	Description
TINI_IRQ_EN	2	rw	TRIGGER minimum hold time violation interrupt enable bit 0 _B Minimum hold time violation of TRIGGER interrupt is not enabled 1 _B The minimum hold time violation of TRIGGER interrupt is enabled
TAXI_IRQ_EN	3	rw	TRIGGER maximum hold time violation interrupt enable bit 0 _B Maximum hold time violation of TRIGGER interrupt is not enabled 1 _B The maximum hold time violation of TRIGGER interrupt is enabled
SISI_IRQ_EN	4	rw	STATE inactive slope interrupt enable bit 0 _B The interrupt at the inactive slope of STATE is not enabled 1 _B The interrupt at the inactive slope of STATE is enabled
TISI_IRQ_EN	5	rw	TRIGGER inactive slope interrupt enable bit 0 _B The interrupt at the inactive slope of TRIGGER is not enabled 1 _B The interrupt at the inactive slope of TRIGGER is enabled
MSI_IRQ_EN	6	rw	Missing STATE interrupt enable 0 _B The missing STATE interrupt is not enabled 1 _B The missing STATE interrupt is enabled
MTI_IRQ_EN	7	rw	Missing TRIGGER interrupt enable 0 _B The missing TRIGGER interrupt is not enabled 1 _B The missing TRIGGER interrupt is enabled
SASI_IRQ_EN	8	rw	STATE active slope interrupt enable 0 _B The active slope STATE interrupt is not enabled. 1 _B The active slope STATE interrupt is enabled
TASI_IRQ_EN	9	rw	TRIGGER active slope interrupt enable 0 _B The active slope TRIGGER interrupt is not enabled 1 _B The active slope TRIGGER interrupt is enabled

Generic Timer Module (GTM)

Field	Bits	Type	Description
PWI_IRQ_EN	10	rw	Plausibility window (PVT) violation interrupt of TRIGGER enable 0 _B The plausibility violation interrupt is not enabled 1 _B The plausibility violation interrupt is enabled
W2I_IRQ_EN	11	rw	RAM write access to RAM region 2 interrupt enable 0 _B The RAM write access interrupt is not enabled 1 _B The RAM write access interrupt is enabled
W1I_IRQ_EN	12	rw	Write access to RAM region 1B or 1C interrupt 0 _B The RAM write access interrupt is not enabled 1 _B The RAM write access interrupt is enabled.
GL1I_IRQ_EN	13	rw	Get of lock interrupt enable, when lock arises 0 _B The lock getting interrupt is not enabled 1 _B The lock getting interrupt is enabled
LL1I_IRQ_EN	14	rw	Loss of lock interrupt enable 0 _B The lock loss interrupt is not enabled 1 _B The lock loss interrupt is enabled
EI_IRQ_EN	15	rw	Error interrupt enable (see status register) 0 _B The error interrupt is not enabled 1 _B The error interrupt is enabled
GL2I_IRQ_EN	16	rw	Get of lock interrupt enable for SUB_INC2 0 _B The lock getting interrupt is not requested 1 _B The lock getting interrupt is requested
LL2I_IRQ_EN	17	rw	Loss of lock interrupt enable for SUB_INC2 0 _B The lock loss interrupt is not requested 1 _B The lock loss interrupt is requested
TE0I_IRQ_EN	18	rw	TRIGGER event interrupt 0 enable 0 _B No Interrupt on TRIGGER event 0 enabled 1 _B Interrupt on TRIGGER event 0 enabled
TE1I_IRQ_EN	19	rw	TRIGGER event interrupt 1 enable 0 _B No Interrupt on TRIGGER event 1 enabled 1 _B Interrupt on TRIGGER event 1 enabled
TE2I_IRQ_EN	20	rw	TRIGGER event interrupt 2 enable 0 _B No Interrupt on TRIGGER event 2 enabled 1 _B Interrupt on TRIGGER event 2 enabled
TE3I_IRQ_EN	21	rw	TRIGGER event interrupt 3 enable 0 _B No Interrupt on TRIGGER event 3 enabled 1 _B Interrupt on TRIGGER event 3 enabled

Generic Timer Module (GTM)

Field	Bits	Type	Description
TE4I_IRQ_EN	22	rw	TRIGGER event interrupt 4 enable 0 _B No Interrupt on TRIGGER event 4 enabled 1 _B Interrupt on TRIGGER event 4 enabled
CDTI_IRQ_EN	23	rw	Enable interrupt when calculation of TRIGGER duration done 0 _B No Interrupt on calculated TRIGGER duration enabled 1 _B Interrupt on calculated TRIGGER duration enabled
CDSI_IRQ_EN	24	rw	Enable interrupt when calculation of TRIGGER duration done 0 _B No Interrupt on calculated STATE duration enabled 1 _B Interrupt on calculated STATE duration enabled
TORI	25	rw	TRIGGER out of range interrupt 0 _B No Interrupt when TRIGGER is out of range enabled 1 _B Interrupt when TRIGGER is out of range enabled
SORI	26	rw	STATE out of range 0 _B No Interrupt when STATE is out of range enabled 1 _B Interrupt when STATE is out of range enabled
DCGI	27	rw	Direction change interrupt 0 _B No Interrupt when a direction change of TRIGGER is detected 1 _B Interrupt when a direction change of TRIGGER is detected
Reserved	[31:28]	r	Reserved Read as zero, should be written as zero.

26.16.11.18 Register DPLL_IRQ_FORCINT

Force interrupt register.

GTM_DPLL_IRQ_FORCINT
DPLL Interrupt Force Register
(28048_H)
Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				TRG_DC GI	TRG_SO RI	TRG_TO RI	TRG_CD SI	TRG_CD TI	TRG_TE4 I	TRG_TE3 I	TRG_TE2 I	TRG_TE1 I	TRG_TE0 I	TRG_LL2 I	TRG_GL ZI
r				w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRG_EI	TRG_LL1 I	TRG_GL I1	TRG_W11	TRG_W21	TRG_PW1	TRG_TA SI	TRG_SA SI	TRG_MTI	TRG_MSI	TRG_TISI	TRG_SISI	TRG_TA XI	TRG_TINI	TRG_PEI	TRG_PDI
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
TRG_PDI	0	w	Force Interrupt PDI 0= the corresponding interrupt is not forced 1= the corresponding interrupt is forced for one clock Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
TRG_PEI	1	w	Force Interrupt PEI see bit 0
TRG_TINI	2	w	Force Interrupt TINI see bit 0
TRG_TAXI	3	w	Force Interrupt TAXI see bit 0
TRG_SISI	4	w	Force Interrupt SISI see bit 0
TRG_TISI	5	w	Force Interrupt TISI see bit 0
TRG_MSI	6	w	Force Interrupt MSI see bit 0
TRG_MTI	7	w	Force Interrupt MTI see bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG_SASI	8	w	Force Interrupt SASI see bit 0
TRG_TASI	9	w	Force Interrupt TASI see bit 0
TRG_PWI	10	w	Force Interrupt PWI see bit 0
TRG_W2I	11	w	Force Interrupt W2IF see bit 0
TRG_W1I	12	w	Force Interrupt W1I see bit 0
TRG_GL1I	13	w	Force Interrupt GL1I see bit 0
TRG_LL1I	14	w	Force Interrupt LL1I see bit 0
TRG_EI	15	w	Force Interrupt EI see bit 0
TRG_GL2I	16	w	Force Interrupt GL2I see bit 0
TRG_LL2I	17	w	Force Interrupt LL2I see bit 0
TRG_TE0I	18	w	Force Interrupt TE0I see bit 0
TRG_TE1I	19	w	Force Interrupt TE1I see bit 0
TRG_TE2I	20	w	Force Interrupt TE2I see bit 0
TRG_TE3I	21	w	Force Interrupt TE3I see bit 0
TRG_TE4I	22	w	Force Interrupt TE4I see bit 0
TRG_CDTI	23	w	Force Interrupt CDTI see bit 0
TRG_CDSI	24	w	Force Interrupt CDSI see bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG_TORI	25	w	Force Interrupt TORI see bit 0
TRG_SOR I	26	w	Force Interrupt SORI see bit 0
TRG_DCG I	27	w	Force Interrupt DCGI see bit 0
Reserved	[31:28]	r	Reserved Read as zero, should be written as zero.

26.16.11.19 Register DPLL_IRQ_MODE

Interrupt request mode register.

GTM_DPLL_IRQ_MODE
DPLL Interrupt Mode Register (2804C_H) **Reset Value: 00000000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																	r										IRQ MOD E				
																	r										rw				

Field	Bits	Type	Description
IRQ_MODE	[1:0]	rw	IRQ mode selection 00 _B Level mode 01 _B Pulse mode 10 _B Pulse-Notify mode 11 _B Single-Pulse mode Note: The interrupt modes are described in Section 26.2.5 .
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero.

26.16.11.20 Register DPLL_EIRQ_EN

Error interrupt enable register.

GTM_DPLL_EIRQ_EN

DPLL Error Interrupt Enable Register(28050_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			DCG I	SOR I	TOR I	CDSI _EIR _Q_E _N	CDTI _EIR _Q_E _N	TE4I _EIR _Q_E _N	TE3I _EIR _Q_E _N	TE2I _EIR _Q_E _N	TE1I _EIR _Q_E _N	TE0I _EIR _Q_E _N	LL2I _EIR _Q_E _N	GL2I _EIR _Q_E _N	
r			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EI_EI RQ_ _EN	LL1I _EIR _Q_E _N	GL1I _EIR _Q_E _N	W1I EIRQ _EN	W2I EIRQ _EN	PWI EIRQ _EN	TASI _EIR _Q_E _N	SASI _EIR _Q_E _N	MTI EIRQ _EN	MSI EIRQ _EN	TISI EIRQ _EN	SISI EIRQ _EN	TAXI _EIR _Q_E _N	TINI EIRQ _EN	PEI EIRQ _EN	PDI EIRQ _EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
PDI_EIRQ_EN	0	rw	DPLL disable interrupt enable, when switch off of the DEN bit 0 _B The DPLL disable interrupt is not enabled 1 _B The DPLL disable interrupt is enabled
PEI_EIRQ_EN	1	rw	DPLL enable interrupt enable, when switch on of the DEN bit 0 _B The DPLL enable interrupt is not enabled 1 _B The DPLL enable interrupt is enabled
TINI_EIRQ_EN	2	rw	TRIGGER minimum hold time violation interrupt enable bit. 0 _B Minimum hold time violation of TRIGGER interrupt is not enabled 1 _B Minimum hold time violation of TRIGGER interrupt is enabled
TAXI_EIRQ_EN	3	rw	TRIGGER maximum hold time violation interrupt enable bit 0 _B Maximum hold time violation of TRIGGER interrupt is not enabled 1 _B Maximum hold time violation of TRIGGER interrupt is enabled

Generic Timer Module (GTM)

Field	Bits	Type	Description
SISI_EIRQ_EN	4	rw	STATE inactive slope interrupt enable bit 0 _B The interrupt at the inactive slope of STATE is not enabled 1 _B The interrupt at the inactive slope of STATE is enabled
TISI_EIRQ_EN	5	rw	TRIGGER inactive slope interrupt enable bit 0 _B The interrupt at the inactive slope of TRIGGER is not enabled 1 _B The interrupt at the inactive slope of TRIGGER is enabled
MSI_EIRQ_EN	6	rw	Missing STATE interrupt enable 0 _B The missing STATE interrupt is not enabled 1 _B The missing STATE interrupt is enabled
MTI_EIRQ_EN	7	rw	Missing TRIGGER interrupt enable 0 _B The missing TRIGGER interrupt is not enabled 1 _B The missing TRIGGER interrupt is enabled
SASI_EIRQ_EN	8	rw	STATE active slope interrupt enable 0 _B The active slope STATE interrupt is not enabled 1 _B The active slope STATE interrupt is enabled
TASI_EIRQ_EN	9	rw	TRIGGER active slope interrupt enable 0 _B The active slope TRIGGER interrupt is not enabled 1 _B The active slope TRIGGER interrupt is enabled
PWI_EIRQ_EN	10	rw	Plausibility window (PVT) violation interrupt of TRIGGER enable 0 _B The plausibility violation interrupt is not enabled 1 _B The plausibility violation interrupt is enabled
W2I_EIRQ_EN	11	rw	RAM write access to RAM region 2 interrupt enable 0 _B The RAM write access interrupt is not enabled 1 _B The RAM write access interrupt is enabled
W1I_EIRQ_EN	12	rw	Write access to RAM region 1B or 1C interrupt 0 _B The RAM write access interrupt is not enabled 1 _B The RAM write access interrupt is enabled
GL1I_EIRQ_EN	13	rw	Get of lock interrupt enable, when lock arises 0 _B The lock getting interrupt is not enabled 1 _B The lock getting interrupt is enabled

Generic Timer Module (GTM)

Field	Bits	Type	Description
LL1I_EIRQ_EN	14	rw	Loss of lock interrupt enable 0 _B The lock loss interrupt is not enabled 1 _B The lock loss interrupt is enabled
EI_EIRQ_EN	15	rw	Error interrupt enable (see status register) 0 _B The lock error interrupt is not enabled 1 _B The lock error interrupt is enabled
GL2I_EIRQ_EN	16	rw	Get of lock interrupt enable for SUB_INC2 0 _B The lock getting interrupt is not requested 1 _B The lock getting interrupt is requested
LL2I_EIRQ_EN	17	rw	Loss of lock interrupt enable for SUB_INC2 0 _B The lock loss interrupt is not requested 1 _B The lock loss interrupt is requested
TE0I_EIRQ_EN	18	rw	TRIGGER event interrupt 0 enable 0 _B No Interrupt on calculated TRIGGER event 0 enabled 1 _B Interrupt on calculated TRIGGER event 0 enabled
TE1I_EIRQ_EN	19	rw	TRIGGER event interrupt 1 enable 0 _B No Interrupt on calculated TRIGGER event 1 enabled 1 _B Interrupt on calculated TRIGGER event 1 enabled
TE2I_EIRQ_EN	20	rw	TRIGGER event interrupt 2 enable 0 _B No Interrupt on calculated TRIGGER event 2 enabled 1 _B Interrupt on calculated TRIGGER event 2 enabled
TE3I_EIRQ_EN	21	rw	TRIGGER event interrupt 3 enable 0 _B No Interrupt on calculated TRIGGER event 3 enabled 1 _B Interrupt on calculated TRIGGER event 3 enabled
TE4I_EIRQ_EN	22	rw	TRIGGER event interrupt 4 enable 0 _B No Interrupt on calculated TRIGGER event 4 enabled 1 _B Interrupt on calculated TRIGGER event 4 enabled
CDTI_EIRQ_EN	23	rw	Enable interrupt when calculation of TRIGGER duration done 0 _B No Interrupt on calculated TRIGGER duration enabled 1 _B Interrupt on calculated TRIGGER duration enabled

Generic Timer Module (GTM)

Field	Bits	Type	Description
CDSI_EIR Q_EN	24	rw	Enable interrupt when calculation of TRIGGER duration done 0 _B No Interrupt on calculated STATE duration enabled 1 _B Interrupt on calculated STATE duration enabled
TORI	25	rw	TRIGGER out of range interrupt 0 _B No Interrupt when TRIGGER is out of range enabled 1 _B Interrupt when TRIGGER is out of range enabled
SORI	26	rw	STATE out of range 0 _B No Interrupt when STATE is out of range enabled 1 _B Interrupt when STATE is out of range enabled
DCGI	27	rw	Direction change interrupt 0 _B No Interrupt when a direction change of TRIGGER is detected 1 _B Interrupt when a direction change of TRIGGER is detected
Reserved	[31:28]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.21 Register DPLL_INC_CNT1

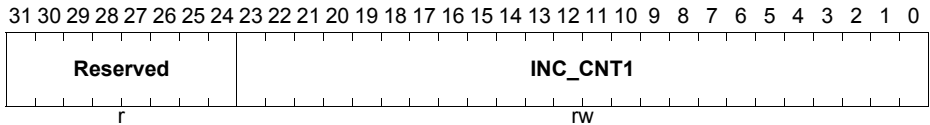
Counter value of sent SUB_INC1 pulses.

GTM_DPLL_INC_CNT1

DPLL Counter for Pulses for TBU_TS1 to be sent in Automatic End Mode

(280B0_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
INC_CNT1	[23:0]	rw	<p>Actual number of pulses to be still sent out at the current increment until the next valid input signal in automatic end mode</p> <p>Automatic addition of the number of demanded pulses MLT/MLS1 when getting a valid TRIGGER/STATE input in normal or emergency mode respectively when SGE1=1; writable only for test purposes when DEN=0</p> <p>In the case of a change of the direction the wrong number of pulses are corrected twice: Add the difference between NMB_T and INC_CNT1 twice to INC_CNT1 before sending out the correction pulses.</p> <p>Note: This value can only be written when the DPLL is disabled.</p>
Reserved	[31:24]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

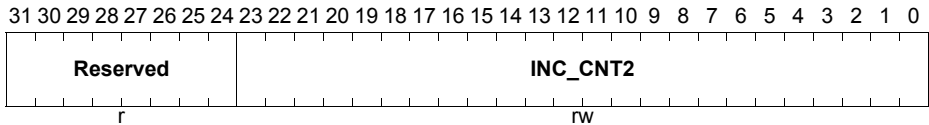
Generic Timer Module (GTM)

26.16.11.22 Register DPLL_INC_CNT2

Counter value of sent SUB_INC2 values (for SMC=1 and ROM=1).

GTM_DPLL_INC_CNT2

DPLL Counter for Pulses for TBU_TS2 to be sent in Automatic End Mode when SMC=RMO=1 (280B4_H) Reset Value: 00000000_H



Field	Bits	Type	Description
INC_CNT2	[23:0]	rw	<p>Actual number of pulses to be still sent out at the current increment until the next valid input signal in automatic end mode</p> <p>Automatic addition of the number of demanded pulses MLS2 when getting a valid TRIGGER/STATE input in normal or emergency mode respectively when SGE2=1; writable only for test purposes when DEN=0</p> <p>In the case of a change of the direction the wrong number of pulses are corrected twice: Add the difference between NMB_S and INC_CNT2 twice to INC_CNT2 before sending out the correction pulses.</p> <p>Note: This value can only be written when the DPLL is disabled.</p>
Reserved	[31:24]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

26.16.11.23 Register DPLL_APT_SYNC

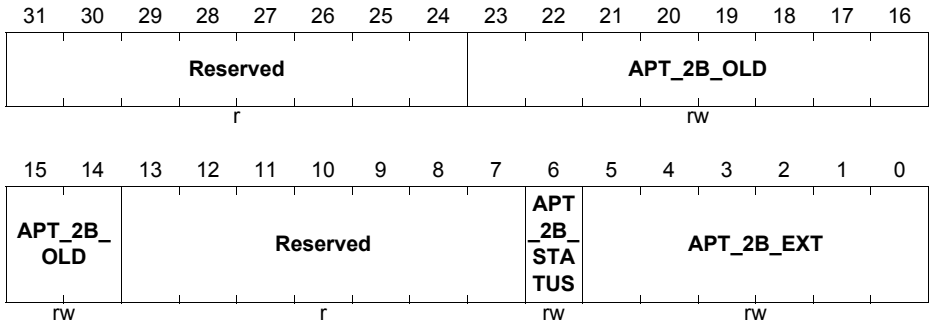
TRIGGER time stamp field offset at synchronisation time.

GTM_DPLL_APT_SYNC

DPLL Old RAM Pointer and Offset Value for TRIGGER

(280B8_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
APT_2B_EXT	[5:0]	rw	<p>Address pointer 2B extension</p> <p>Set by CPU before the synchronization is performed This offset value is the number of virtual increments to be considered for an imminent intended synchronization; the CPU sets its value dependent on the gaps until the synchronization time taking into account the considered NUTE value to be set and including the next future increment (when SYN_T_OLD is still 1). When the synchronization takes place, this value is to be added to the APT_2B address pointer (for forward direction, DIR1=0) and the APT_2B_STATUS bit is cleared after it. For backward direction subtract APT_2B_EXT accordingly.</p> <p>Note: When the synchronization is intended and the NUTE value is to be set to FULL_SCALE after it, the APT_2B_EXT value must be set to 2*SYN_NT in order to be able to fill all gaps in the extended TSF_T with the corresponding values by the CPU.</p> <p>When still not all values for FULL_SCALE are available, the APT_2B_EXT value considers only a share according to the corresponding NUTE value to be set after the synchronization.</p>
APT_2B_STATUS	6	rwh	<p>Address pointer 2B status</p> <p>Set by CPU before the synchronization is performed The value is cleared when the APT_2B_OLD value is written status bit</p> <p>0_B APT_2B_EXT is not to be considered. 1_B APT_2B_EXT has to be considered for time stamp field extension.</p>
Reserved	[13:7]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
APT_2B_OLD	[23:14]	rw	<p>Address pointer TRIGGER for RAM region 2B at synchronization time</p> <p>This value is set by the current APT_2B value when the synchronization takes place for the first valid TRIGGER event after writing APT_2C but before adding the offset value APT_2B_EXT (that means when APT_2B_STATUS=1)</p> <p>Address pointer APT_2B value at the moment of synchronization, before the offset value is added, that means the pointer with this value points to the last value before the additional inserted gap</p>
Reserved	[31:24]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

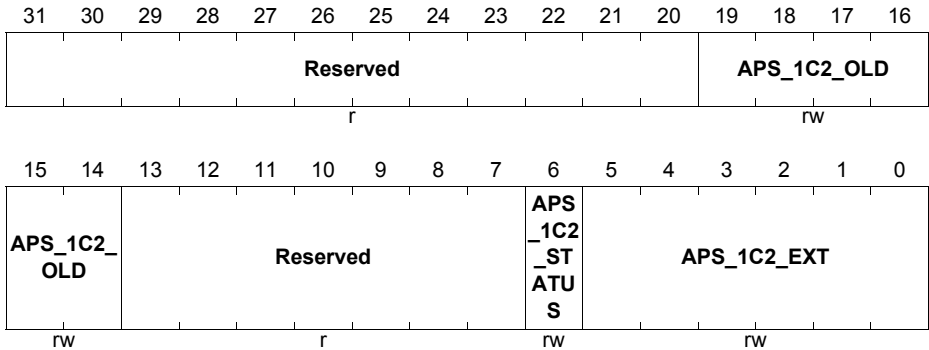
26.16.11.24 Register DPLL_APS_SYNC

STATE time stamp field offset at synchronization time.

GTM_DPLL_APS_SYNC

DPLL Old RAM Pointer and Offset Value for STATE
(280BC_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
APS_1C2_EXT	[5:0]	rw	<p>Address pointer 1C2 extension set by CPU before the synchronization is performed This offset value is the number of virtual increments to be considered for an imminent intended synchronization; the CPU sets its value dependent on the gaps until the synchronization time taking into account the considered NUSE value to be set and include the next future increment (when SYN_S_OLD is still 1). When the synchronization takes place, this value is to be added to the APS_1C2 address pointer (for forward direction, DIR2=0) and the APT_1C2_status bit is cleared after it. For backward direction subtract APS_1C2_EXT accordingly. Note: When the synchronization is intended and the NUSE value is to be set to FULL_SCALE after it, the APS_1C2_EXT value must be set to SYN_NS (for SYSF=1) or 2*SYN_NS (for SYSF=0) in order to be able to fill all gaps in the extended TSF_S with the corresponding values by the CPU. When still not all values for FULL_SCALE are available, the APS_1C2_EXT value considers only a share according to the NUSE value to be set after the synchronization.</p>
APS_1C2_STATUS	6	rwh	<p>Address pointer 1C2 status set by CPU before the synchronization is performed The value is cleared automatically when the APS_1C2_OLD value is written 0_B APT_2B_EXT is not to be considered. 1_B APT_2B_EXT has to be considered for time stamp field extension.</p>
Reserved	[13:7]	r	<p>Reserved Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
APS_1C2_ OLD	[19:14]	rw	<p>Address pointer STATE for RAM region 1C2 at synchronization time</p> <p>This value is set by the current APS_1C2 value when the synchronization takes place for the first valid STATE event after writing APS_1C3 but before adding the offset value APS_1C2_EXT (that means when APS_1C2_STATUS=1)</p> <p>Address pointer APS_1C2 value at the moment of synchronization, before the offset value is added, that means the pointer with this value points to the last value before the additional inserted gap</p>
Reserved	[31:20]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

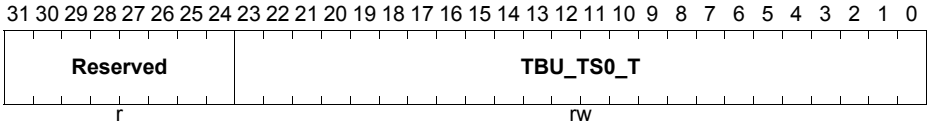
26.16.11.25 Register DPLL_TBU_TS0_T

Time stamp value for the last valid TRIGGER.

GTM_DPLL_TBU_TS0_T

DPLL TBU_TS0 Value at last TRIGGER Event
(280C0_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
TBU_TS0_T	[23:0]	rw	value of TBU_TS0 at the last TRIGGER event For each T_VALID the value of TBU_TS0 is stored in this register; The register is writable only for test purposes when DEN=0. Note: This value can only be written when the DPLL is disabled.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

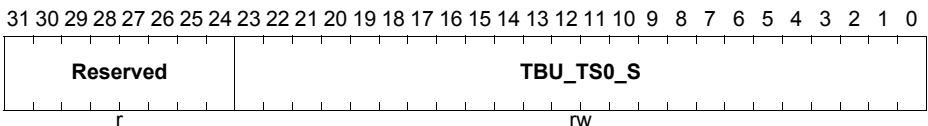
26.16.11.26 Register DPLL_TBU_TS0_S

Time stamp value for the last valid STATE.

GTM_DPLL_TBU_TS0_S

DPLL TBU_TS0 Value at last STATE Event
(280C4_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
TBU_TS0_S	[23:0]	rw	value of TBU_TS0 at the last STATE event For each S_VALID the value of TBU_TS0 is stored in this register; The register is writable only for test purposes when DEN=0. Note: This value can only be written when the DPLL is disabled.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.27 Register DPLL_ADD_IN_LD1

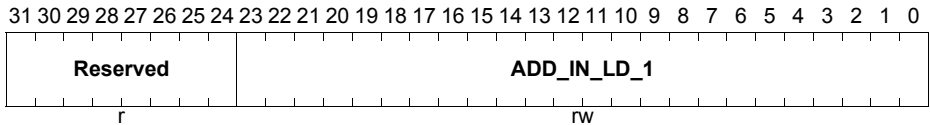
ADD_IN value in direct load mode for TRIGGER.

GTM_DPLL_ADD_IN_LD1

DPLL Direct Load Input Value for SUB_INC1

(280C8_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
ADD_IN_LD_1	[23:0]	rw	<p>Input value for SUB_INC1 generation given by CPU This value can be used in normal and emergency mode (SMC=0) as well as for SMC=1</p> <p><i>Note: The value is loaded by the CPU but used by the DPLL only for DLM1=1 (see DPLL_CTRL_1 register). When switching DLM1 to 1, the value in the register is used for the SUB_INC1 generation beginning from the next valid TRIGGER or STATE event respectively independently if new values are written by the CPU or not.</i></p> <p><i>Note: When a new value is written the output frequency changes according to the given value beginning immediately from the moment of writing. Do not wait for performing step 10 in the state machine for ADD_in calculations.</i></p> <p><i>Note: If ADD_IN_LD1 value is zero all pulses are sent with the highest possible frequency.</i></p>
Reserved	[31:24]	r	<p>Reserved Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

26.16.11.28 Register DPLL_ADD_IN_LD2

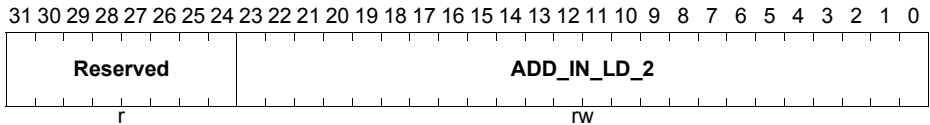
ADD_IN value in direct load mode STATE.

GTM_DPLL_ADD_IN_LD2

DPLL Direct Load Input Value for SUB_INC1

(280CC_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
ADD_IN_LD_2	[23:0]	rw	<p>Input value for SUB_INC2 generation given by CPU This value can be used for SMC=1 while RMO=1</p> <p><i>Note: The value is loaded by the CPU but used by the DPLL only for DLM2=1 (see DPLL_CRTL_1 register). When switching DLM2 to 1, the value in the register is used for the SUB_INC2 generation beginning from the next valid STATE event respectively independently if new values are written by the CPU or not.</i></p> <p><i>Note: When a new value is written the output frequency changes according to the given value beginning immediately from the moment of writing. Do not wait for performing step 30 in the state machine for ADD_in calculations.</i></p> <p><i>Note: If ADD_IN_LD2 value is zero all pulses are sent with the highest possible frequency.</i></p>
Reserved	[31:24]	r	<p>Reserved Read as zero, should be written as zero.</p>

26.16.11.29 Register DPLL_STATUS

Status register.

The DPLL_STATUS register is reset, when the DPLL is disabled (switching DEN from 1 to 0).

GTM_DPLL_STATUS
DPLL Status Register
(280FC_H)
Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ERR	LOC K1	FTD	FSD	SYT	SYS	LOC K2	Reserved	BWD 1	BWD 2	ITN	ISN	CAIP 1	CAIP 2	CSV T	CSV S
rh	rh	rh	rh	rh	rh	rh	r	rh	rh	rh	rh	rh	rh	rh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOW_RE S	Reserved	RAM 2_E RR	MT	TOR	MS	SOR	PSE	RCT	RCS	CRO	CTO	Reserved	CSO	Reserved	
rwh	r	rwh	rwh	rh	rh	rh	rwh	rwh	rh	rwh	rwh	r	rwh	r	

Field	Bits	Type	Description
Reserved	0	r	Reserved Read as zero, should be written as zero.
CSO	1	rwh	Calculated STATE duration overflow Bit is set when equations DPLL-10a or DPLL-10b lead to an overflow. 0 _B No overflow at equation DPLL-10a or b 1 _B overflow at equation DPLL-10a or b
Reserved	2	r	Reserved Read as zero, should be written as zero.
CTO	3	rwh	Calculated TRIGGER duration overflow Bit is set when equations DPLL-5a or DPLL-5b lead to an overflow. 0 _B No overflow at equation DPLL-5a or b 1 _B overflow at equation DPLL-5a or b <i>Note: When one of the above bits is set the corresponding register contains the maximum value 0xFFFFF.</i>

Generic Timer Module (GTM)

Field	Bits	Type	Description
CRO	4	rwh	<p>Calculated Reciprocal value overflow</p> <p>Bit is set when the calculation of RDT_T_actual or RDT_S_actual leads to an overflow.</p> <p>0_B No overflow at any reciprocal calculation 1_B overflow for at least one reciprocal calculation</p> <p><i>Note: An overflow in calculation of reciprocal values can occur, when the condition of Note 3) to the DPLL_CTRL_0 register is violated (see chapter 16.11.1). Such an overflow can occur according to the calculations in equations (DPLL-1c) or (DPLL-6c).</i></p> <p>The overflow is detected when after the calculation and shifting left 32 bits at least one of the bits 31 to 24 is not zero. In that case the corresponding register is set to 0xFFFFF.</p>
RCS	5	rh	<p>RCS</p> <p>0_B No resolution conflict detected 1_B the TS0_HRT value is set to 1 while LOW_RES=0</p>
RCT	6	rwh	<p>RCT</p> <p>0_B No resolution conflict detected 1_B the TS0_HRS value is set to 1 while LOW_RES=0</p>
PSE	7	rwh	<p>Prediction space configuration error</p> <p>0_B No prediction space error detected 1_B Configured offset value of RAM2 is too small in order to store all TNU+1 values twice in FULL_SCALE</p>
SOR	8	rh	<p>STATE out of range</p> <p>0_B all STATE signal events appear within SLR interval or a direction change was detected 1_B at least one STATE signal event is out of SLR; address pointers APS, APS_1C2, and APS_1C3 are frozen and generated of pulse SUB_INC1,2 respectively is stopped</p> <p>The SOR bit is set, when the time to the next active STATE slope exceeds the value of the last nominal STATE duration multiplied with the value of the SLR register (see chapter 16.13.37) and is reset, when at the current or last valid input event a direction change was detected. the SYS bit is not influenced by setting the SOR bit.</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
MS	9	rh	Missing STATE detected according to TOV_S 0 _B No missing STATE detected or a new valid STATE slope occurred 1 _B At least one missing STATE detected after the last valid slope
TOR	10	rh	TRIGGER out of range 0 _B all TRIGGER signal events appear within TLR interval or a direction change was detected 1 _B at least one TRIGGER signal event is out of TLR; address pointers APT, APT_2B, and APT_2C are frozen and generated of pulse SUB_INC1 respectively is stopped The TOR bit is set, when the time to the next active TRIGGER slope exceeds the value of the last nominal TRIGGER duration multiplied with the value of the TLR register (see chapter 16.13.36) and is reset, when at the current or last valid input event a direction change was detected. The SYT bit is not influenced by setting the TOR bit.
MT	11	rwh	Missing TRIGGER detected according to TOV 0 _B No missing TRIGGER detected or a new valid TRIGGER slope occurred 1 _B At least one missing TRIGGER detected after the last valid slope
RAM2_ER R	12	rwh	DPLL internal access to not configured RAM2 0 _B No access to not configured RAM2 memory space 1 _B access to not configured RAM2 memory space
Reserved	[14:13]	r	Reserved Read as zero, should be written as zero.
LOW_RES	15	rwh	Low resolution of TBU_TS0 is used for DPLL input This value reflects the input signal LOW_RES. 0 _B the lower 24 Bits of TBU_TS0 are used as input for the DPLL 1 _B the higher 24 Bits of TBU_TS0 are used as input for the DPLL
CSVS	16	rh	Current signal value STATE 0 _B the last STATE_S value was 0 1 _B the last STATE_S value was 1
CSVT	17	rh	Current signal value TRIGGER 0 _B the last TRIGGER_S value was 0 1 _B the last TRIGGER_S value was 1

Generic Timer Module (GTM)

Field	Bits	Type	Description
CAIP2	18	rh	Calculation of actions 12 to 23 in progress (2nd part) 0_B currently no action calculation, new data requests possible 1_B action calculation in progress, no new data requests possible
CAIP1	19	rh	Calculation of actions 0 to 11 in progress (1st part) 0_B currently no action calculation, new data requests possible 1_B action calculation in progress, no new data requests possible
ISN	20	rh	Number of STATE is not plausible Bit is set when the number of STATES is different to profile. 0_B the number of STATE events between synchronization gaps is plausible, a direction change is detected or the APS_1C3 pointer is written 1_B after setting LOCK1 in emergency mode (SMC=0 and RMO=1) or LOCK2 for SMC=RMO=1 missing or additional STATE signals detected; bit is cleared when a direction change is detected or the APS_1C3 is written
ITN	21	rh	Increment number of TRIGGER is not plausible Bit is set when the number of TRIGGERS is different to profile. 0_B the number of TRIGGER events between synchronization gaps is plausible, a direction change is detected or the address pointer APT_2C is written 1_B after setting LOCK1 in normal mode (for SMC=0 or SMC=1) or in emergency mode (only for SMC=0) for missing or additional TRIGGER signals detected; bit is cleared when a direction change is detected or the APT_2C is written
BWD2	22	rh	Backwards drive of SUB_INC2 0_B forward direction 1_B backward direction
BWD1	23	rh	Backwards drive of SUB_INC1 0_B forward direction 1_B backward direction
Reserved	24	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

Field	Bits	Type	Description
LOCK2	25	rh	<p>DPLL Lock status concerning SUB_INC2</p> <p>0_B The DPLL is not locked concerning STATE for SMC=1 1_B The DPLL is locked concerning STATE for SMC=1</p> <p><i>Note: Locking of SUB_INC2 appears for RMO=SMC=1: Bit is set, when SYS is set and the number of events between two missing STATES is as expected by the SYN_S values.</i></p> <p><i>Note: LOCK2 is set for SMC=RMO=1: for a valid STATE event when SYS is set and SYN_NS=0 for a valid STATE event when SYS is set and SYN_NS=0 or when SYS is set and the profile stored in the ADT_Si field matches once between two gaps.</i></p> <p><i>Note: LOCK2 is reset: for SMC=RMO=1</i> - when a missing STATE event occurs while SYN_S=1. This does mean an unexpected missing STATE. - when the corresponding input signal STATE is out of locking range SLR.</p>
SYS	26	rh	<p>Synchronization condition of STATE fixed</p> <p>This bit is set when the CPU writes to the APS_1C3 address pointer.</p>
SYT	27	rh	<p>Synchronization condition of TRIGGER fixed</p> <p>This bit is set when the CPU writes to the APT_2C address pointer.</p>
FSD	28	rh	<p>STATE detected</p> <p>0_B Still no valid STATE event was detected after enabling DPLL 1_B At least one valid STATE event was detected after enabling DPLL</p> <p><i>Note: No change of FSD for switching from normal to emergency mode or vice versa.</i></p>
FTD	29	rh	<p>First TRIGGER detected</p> <p>0_B No valid TRIGGER event was detected after enabling DPLL 0_B At least one valid TRIGGER event was detected after enabling DPLL</p> <p><i>Note: No change of FTD for switching from normal to emergency mode or vice versa.</i></p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
LOCK1	30	rh	<p>DPLL Lock status concerning SUB_INC1</p> <p>0_B The DPLL is not locked for TRIGGER (while SMC=RMO=0 or SMC=1) or for STATE (while SMC=0 and RMO=1)</p> <p>1_B The DPLL is locked for TRIGGER (while SMC=RMO=0 or SMC=1) or for STATE (while SMC=0 and RMO=1)</p> <p>LOCK1 is set:</p> <ul style="list-style-type: none"> - in normal mode (for RMO=SMC=0, LCD=0): Bit is set for a valid TRIGGER event when SYT is set and the number of events between two gaps is as expected by the profile (NT values in the ADT_T[i] field) or when SYN_NT=0 and SYT=1. - in normal mode (for RMO=SMC=0, LCD=1): Bit is set for a valid TRIGGER event when SYT is set and the number of events between two increments without missing TRIGGER (no gap) is as expected by profile (NT values in the ADT_Ti filed). - in emergency mode (for RMO=1 and SMC=0): Bit is set for a valid STATE event, when SYS is set and the received events are in correspondence to the profile (NS values in the ADT_S[i] field) for at least two expected missing STATE events or when SYN_NS=0. - for SMC=1: Bit is set for a valid TRIGGER even when SYT is set and SYN_NT=0 or when SYT is set and the profile stored in the ADT_T[i] field matches once between two gaps. <p>LOCK1 is reset for RMO=SMC=0:</p> <ul style="list-style-type: none"> - when a corresponding missing TRIGGER event occurs while SYN_T=1. this does mean an unexpected missing TRIGGER. - when the corresponding input signal TRIGGER is out of locking range TLR, - when a corresponding direction change is detected for RMO=1 and SMC=0: - when a corresponding missing STATE event occurs while SYN_S=1. This does mean an unexpected missing STATE. - when the corresponding input signal STATE is out of locking range TLR <p>for SMC=1:</p> <ul style="list-style-type: none"> - when a corresponding missing TRIGGER event occurs while SYN_T=1. This does mean an unexpected missing TRIGGER. - when the corresponding input signal TRIGGER is out of locking range TLR - when a corresponding direction change is detected

Generic Timer Module (GTM)

Field	Bits	Type	Description
ERR	31	rh	Error during configuration or operation resulting in unexpected values 0_{B} when all bits in position 8 to 0 and 10 and 12 are zero 1_{B} when at least one bit in position 8 to 0 or 10 or 12 is one

Generic Timer Module (GTM)

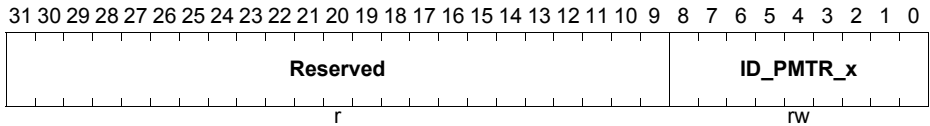
26.16.11.30 Register DPLL_ID_PMTR_x

ID information for input signal PMTRx (position minus time request), x=0...23.

GTM_DPLL_ID_PMTR_x (x=0-23)

DPLL ID Information For Input Signal PMTR x Register
(28100_H+x*04_H)

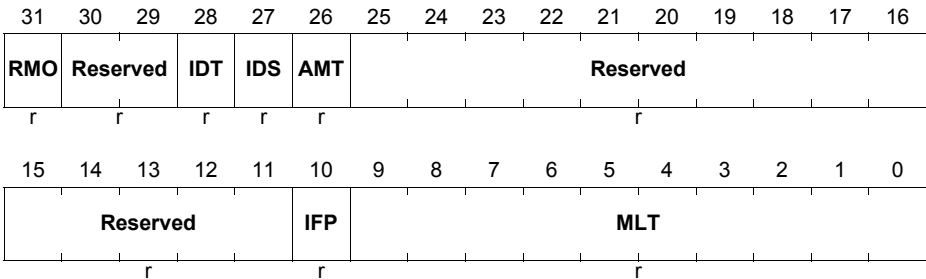
Reset Value: 000001FE_H



Field	Bits	Type	Description
ID_PMTR_x	[8:0]	rw	<p>ID information to the input signal PMTR_x from the ARU</p> <p><i>Note: The DPLL_ID_PMTR_x can be changed also when the DPLL is enabled.</i></p>
Reserved	[31:9]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)
26.16.11.31 Register DPLL_CTRL_0_SHADOW_TRIGGER

Shadow register of DPLL_CTRL_0 controlled by valid TRIGGER slope.

GTM_DPLL_CTRL_0_SHADOW_TRIGGER
DPLL Control0 Shadow Trigger Register(281E0_H)
Reset Value: 00000257_H


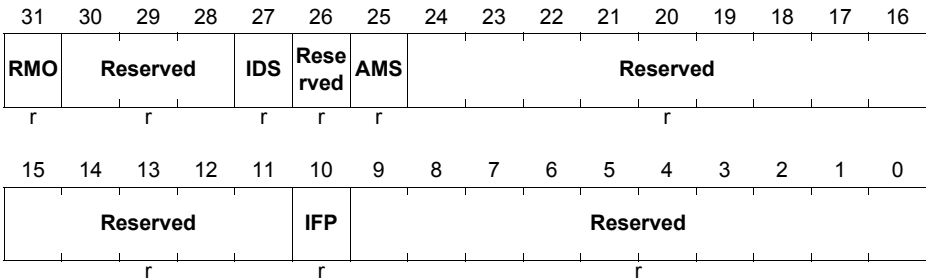
Field	Bits	Type	Description
MLT ¹⁾	[9:0]	r	multiplier for TRIGGER MLT+1 is number of SUB_INC1 pulses between two TRIGGER events in normal mode (11024).
IFP ¹⁾	10	r	Input filter position Value contains position or time related information.
Reserved	[25:11]	r	Reserved Read as zero, should be written as zero.
AMT ¹⁾	26	r	Adapt mode TRIGGER Use of adaptation information of TRIGGER.
IDS	27	r	Input delay STATE Use of input delay information transmitted in FT part of the STATE signal.
IDT ¹⁾	28	r	Input delay TRIGGER Use of input delay information transmitted in FT part of the TRIGGER signal.
Reserved	[30:29]	r	Reserved Read as zero, should be written as zero.
RMO ¹⁾	31	r	Reference mode; selection of the relevant the input signal for generation of SUB_INC1

Generic Timer Module (GTM)

- 1) Only the values are stored for a valid TRIGGER slope. All other values remain 0. When DEN=0 the relevant bit values of the original register DPLL_CTRL_0 are transferred without any input event at the next system clock. This results in the above reset value.

Generic Timer Module (GTM)
26.16.11.32 Register DPLL_CTRL_0_SHADOW_STATE

Shadow register of DPLL_CTRL_0 controlled by a valid STATE slope.

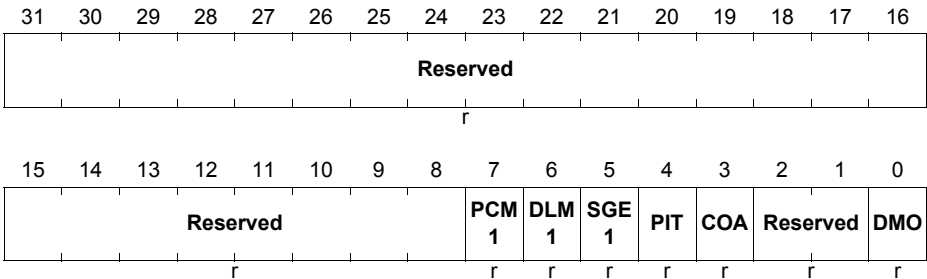
GTM_DPLL_CTRL_0_SHADOW_STATE
DPLL Control 0 Shadow STATE Register(281E4_H)
Reset Value: 00000000_H


Field	Bits	Type	Description
Reserved	[9:0]	r	Reserved Read as zero, should be written as zero.
IFP ¹⁾	10	r	Input filter position Value contains position or position or time related information.
Reserved	[24:11]	r	Reserved Read as zero, should be written as zero.
AMS ¹⁾	25	r	Adapt mode STATE Use of adaptation information of STATE.
Reserved	26	r	Reserved Read as zero, should be written as zero.
IDS ¹⁾	27	r	Input delay STATE Use of input delay information transmitted in FT part of the STATE signal.
Reserved	[30:28]	r	Reserved Read as zero, should be written as zero.
RMO ¹⁾	31	r	Reference mode; selection of the relevant the input signal for generation of SUB_INC1

1) Only the values are stored for a valid TRIGGER slope. All other values remain 0. When DEN=0 the relevant bit values of the original register DPLL_CTRL_0 are transferred without any input event at the next system clock. This results in the above reset value.

26.16.11.33 Register DPLL_CTRL_1_SHADOW_TRIGGER

Shadow register of DPLL_CTRL_1 controlled by a valid TRIGGER slope.

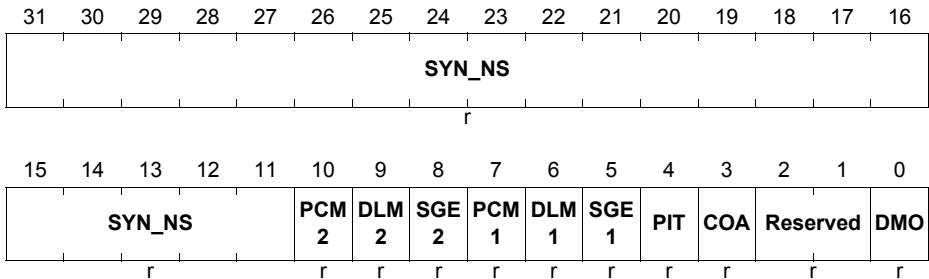
GTM_DPLL_CTRL_1_SHADOW_TRIGGER
DPLL Control 1 Shadow TRIGGER Register(281E8_H)
Reset Value: 00000000_H


Field	Bits	Type	Description
DMO ¹⁾	0	r	DPLL mode select
Reserved	[2:1]	r	Reserved Read as zero, should be written as zero.
COA ¹⁾	3	r	Correction strategy in automatic end mode (DMO=0)
PIT ¹⁾	4	r	Plausibility value PVT to next valid TRIGGER is time related
SGE1 ¹⁾	5	r	SUB_INC1 generator enable
DLM1 ¹⁾	6	r	Direct Load Mode for SUB_INC1 generation
PCM1 ¹⁾	7	r	Pulse Correction Mode for SUB_INC1 generation
Reserved	[31:8]	r	Reserved Read as zero, should be written as zero.

1) Only the values are stored for a valid TRIGGER slope. All other values remain 0. When DEN=0 the relevant bit values of the original register DPLL_CTRL_1 are transferred without any input event at the next system clock. This results in the above reset value.

26.16.11.34 Register DPLL_CTRL_1_SHADOW_STATE

Shadow register of DPLL_CTRL_1 controlled by a valid STATE slope.

GTM_DPLL_CTRL_1_SHADOW_STATE
DPLL Control 1 Shadow STATE Register(281EC_H)
Reset Value: 00000000_H


Field	Bits	Type	Description
DMO ¹⁾	0	r	DPLL mode select
Reserved	[2:1]	r	Reserved Read as zero, should be written as zero.
COA ¹⁾	3	r	Correction strategy in automatic end mode (DMO=0)
PIT	4	r	Plausibility value PVT to next valid TRIGGER is time related
SGE1 ¹⁾	5	r	SUB_INC1 generator enable
DLM1 ¹⁾	6	r	Direct Load Mode for SUB_INC1 generation
PCM1 ¹⁾	7	r	Pulse Correction Mode for SUB_INC1 generation
SGE2 ¹⁾	8	r	SUB_INC2 generator enable
DLM2 ¹⁾	9	r	Direct Load Mode for SUB_INC2 generation
PCM2 ¹⁾	10	r	Pulse Correction Mode for SUB_INC2 generation
SYN_NS	[31:11]	r	Synchronization number of STATE Summarized number of virtual increments in HALF_SCALE

Generic Timer Module (GTM)

- 1) Only the values are stored for a valid TRIGGER slope. All other values remain 0. When DEN=0 the relevant bit values of the original register DPLL_CTRL_1 are transferred without any input event at the next system clock. This results in the above reset value.

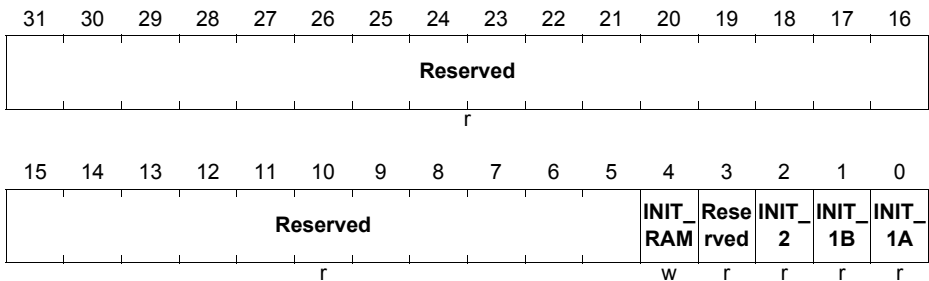
26.16.11.35 Register DPLL_RAM_INI

Register to control the RAM initialisation.

GTM_DPLL_RAM_INI

DPLL RAM Initatlisation Register (281FC_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
INIT_1A	0	r	<p>RAM region 1A initialization in progress</p> <p>0_B No initialization of considered RAM region in progress</p> <p>1_B Initialization of considered RAM region in progress</p>
INIT_1B	1	r	<p>RAM region 1B initialization in progress</p> <p>see bit 0</p>
INIT_2	2	r	<p>RAM region 2 initialization in progress</p> <p>see bit 0</p>
Reserved	3	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

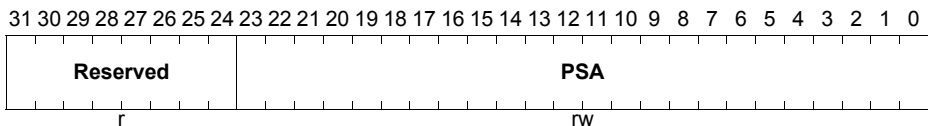
Generic Timer Module (GTM)

Field	Bits	Type	Description
INIT_RAM	4	w	RAM regions 1A, 1B and 2 are to be initialized 0_B Do not start initialization of all RAM regions 1_B Start initialization of all RAM regions Note: Setting the INIT_RAM bit results only in a RAM reset when the DPLL is not enabled (DEN=0). Note: Depending on the vendor configuration the connected RAM regions are initialized to zero in the case of a module HW reset or for setting the RST bit in the GTM_RST register. Note: In the case of no RAM initialization it must be ensured that all relevant parameters are configured correctly. Otherwise there is no guarantee to get a predictable behaviour.
Reserved	[31:5]	r	Reserved Read as zero, should be written as zero.

26.16.11.36 Memory DPLL_PSA[i]

Position request for action i, RAM1A.

GTM_DPLL_PSAi (i=0-23)
DPLL ACTION_i Position/Value Action Request Register
 $(28200_H + i * 04_H)$

 Reset Value: 00000000_H


Field	Bits	Type	Description
PSA	[23:0]	rw	Position information of a desired action (i=0...23) Get position values via ARU Note: This value can only be written when the DPLL is disabled.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.37 Memory DPLL_DLA[i]

Time to react for action i, RAM1A

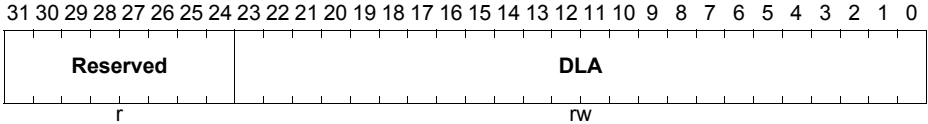
Generic Timer Module (GTM)

GTM_DPLL_DLAI (i=0-23)

DPLL ACTION_i Time To React Before PSA_i Register

$$(28280_H + i * 04_H)$$

Reset Value: 00000000_H



Field	Bits	Type	Description
DLA	[23:0]	rw	Time to react before the corresponding position value of a desired action is reached In case of LOW_RES=1 (see Chapter 26.16.4.2) this delay value must be also given as low resolution value. Note: This value can only be written when the DPLL is disabled.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.38Memory DPLL_NA[i]

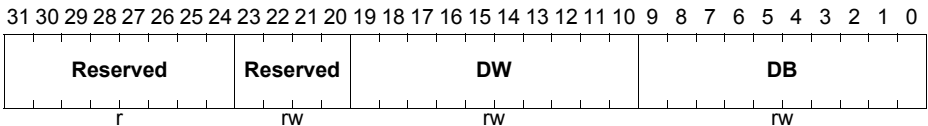
Calculated number of TRIGGER / STATE increments to action i, RAM1A.

GTM_DPLL_NAI (i=0-23)

DPLL Calculated Number Of TRIGGER/STATE Increments To ACTION_i

$$(28300_H + i * 04_H)$$

Reset Value: 00000000_H



Field	Bits	Type	Description
DB	[9:0]	rw	Number of events to Action_i (fractional part) Note: This value can only be written when the DPLL is disabled.
DW	[19:10]	w	Number of events to Action_i (integer part) Note: This value can only be written when the DPLL is disabled.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[23:20]	rw	Reserved Note: must be written as zero.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero. Note: Use the maximum value for DW=0x3FF in the case of a calculated value which exceeds the representable value.

Generic Timer Module (GTM)

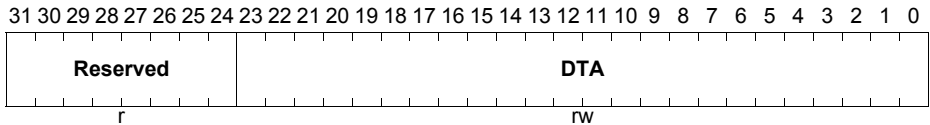
26.16.11.39Memory DPLL_DTA[i]

Calculated relative time to action i, RAM1A.

GTM_DPLL_DTAi (i=0-23)

DPLL Calculated Relative Time To ACTION_i Register
 (28380_H + i * 04_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
DTA	[23:0]	rw	Calculated relative time to ACTION_i Note: This value can only be written when the DPLL is disabled. The DTA value is a positive integer value. When calculations using equations DPLL-12 or DPLL-14 result in a negative value, it is replaced by zero.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.40Memory DPLL_TS_T

Actual TRIGGER time stamp value.

GTM_DPLL_TS_T_0

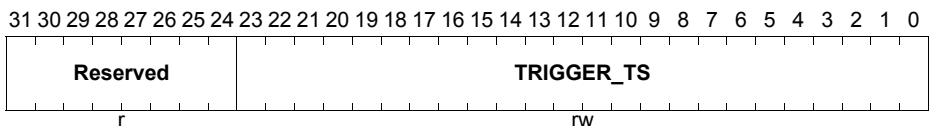
DPLL Actual Signal TRIGGER Time Stamp Register
 (28400_H)

Reset Value: 00000000_H

GTM_DPLL_TS_T_1

DPLL Actual Signal TRIGGER Time Stamp Register
 (28404_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
TRIGGER_TS	[23:0]	rw	Time stamp value of the last valid TRIGGER input measured TRIGGER time stamp Note: The LSB address is determined using the SWON_T value in the OSW register (see Section 26.16.11.7).
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.41 Memory DPLL_TS_T_OLD

Previous TRIGGER time stamp value.

GTM_DPLL_TS_T_OLD_0

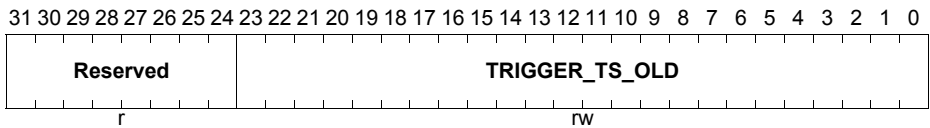
DPLL Previous Signal TRIGGER Time Stamp Register
(28400_H)

Reset Value: 00000000_H

GTM_DPLL_TS_T_OLD_1

DPLL Previous Signal TRIGGER Time Stamp Register
(28404_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
TRIGGER_TS_old	[23:0]	rw	Time stamp value of the last but one valid TRIGGER input previous measured TRIGGER time stamp Note: The LSB address is determined using the SWON_T value in the OSW register (see Section 26.16.11.7).
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

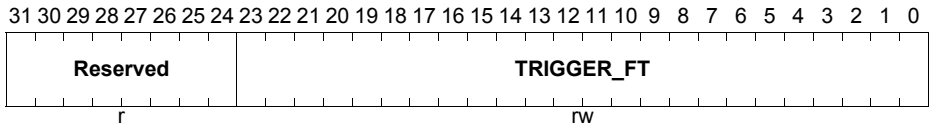
26.16.11.42 Memory DPLL_FTV_T

Actual TRIGGER filter value.

GTM_DPLL_FTV_T

DPLL Actual Signal TRIGGER Filter Value Register
(28408_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
TRIGGER_FT	[23:0]	rw	Filter value of the last valid TRIGGER input transmitted filter value Note: The LSB address is determined using the SWON_T value in the OSW register (see Section 26.16.11.7).
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.43 Memory DPLL_TS_S

Actual STATE time stamp register.

GTM_DPLL_TS_S_0

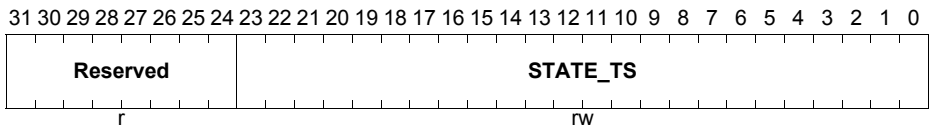
DPLL Actual Signal STATE Time Stamp Register
(28410_H)

Reset Value: 00000000_H

GTM_DPLL_TS_S_1

DPLL Actual Signal STATE Time Stamp Register
(28414_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
STATE_TS	[23:0]	rw	Time stamp value of the last valid STATE input Note: The LSB address is determined using the SWON_S value in the OSW register (see Section 26.16.11.7).
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.44 Memory DPLL_TS_S_OLD

Previous STATE time stamp register.

GTM_DPLL_TS_S_OLD_0

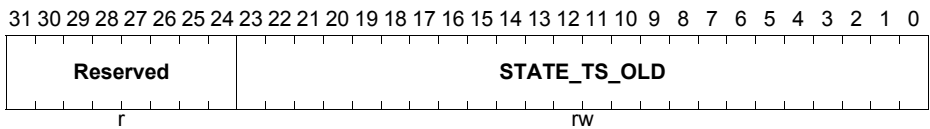
DPLL Previous Signal STATE Time Stamp Register
(28410_H)

Reset Value: 00000000_H

GTM_DPLL_TS_S_OLD_1

DPLL Previous Signal STATE Time Stamp Register
(28414_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
STATE_T S_OLD	[23:0]	rw	Time stamp value of the last valid STATE input Note: The LSB address is determined using the SWON_S value in the OSW register (see Section 26.16.11.7).
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

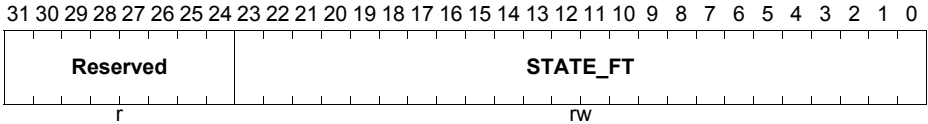
26.16.11.45 Memory DPLL_FTV_S

Actual STATE filter value.

GTM_DPLL_FTV_S

DPLL Actual Signal STATE Filter Value Register (28418_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
STATE_FT	[23:0]	rw	Filter value of the last valid STATE input transmitted filter value Note: The LSB address is determined using the SWON register (see Section 26.16.11.7).
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

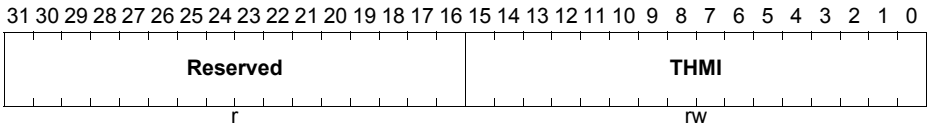
26.16.11.46 Memory DPLL_THMI

TRIGGER hold time min value.

GTM_DPLL_THMI

DPLL TRIGGER hold time min value (28420_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
THMI	[15:0]	rw	<p>Minimal time between active and inactive TRIGGER slope (uint16)</p> <p>the time value corresponds to the time stamp clock counts; this does mean the clock selected for the TBU_CHO_BASE (see TBU_CHOCTRL register). set min value; generate the TINI interrupt in the case of a violation for THMI>0.</p> <p>Note: Typical retention time values after a valid slope can be e.g. between 45 μs (forwards) and 90 μs (backwards). When THMI is zero, consider always a THMI violation (forwards).</p>
Reserved	[31:16]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

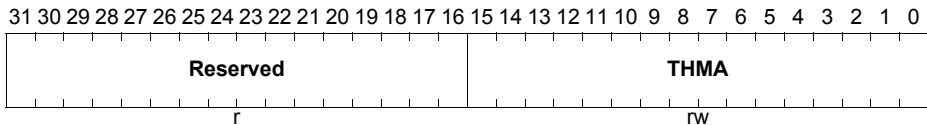
26.16.11.47 Memory DPLL_THMA

TRIGGER hold time max value.

GTM_DPLL_THMA

DPLL TRIGGER Hold Time Max Value(28424_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
THMA	[15:0]	rw	Maximal time between active and inactive TRIGGER slope (uint16) the time value corresponds to the time stamp clock counts; this does mean the clock selected for the TBU_CH0_BASE (see TBU_CH0_CTRL register). max value to be set; generate the TAX interrupt in the case of a violation for THMA>0.
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero.

26.16.11.48 Memory DPLL_THVAL

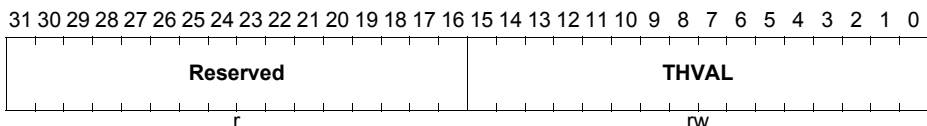
Measured TRIGGER hold time value.

GTM_DPLL_THVAL

DPLL Measured Last Pulse Time from Valid to Invalid TRIGGER Slope

(28428_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
THVAL	[15:0]	rw	Measured time from the last valid slope to the next inactive TRIGGER slope in time stamp clock counts: this does mean the clock selected for the TBU_CH0_BASE (uint16) measured value
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero.

Note: In the case of LOW_RES=1 and TBU_HRT=0 the difference between the time stamps of valid and invalid slope is multiplied by 8. The register contains this value.

Generic Timer Module (GTM)

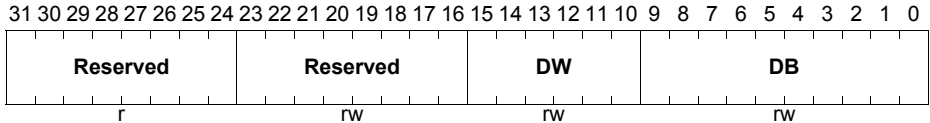
26.16.11.49 Memory DPLL_TOV

Time out value active TRIGGER slope (for missing TRIGGER generation).

GTM_DPLL_TOV

DPLL Time Out Value of active TRIGGER Slope
(28430_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
DB	[9:0]	rw	Decision value (fractional part) for missing TRIGGER interrupt
DW	[15:10]	rw	Decision value (integer part) for missing TRIGGER interrupt TOV(15:0) is to be multiplied with the duration of the last increment and divided by 1024 in order to get the timeout time value for a missing trigger event. Note: For the case of LOW_RES=1 (see DPLL_STATUS register) consider for the calculation of the time out value the following cases: LOW_RES=1 and DPLL_CTRL_1/TS0_HRT=1: multiply the TBU_TS0 value by 8 LOW_RES=1 and DPLL_CTRL_1/TS0_HRT=0: multiply the TBU_TS0 value by 8 multiply the estimated time point value (using TS_T, dt_t_ACT and TOV) by 8 LOW_RES=0 and DPLL_CTRL_1/TS0_HRT=0: use TBU_TS0 and the estimated time point value unchanged.
Reserved	[23:16]	r	Reserved Must be written as zero.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

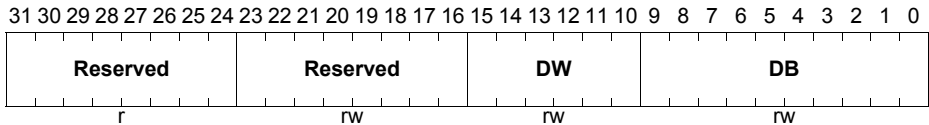
Generic Timer Module (GTM)
26.16.11.50 Memory DPLL_TOV_S

Time out value of active STATE slope (for a missing STATE generation).

GTM_DPLL_TOV_S

DPLL Time Out Value of active STATE Slope
(28434_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
DB	[9:0]	rw	Decision value (fractional part) for missing STATE interrupt
DW	[15:10]	rw	Decision value (integer part) for missing STATE interrupt TOV_S (15:0) is to be multiplied with the duration of the last increment and divided by 1024 in order to get the timeout time value for a missing STATE event. Note: For the case of LOW_RES=1 (see DPLL_STATUS register) consider for the calculation of the time out value the following cases: LOW_RES=1 and DPLL_CTRL_1/TS0_HRS=1: multiply the TBU_TS0 value by 8 LOW_RES=1 and DPLL_CTRL_1/TS0_HRS=0: multiply the TBU_TS0 value by 8 multiply the estimated time point value (using TS_T, dt_s_ACT and TOV_S) by 8 LOW_RES=0 and DPLL_CTRL_1/TS0_HRS=0: use TBU_TS0 and the estimated time point value unchanged.
Reserved	[23:16]	r	Reserved Must be written as zero.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.51 Memory DPLL_ADD_IN_CAL1

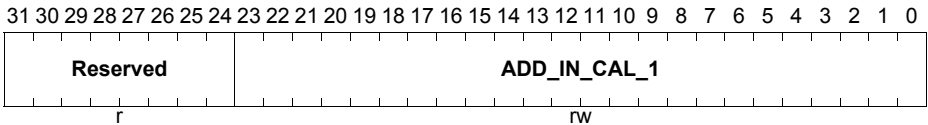
Calculated ADD_IN value for SUB_INC1 generation.

GTM_DPLL_ADD_IN_CAL1

DPLL Calculated ADD_IN Value for SUB_INC1 Generation

(28438_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
ADD_IN_CAL_1	[23:0]	rw	Calculated input value for SUB_INC1 generation, calculated by the DPLL calculated value The update of the ADD_IN value by the new calculated value ADD_IN_CAL1 is suppressed for one increment when an unexpected missing TRIGGER (SMC = 1 or RMO = 0) or an unexpected STATE (RMO = 1 and SMC = 0) is detected.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.52 Memory DPLL_ADD_IN_CAL2

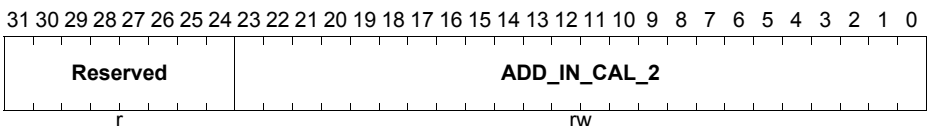
Calculated ADD_IN value for SUB_INC2 generation.

GTM_DPLL_ADD_IN_CAL2

DPLL Calculated ADD_IN Value for SUB_INC2 Generation

(2843C_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
ADD_IN_CAL_2	[23:0]	rw	<p>Input value for SUB_INC2 generation, calculated by the DPLL for SMC=RMO=1 calculated value</p> <p>The update of the ADD_IN value by the calculated value ADD_IN_CAL2 is suppressed for one increment when an unexpected missing STATE(RMO = SMC = 1) is detected.</p>
Reserved	[31:24]	r	<p>Reserved Read as zero, should be written as zero.</p>

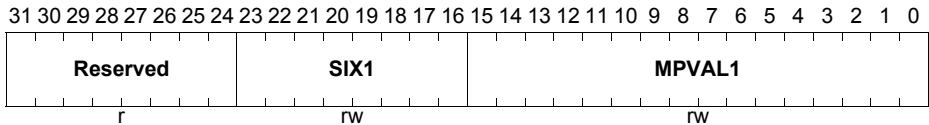
Generic Timer Module (GTM)

26.16.11.53 Memory DPLL_MPVAL1

Missing pluses to be added or subtracted directly.

GTM_DPLL_MPVAL1

DPLL Missing Pulses to be Added/Subtracted Directly to SUB_INC1 and INC_CNT1 Once (28440_H) Reset Value: 00000000_H



Field	Bits	Type	Description
MPVAL1	[15:0]	rw	<p>Missing pulses for direct correction of SUB_INC1 pulses by the CPU (sint16)</p> <p>used only for RMO=0 or SMC=1 for the case PCM1=1. Add MPVAL1 once to INC_CNT1 and reset PCM1 after applying once</p> <p>Note: Do not provide negative values which exceed the amount of NT*(MLT+1) or MLS1 respectively; when considered negative PD values the sum of both should not exceed the amount of NT*(MLT+1) or MLS1 respectively.</p>
SIX1	[23:16]	rw	<p>sign extension for MPVAL1</p> <p>0x00 =MPVAL1 is a positive number 0xFF =MPVAL1 is a negative number</p> <p>Note: All bits must be written to either all zeros or all ones.</p>
Reserved	[31:24]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p> <p>Note: Do not provide negative values which exceed the amount of NT*(MLT+1) or MLS1 respectively; when considered negative PD values the sum of both should not exceed the amount of NT*(MLT+1) or MLS1 respectively.</p>

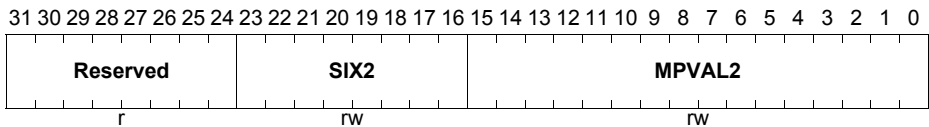
Generic Timer Module (GTM)

26.16.11.54 Memory DPLL_MPVAL2

Missing pulses to be added or subtracted directly.

GTM_DPLL_MPVAL2

DPLL Missing Pulses to be Added/Subtracted Directly to SUB_INC2 and
 INC_CNT2 Once (28444_H) Reset Value: 00000000_H



Field	Bits	Type	Description
MPVAL2	[15:0]	rw	missing pulses for direct correction of SUB_INC2 pulses by the CPU (sint16) used only for SMC=RMO=1 for the case PCM2=1. Add MPVAL2 once to INC_CNT2 and reset PCM2 after applying once Note: Do not provide negative values which exceed the amount of MLS2; when considered negative PD_S values the sum of both should not exceed the amount of MLS2.
SIX2	[23:16]	rw	sign extension for MPVAL2 0x00 =MPVAL2 is a positive number 0xFF =MPVAL2 is a negative number Note: All bits must be written to either all zeros or all ones.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

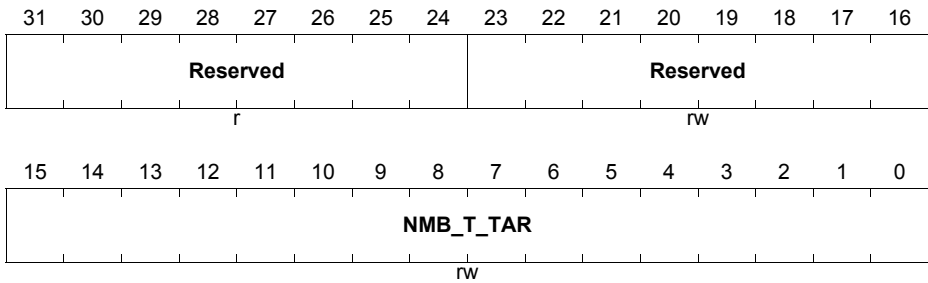
26.16.11.55 Memory DPLL_NMB_T_TAR

target number of pluses to be sent in normal mode.

GTM_DPLL_NMB_T_TAR

DPLL Target Number of Pulses to be sent in normal mode Register (28448_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
NMB_T_TAR	[15:0]	rw	<p>Target Number of pulses for TRIGGER</p> <p>Calculated number of pulses in normal mode for the current TRIGGER increment without missing pulses. calculated target pulse number</p> <p><i>Note: The LSB address is determined using the SWON_S value in the OSW register (see Chapter 26.16.11.7).</i></p>
Reserved	[23:16]	rw	<p>Reserved</p> <p>Note: must be written to zero.</p>
Reserved	[31:24]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

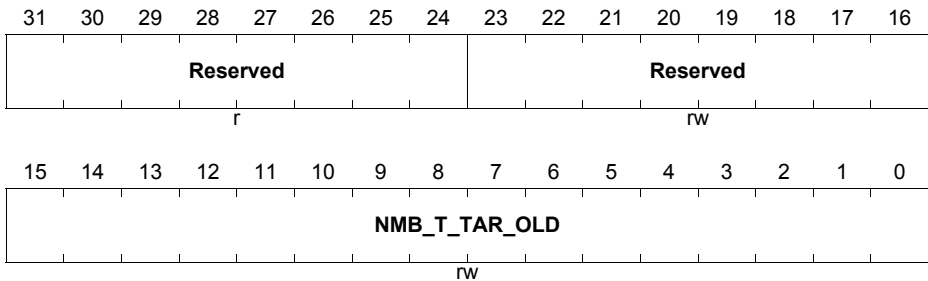
26.16.11.56 Memory DPLL_NMB_T_TAR_OLD

Last but one target number of pulses to be sent in normal mode.

GTM_DPLL_NMB_T_TAR_OLD

DPLL Target Number of Pulses to be sent
in normal mode Register (2844C_H)

Reset Value: 00000000_H

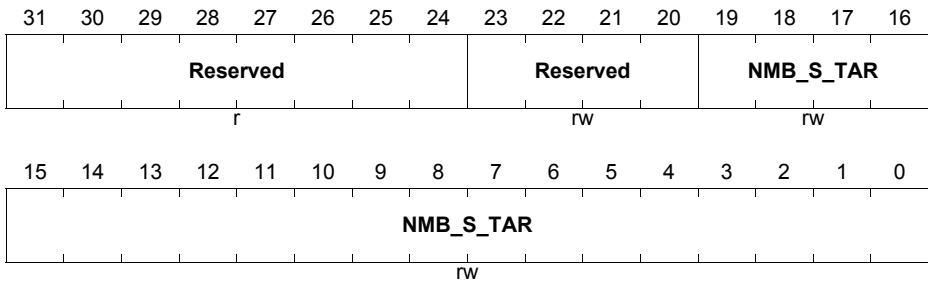


Field	Bits	Type	Description
NMB_T_TAR_OLD	[15:0]	rw	<p>Target Number of pulses for TRIGGER</p> <p>Calculated number of pulses in normal mode for the current TRIGGER increment without missing pulses. calculated target pulse number</p> <p><i>Note: The LSB address is determined using the SWON_S value in the OSW register (see Chapter 26.16.11.7).</i></p>
Reserved	[23:16]	rw	<p>Reserved</p> <p>Note: must be written to zero.</p>
Reserved	[31:24]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

26.16.11.57 Memory DPLL_NMB_S_TAR

Target number of pulses to be sent in emergency mode.

GTM_DPLL_NMB_S_TAR
**DPLL Target Number of Pulses to be sent
in emergency mode Register (28450_H)**
Reset Value: 00000000_H


Field	Bits	Type	Description
NMB_S_TAR	[19:0]	rw	Target Number of pulses for STATE Calculated number of pulses in emergency mode for the current STATE increment without missing pulses. calculated target pulse number. <i>Note: The LSB address is determined using the SWON_S value in the OSW register (see Chapter 26.16.11.7).</i>
Reserved	[23:20]	rw	Reserved Note: must be written to zero.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

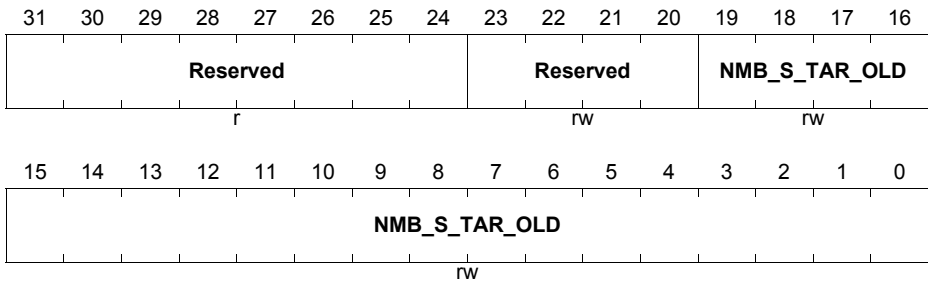
26.16.11.58 Memory DPLL_NMB_S_TAR_OLD

Last but one target number of pulses to be sent in emergency mode.

GTM_DPLL_NMB_S_TAR_OLD

**DPLL Target Number of Pulses to be sent
in emergency mode Register (28454_H)**

Reset Value: 00000000_H



Field	Bits	Type	Description
NMB_S_TAR_OLD	[19:0]	rw	Target Number of pulses for STATE Calculated number of pulses in emergency mode for the current STATE increment without missing pulses. calculated target pulse number. <i>Note: The LSB address is determined using the SWON_S value in the OSW register (see Chapter 26.16.11.7).</i>
Reserved	[23:20]	rw	Reserved Note: must be written to zero.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.59 Memory DPLL_RCDT_TX

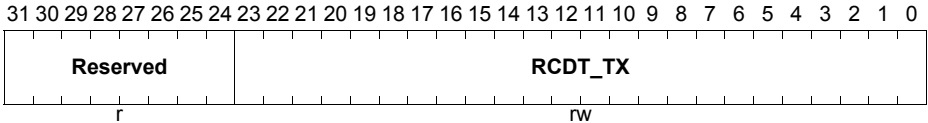
reciprocal value of the expected increment duration of TRIGGER.

GTM_DPLL_RCDT_TX

DPLL Reciprocal Value of Expected Increment Duration TRIGGER

(28460_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
RCDT_TX	[23:0]	rw	Reciprocal value of expected increment duration *2 ³² while only the lower 24 bits are used calculated value; when an overflow occurs in calculation the value is set to 0xFFFFF.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.60 Memory DPLL_RCDT_SX

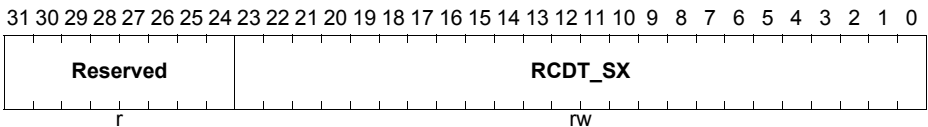
Reciprocal value of the expected increment duration of STATE.

GTM_DPLL_RCDT_SX

DPLL Reciprocal Value of Expected Increment Duration STATE

(28464_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
RCDT_SX	[23:0]	rw	Reciprocal value of expected increment duration *2 ³² while only the lower 24 bits are used calculated value; when an overflow occurs in calculation the value is set to 0xFFFFF.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

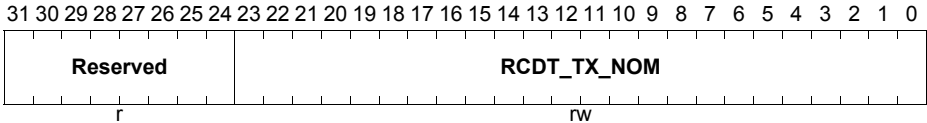
Generic Timer Module (GTM)

26.16.11.61Memory DPLL_RCDT_TX_NOM

Reciprocal value of the expected nominal increment duration of TRIGGER.

GTM_DPLL_RCDT_TX_NOM

DPLL Reciprocal Value of the Expected Nominal Increment Duration TRIGGER
(28468_H) **Reset Value: 00000000_H**



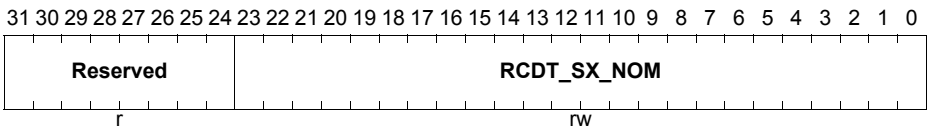
Field	Bits	Type	Description
RCDT_TX_NOM	[23:0]	rw	Reciprocal value of nominal increment duration *2 ³² while only the lower 24 bits are used calculated value; when an overflow occurs in calculation the value is set to 0xFFFFF.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.62Memory DPLL_RCDT_SX_NOM

Reciprocal value of the expected nominal increment duration of STATE.

GTM_DPLL_RCDT_SX_NOM

DPLL Reciprocal Value of the Expected Nominal Increment Duration STATE
(2846C_H) **Reset Value: 00000000_H**



Field	Bits	Type	Description
RCDT_SX_NOM	[23:0]	rw	Reciprocal value of nominal increment duration *2 ³² while only the lower 24 bits are used calculated value; when an overflow occurs in calculation the value is set to 0xFFFFF.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Note: RCDT_TX_NOM and RCDT_SX_NOM are calculated by the values RCDT_TX and RCDT_SX to be multiplied with SYN_T or SYN_S respectively.

Generic Timer Module (GTM)

26.16.11.63Memory DPLL_RDT_T_ACT

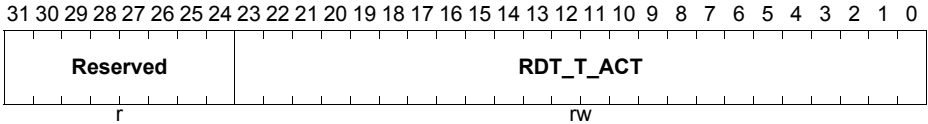
Reciprocal value of the last increment of TRIGGER.

GTM_DPLL_RDT_T_ACT

DPLL Actual Reciprocal Value of TRIGGER

(28470_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
RDT_T_A CT	[23:0]	rw	Reciprocal value of last TRIGGER increment *2³² while only the lower 24 bits are used the LSB is rounded up when the next truncated bit is 1. calculated value; when an overflow occurs in calculation the value is set to 0xFFFFF and the CRO bit in the DPLL_STATUS register is set.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

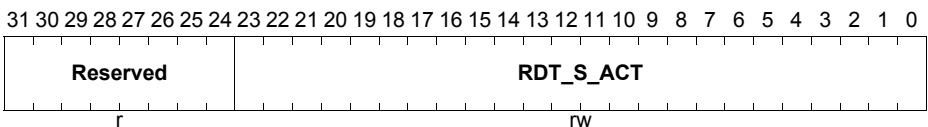
26.16.11.64Memory DPLL_RDT_S_ACT

Reciprocal value of the last increment of STATE.

GTM_DPLL_RDT_S_ACT

DPLL Actual Reciprocal Value of STATE(28474_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
RDT_S_A CT	[23:0]	rw	Reciprocal value of last STATE increment *2³² while only the lower 24 bits are used and the LSB is rounded up when the next truncated bit is 1. calculated value; when an overflow occurs in calculation the value is set to 0xFFFFF and the CRO bit in the DPLL_STATUS register is set.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

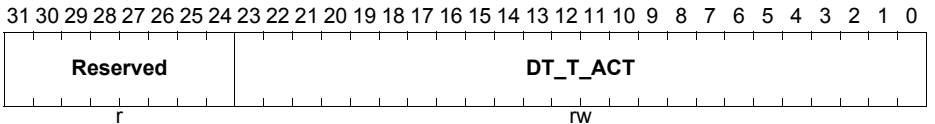
Generic Timer Module (GTM)

26.16.11.65 Memory DPLL_DT_T_ACT

Duration of the last TRIGGER increment.

GTM_DPLL_DT_T_ACT

DPLL Duration of Last TRIGGER Increment (28478_H) **Reset Value: 00000000_H**



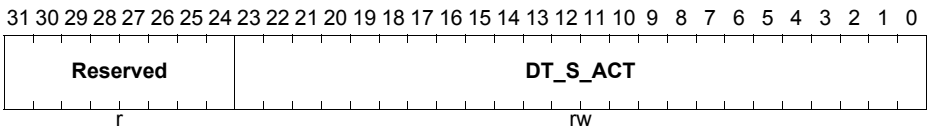
Field	Bits	Type	Description
DT_T_ACT	[23:0]	rw	Calculated duration of the last TRIGGER increment calculated duration of the last increment; Value will be written into the corresponding RAM field, when all calculations for the considered increment are done and APT is valid.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.66 Memory DPLL_DT_S_ACT

Duration of the last STATE increment.

GTM_DPLL_DT_S_ACT

DPLL Duration of Last STATE Increment
[DT_S_ACT] (2847C_H) **Reset Value: 00000000_H**



Field	Bits	Type	Description
DT_S_ACT	[23:0]	rw	Calculated duration of the last STATE increment Calculated increment duration Value will be written into the corresponding RAM field, when all calculations for the considered increment are done and APS is valid.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

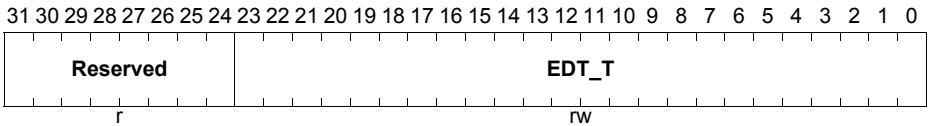
26.16.11.67Memory DPLL_EDT_T

Difference of prediction to actual value of the last TRIGGER increment.

GTM_DPLL_EDT_T

DPLL Difference of prediction to actual value of the last TRIGGER increment (28480_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
EDT_T	[23:0]	rw	Signed difference between actual value and a simple prediction of the last TRIGGER increment: sint24 calculated error value
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

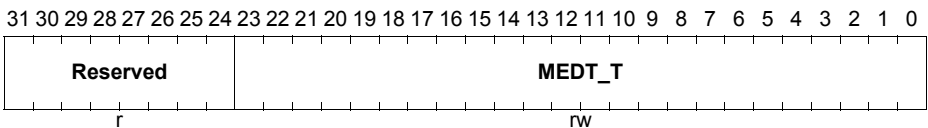
26.16.11.68Memory DPLL_MEDT_T

Weighted difference of prediction errors of TRIGGER.

GTM_DPLL_MEDT_T

DPLL Weighted difference of Prediction up to the Last TRIGGER Increment (28484_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
MEDT_T	[23:0]	rw	<p>Signed middle weighted difference between actual value and prediction of the last TRIGGER increments: sint24</p> <p>only calculated for SYT=1 calculated medium error value The value is calculated only after synchronization (SYT=1) and the update is suppressed for one increment when an unexpected missing TRIGGER is detected.</p>
Reserved	[31:24]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

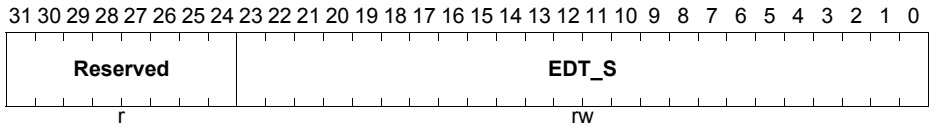
26.16.11.69Memory DPLL_EDT_S

Difference of prediction to actual value of the last STATE increment.

GTM_DPLL_EDT_S

**DPLL Difference of Prediction to actual value
for Last STATE Increment (28488_H)**

Reset Value: 00000000_H



Field	Bits	Type	Description
EDT_S	[23:0]	rw	Signed difference between actual value and prediction of the last STATE increment: sint24 calculated error value
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

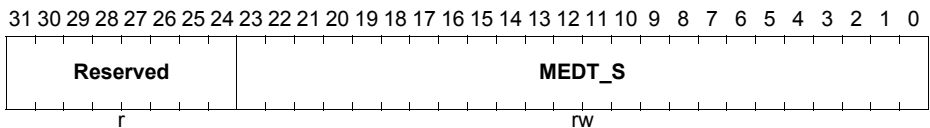
26.16.11.70Memory DPLL_MEDT_S

Weighted difference of prediction errors of STATE.

GTM_DPLL_MEDT_S

**DPLL Weighted difference of Prediction up to the Last STATE Increment
(2848C_H)**

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
MEDT_S	[23:0]	rw	<p>Signed middle weighted difference between actual value and prediction of the last STATE increments: sint24</p> <p>only calculated for SYS=1 calculated medium error value The value is calculated only after synchronization (SYS=1) and the update is suppressed for one increment when an unexpected missing STATE is detected.</p>
Reserved	[31:24]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

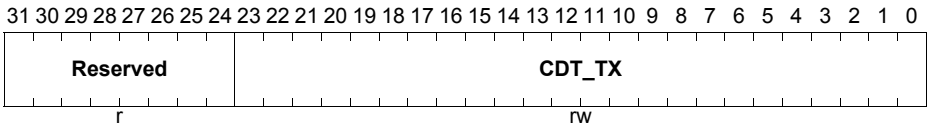
26.16.11.71 Memory DPLL_CDT_TX

Prediction of the actual TRIGGER duration.

GTM_DPLL_CDT_TX

DPLL Prediction of the actual TRIGGER Increment
(28490_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CDT_TX	[23:0]	rw	Calculated duration of the current TRIGGER increment calculated value
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

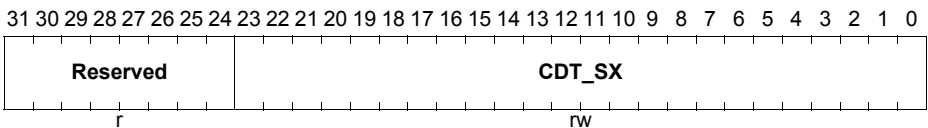
26.16.11.72 Memory DPLL_CDT_SX

Prediction of the actual STATE increment duration.

GTM_DPLL_CDT_SX

DPLL Prediction of the actual STATE Increment
(28494_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CDT_SX	[23:0]	rw	Calculated duration of the current STATE increment calculated value
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.73 Memory DPLL_CDT_TX_NOM

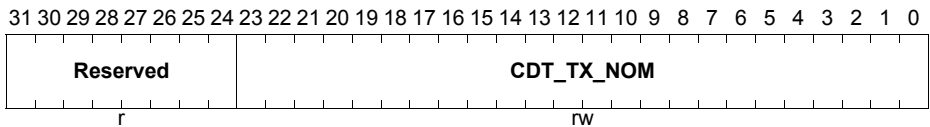
Prediction of the nominal TRIGGER increment duration.

GTM_DPLL_CDT_TX_NOM

DPLL Prediction of the nominal TRIGGER Increment duration

(28498_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CDT_TX_NOM	[23:0]	rw	Calculated duration of the current nominal TRIGGER event calculated value
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.74 Memory DPLL_CDT_SX_NOM

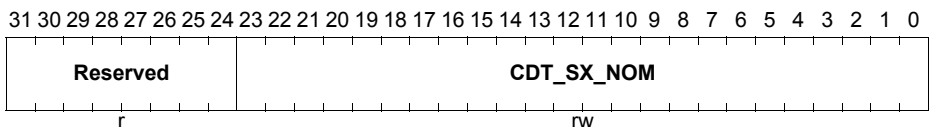
Prediction of the nominal STATE increment duration.

GTM_DPLL_CDT_SX_NOM

DPLL Prediction of the nominal STATE increment duration

(2849C_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CDT_SX_NOM	[23:0]	rw	Calculated duration of the current t nominal STATE event calculated value
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

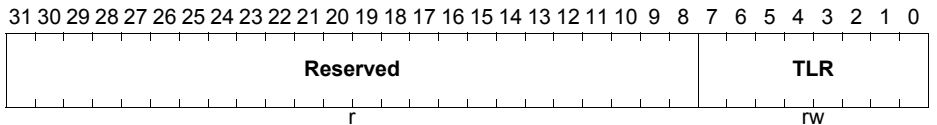
Generic Timer Module (GTM)

26.16.11.75 Memory DPLL_TLR

TRIGGER locking range.

GTM_DPLL_TLR

DPLL TRIGGER locking range (284A0_H) **Reset Value: 00000000_H**



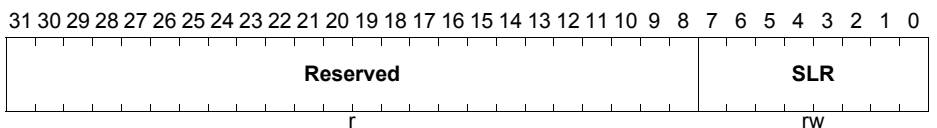
Field	Bits	Type	Description
TLR	[7:0]	rw	Value is to be multiplied with the last nominal TRIGGER duration in order to get the range for the next TRIGGER event without setting TOR in the DPLL_STATUS register multiply value with the last nominal increment duration and check violation; when TLR=0 don't perform the check
Reserved	[31:8]	r	Reserved Read as zero, should be written as zero.

26.16.11.76 Memory DPLL_SLR

STATE locking range.

GTM_DPLL_SLR

DPLL STATE Locking Range (284A4_H) **Reset Value: 00000000_H**



Generic Timer Module (GTM)

Field	Bits	Type	Description
SLR	[7:0]	rw	<p>Value is to be multiplied with the last nominal STATE duration in order to get the range for the next STATE event without setting SOR in the DPLL_STATUS register</p> <p>multiply value with the last nominal increment duration and check violation; when SLR=0 don't perform the check</p>
Reserved	[31:8]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

26.16.11.77 Memory DPLL_PDT[i]

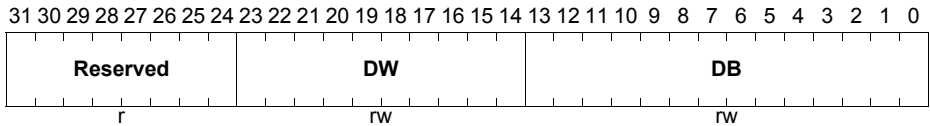
Projected increment sum relations for action i.

GTM_DPLL_PDT_Ti (i=0-23)

DPLL Projected TRIGGER Increment Sum Relations for Action_i

$$(28500_H + i * 04_H)$$

Reset Value: 00000000_H



Field	Bits	Type	Description
DB	[13:0]	rw	Fractional part of relation between TRIGGER or STATE increments
DW	[23:14]	rw	Integer part of relation between TRIGGER or STATE increments Definition of relation values between TRIGGER or STATE increments PDT[i] according to Equation DPLL-11 or DPLL-13.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.78 Memory DPLL_MLS1

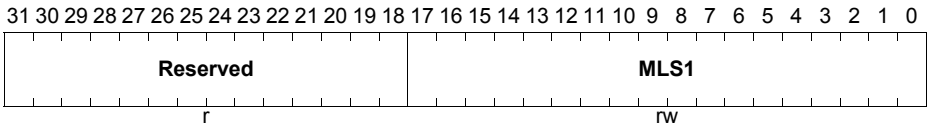
Calculated number of sub-pulses between two STATE events for SMC=0.

GTM_DPLL_MLS1

DPLL Calculated Number of Sub-Pulses between Two STATE Events

(285C0_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
MLS1	[17:0]	rw	<p>Number of pulses between two STATE events (to be set and updated by the CPU)</p> <p>For SMC=0 the value of MLS1 is calculated once by the CPU for fixed values in the DPLL_CTRL_0 register by the formula $MLS1 = ((MLT+1)*(TNU+1)/(SNU+1))$ and set accordingly</p> <p>FOR SMC=1 the value of MLS1 represents the number of pulses between two TRIGGER events (to be set and updated by the CPU)</p>
Reserved	[31:18]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

26.16.11.79 Memory DPLL_MLS2

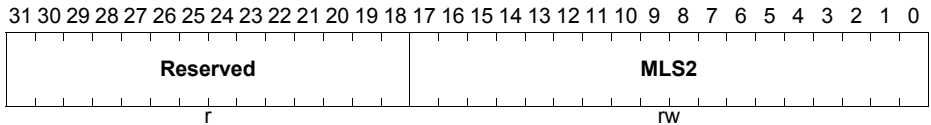
Calculated number of sub-pulses between two STATE events for SMC=1 and RMO=1.

GTM_DPLL_MLS2

DPLL Calculated Number of Sub-Pulses between Two STATE Events

(285C4_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
MLS2	[17:0]	rw	Counter for number of SUB_INC2 pulses Number of pulses between two STATE events (to be set and updated by the CPU). Using adapt information and the missing STATE event information SYN_S, this value can be corrected for each increment automatically.
Reserved	[31:18]	r	Reserved Read as zero, should be written as zero.

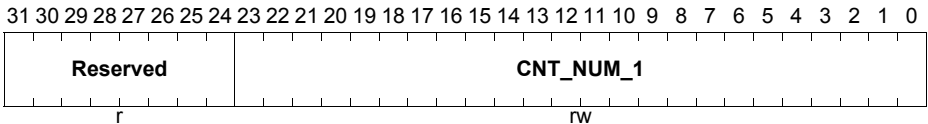
26.16.11.80 Memory DPLL_CNT_NUM1

GTM_DPLL_CNT_NUM1

DPLL Number of Sub-Pulses of SUB_INC1 in Continuous Mode

(285C8_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CNT_NUM_1	[23:0]	rw	Counter for number of SUB_INC1 pulses Number of pulses in continuous mode for a nominal increment in normal and emergency mode for SUB_INC1 given and updated by CPU only count value for continuous mode
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

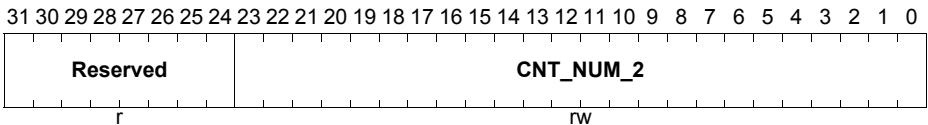
26.16.11.81 Memory DPLL_CNT_NUM2

GTM_DPLL_CNT_NUM2

DPLL Number of Sub-Pulses of SUB_INC2 in Continuous Mode

(285CC_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CNT_NUM_2	[23:0]	rw	Counter for number of SUB_INC2 pulses Number of pulses in continuous mode for a nominal increment in normal and emergency mode for SUB_INC2, given and updated by CPU only count value for continuous mode
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.82 Memory DPLL_PVT

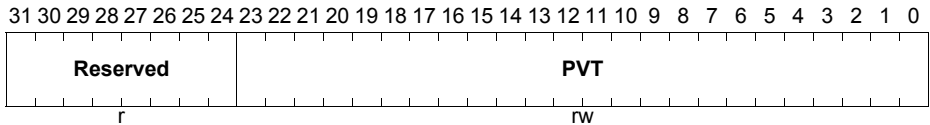
Plausibility value of next TRIGGER slope.

GTM_DPLL_PVT

DPLL Plausibility Value of Next Active TRIGGER Slope

(285D0_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
PVT	[23:0]	rw	<p>Plausibility value of next valid TRIGGER slope</p> <p>The meaning of the value depends on the value of the PIT value in the DPLL_CTRL_1 register.</p> <p>For PIT=0: the number of SUB_INC1 pulses to be waited for until a next valid TRIGGER event is accepted.</p> <p>For PIT=1: PVT is to be multiplied with the last nominal increment time DT_T_ACT and divided by 1024 and reduced to a 24 bit value in order to get the time to be waited for until the next valid TRIGGER event is accepted. The wait time must be exceeded for a valid slope.</p> <p>Note: When a valid TRIGGER slope is detected while the wait condition is not fulfilled the interrupt PWI is generated. Please note, that the SGE1 must be set, when PIT=0 in order to provide the necessary SUB_INC1 pulses for checking.</p> <p>After an unexpected missing TRIGGER the plausibility check is suppressed for the following increment.</p>
Reserved	[31:24]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

26.16.11.83 Memory DPLL_PSTC

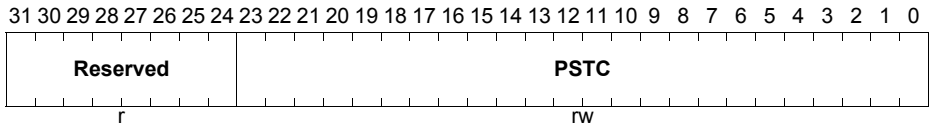
Actual calculated position stamp of TRIGGER.

GTM_DPLL_PSTC

DPLL Actual Calculated Position Stamp of Last TRIGGER Input

(285E0_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
PSTC	[23:0]	rw	<p>Calculated position stamp of last TRIGGER input value is set by the DPLL and can be updated by the CPU when filter values are to be considered for the exact position (see DPLL_STATUS and DPLL_CTRL registers for explanation of the status and control bits used); For each valid slope of TRIGGER in normal and emergency mode</p> <ul style="list-style-type: none"> • when FTD=0: PSTC is set from actual position value, for the first valid TRIGGER event (no filter delay considered) the CPU must update the value once, taking into account the filter value • when FTD=1: PSTC is increment at each TRIGGER event by • SMC=0: (MLT+1)*(SYN_T) +PD; while PD=0 for AMT=0 • SMC=1: (MLS1)*(SYN_T) +PD; while PD=0 for AMT=0
Reserved	[31:24]	r	<p>Reserved Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

26.16.11.84 Memory DPLL_PSSC

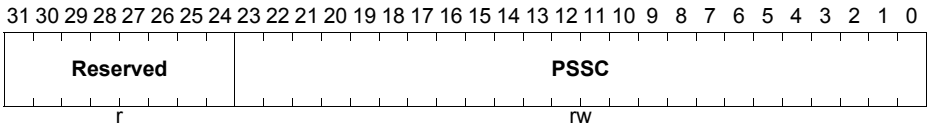
Actual calculated position stamp of STATE.

GTM_DPLL_PSSC

DPLL Accurate Calculated Position Stamp of Last STATE Input

(285E4_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
PSSC	[23:0]	rw	<p>Calculated position stamp for the last STATE input first value is set by the DPLL and can be updated by the CPU when the filter delay is to be considered. For each valid slope of STATE in normal and emergency mode</p> <ul style="list-style-type: none"> when FSD=0: PSSC is set from actual position value (no filter delay considered), the CPU must update the value once, taking into account the filter value when FSD=1: at each valid slope of STATE (PD_S_store=0 for AMS=0): SMC=0: add $MLS1 * (SYN_S) + PD_S_store$; SMC=1: add $MLS2 * (SYN_S) + PD_S_store$;
Reserved	[31:24]	r	<p>Reserved Read as zero, should be written as zero.</p>

Generic Timer Module (GTM)

26.16.11.85 Memory DPLL_PSTM

Measured position stamp at last TRIGGER input.

GTM_DPLL_PSTM_0

DPLL Measured Position Stamp of Last TRIGGER Input

(285E8_H)

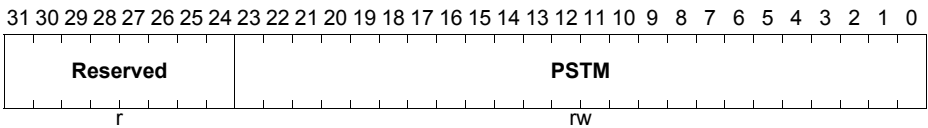
Reset Value: 00000000_H

GTM_DPLL_PSTM_1

DPLL Measured Position Stamp of Last TRIGGER Input

(285EC_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
PSTM	[23:0]	rw	Position stamp of TRIGGER, measured Measured position stamp of last TRIGGER input, measured at processing the input signal. Store the value TBU_TS1 at the moment when T_VALID is set in an intermediate register, but transmit this value to PSTM only, when the plausibility value (according to PVT) is passed.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Note: The LSB address is determined using the SWON_T value in the OSW register (see [Section 26.16.11.7](#)).

Generic Timer Module (GTM)

26.16.11.86 Memory DPLL_PSTM_OLD

Measured position stamp at last but one TRIGGER input.

GTM_DPLL_PSTM_OLD_0

DPLL Measured Position Stamp of Last But One TRIGGER Input

(285EC_H)

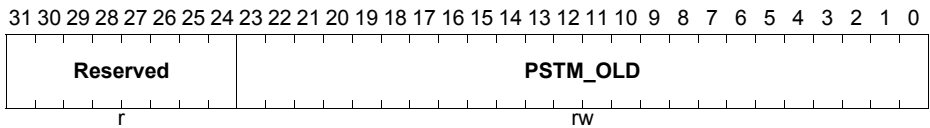
Reset Value: 00000000_H

GTM_DPLL_PSTM_OLD_1

DPLL Measured Position Stamp of Last But One TRIGGER Input

(285E8_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
PSTM_OL D	[23:0]	rw	Last but one position stamp of TRIGGER, measured Measured position stamp of last but one TRIGGER input, measured at processing the input signal. (because of the input and transmission delay the value of PSTM_OLD can be higher than the exact value PSTC_OLD)
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Note: The LSB address is determined using the SWON_T value in the OSW register (see [Section 26.16.11.7](#)).

26.16.11.87 Memory DPLL_PSSM

Measured position stamp at last STATE input.

GTM_DPLL_PSSM_0

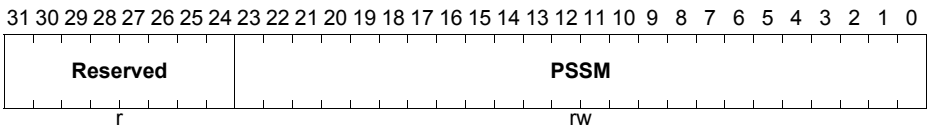
DPLL Measured Position Stamp of Last STATE Input
(285F0_H)

Reset Value: 00000000_H

GTM_DPLL_PSSM_1

DPLL Measured Position Stamp of Last STATE Input
(285F4_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
PSSM	[23:0]	rw	Position stamp of STATE, measured Measured position stamp of last STATE input, measured at processing the input signal. Store the value TBU_TS1 or TBU_TS2 respectively at the moment when S_VALID is set in PSSM.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Note: The LSB address is determined using the SWON_S value in the OSW register (see [Section 26.16.11.7](#)).

Generic Timer Module (GTM)

26.16.11.88 Memory DPLL_PSSM_OLD

Measured position stamp at last but one STATE.

GTM_DPLL_PSSM_OLD_0

DPLL Measured Position Stamp of Last But One STATE Input

(285F4_H)

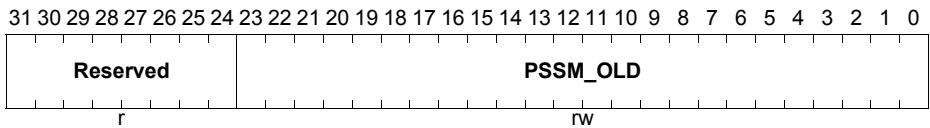
Reset Value: 00000000_H

GTM_DPLL_PSSM_OLD_1

DPLL Measured Position Stamp of Last But One STATE Input

(285F0_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
PSSM_OLD	[23:0]	rw	Last but one position stamp of STATE, measured Measured position stamp of last but one STATE input, measured at processing the input signal. (because of the input and transmission delay the value of PSSM_OLD can be higher than the exact value PSSC_OLD).
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Note: The LSB address is determined using the SWON_S value in the OSW register (see [Section 26.16.11.7](#)).

Generic Timer Module (GTM)

26.16.11.89 Memory DPLL_NMB_T

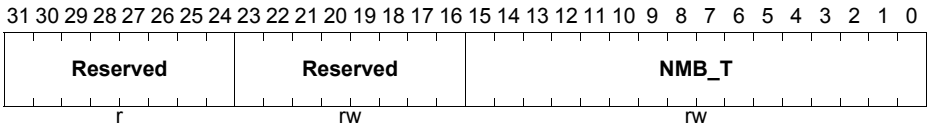
Number of pulses to be sent in normal mode.

GTM_DPLL_NMB_T

DPLL Number of Pulses of Current Increment in Normal Mode

(285F8_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
NMB_T	[15:0]	rw	Number of pulses for TRIGGER Calculated number of pulses in normal mode for the current TRIGGER increment. calculated pulse number
Reserved	[23:16]	rw	Reserved Must be written as zero.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.90 Memory DPLL_NMB_S

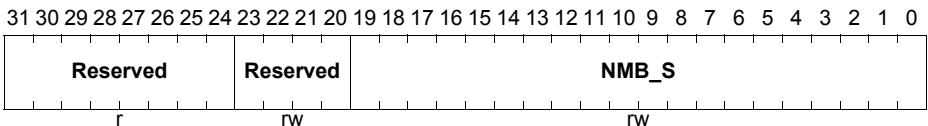
Number of pulses to be sent in emergency mode.

GTM_DPLL_NMB_S

DPLL Number of Pulses of Current Increment in Emergency Mode

(285FC_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

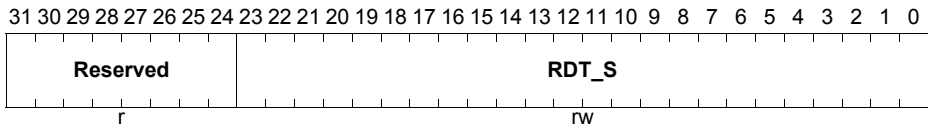
Field	Bits	Type	Description
NMB_S	[19:0]	rw	Number of pulses for STATE Calculated number of pulses in emergency mode for the current STATE increment. calculated pulse number
Reserved	[23:20]	rw	Reserved Must be written as zero.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.91Memory DPLL_RDT_Sx

Reciprocal values of normal STATE increment durations in FULL_SCALE.

GTM_DPLL_RDT_Sx (x=0-63)
DPLL Nominal STATE x Reciprocal Values in FULL_SCALE

$$(28600_H + x * 04_H) \quad \text{Reset Value: } 00000000_H$$



Field	Bits	Type	Description
RDT_S	[23:0]	rw	Reciprocal difference time of TRIGGER Nominal reciprocal value of the number of time stamp clocks measured in the corresponding increment $*2^{32}$ while only the lower 24 bits are used; no gap considered. The LSB is rounded up when the next truncated bit is 1. Note: There are $2*(SNU+1)-SYN_NS$ entries for $SYSF=0$ or $2*(SNU+1)-SYN_NS$ entries for $SYSF=1$ respectively.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.92Memory DPLL_TSF_S[i]

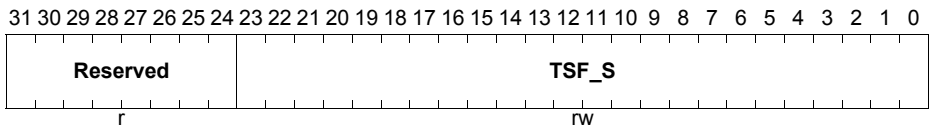
Time stamp of normal STATE events in FULL_SCALE.

GTM_DPLL_TSF_Sx (x=0-63)

DPLL Time Stamp Field of STATE x Events

$$(28700_H + x * 04_H)$$

Reset Value: 00000000_H



Field	Bits	Type	Description
TSF_S	[23:0]	rw	Time stamp field of STATE Time stamp value of each valid STATE event. Note: There are 2* (SNU+1) entries.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.93Memory DPLL_ADT_S[i]

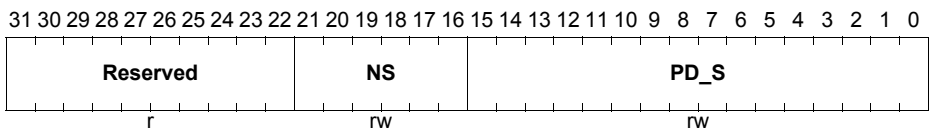
Adapt and profile values of the STATE increments in FULL_SCALE

GTM_DPLL_ADT_Sx (x=0-63)

DPLL Adapt Values for All STATE x Increments

$$(28800_H + x * 04_H)$$

Reset Value: 00000000_H



Field	Bits	Type	Description
PD_S	[15:0]	rw	Physical deviation of STATE Adapt values for each STATE increment in FULL_SCALE (sint16); This value represents the number of pulses to be added to the correspondent increment.

Generic Timer Module (GTM)

Field	Bits	Type	Description
NS	[21:16]	rw	Number of STATES number of nominal STATE parts in the corresponding increment. Note: There are 2* (SNU+1-SYN_NS) entries.
Reserved	[31:22]	r	Reserved Read as zero, should be written as zero.

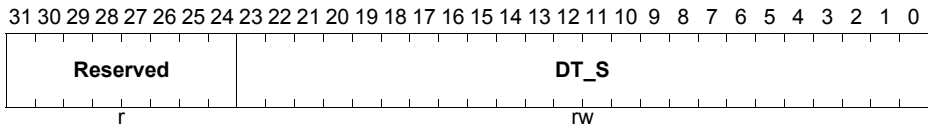
26.16.11.94Memory DPLL_DT_S[i]

Nominal STATE increment durations in FULL_SCALE.

GTM_DPLL_DT_Sx (x=0-63)

DPLL Nominal STATE x Increment Values for FULL_SCALE

$$(28900_H + x * 04_H) \quad \text{Reset Value: } 00000000_H$$



Field	Bits	Type	Description
DT_S	[23:0]	rw	Difference time of STATE nominal increment duration values for each STATE increment in FULL_SCALE (considering no gap). Note: There are 2* (SNU+1-SYN_NS) entries.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

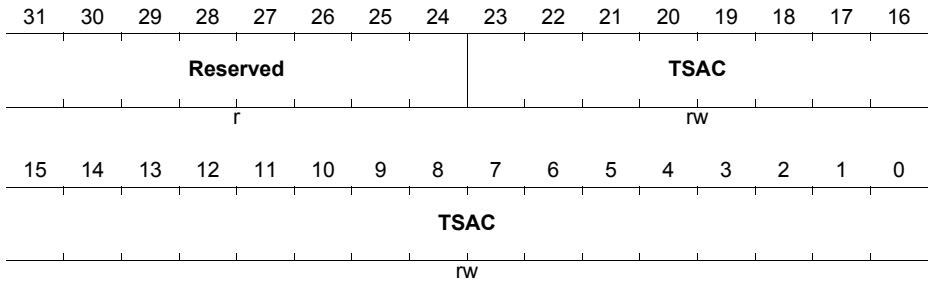
Generic Timer Module (GTM)

26.16.11.95 Memory DPLL_TSAC[i]

Calculated time value to start action i.

GTM_DPLL_TSACi (i=0-23)

DPLL Calculate Time Stamp Register(28E00_H + i * 04_H) Reset Value: 007FFFFF_H



Field	Bits	Type	Description
TSAC	[23:0]	rw	calculated time stamp for ACTION_i Note: This value can only be written when the DPLL is disabled.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.96 Memory DPLL_PSAC[i]

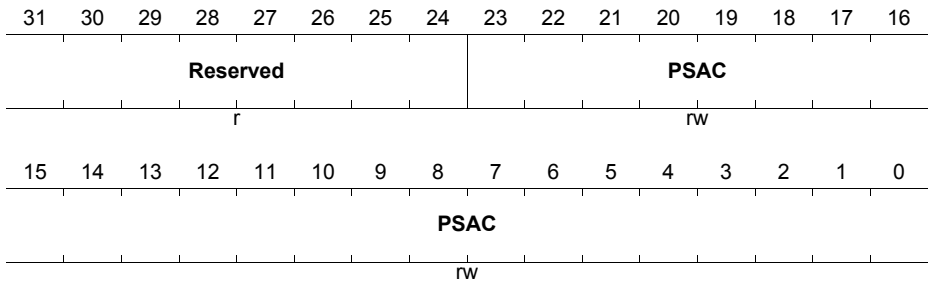
Calculated position to start action i.

GTM_DPLL_PSACi (i=0-23)

DPLL Calculated Position Value Register

$$(28E80_H + i * 04_H)$$

Reset Value: 007FFFFFFH



Field	Bits	Type	Description
PSAC	[23:0]	rw	calculated position value for the start of ACTION_i in normal or emergency mode according to equations DPLL-17 or DPLL-20 respectively Note: This value can only be written when the DPLL is disabled.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.97 Memory DPLL_ACBi

Control bits for actions.

GTM_DPLL_ACBi (i=0-5)
DPLL Action Control i Register (28F00_H + i * 04_H)
Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ACB_3				Reserved			ACB_2					
r			rw				r			rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ACB_1				Reserved			ACB_0					
r			rw				r			rw					

Field	Bits	Type	Description
ACB_0	[4:0]	rw	Action Control Bits of ACTION_j reflects ACT_D[j](52:48), j=4*i Note: This value can only be written when the DPLL is disabled.
Reserved	[7:5]	r	Reserved Read as zero, should be written as zero.
ACB_1	[12:8]	rw	Action Control Bits of ACTION_(j + 1) reflects ACT_D[j+1](52:48), j=4*i Note: This value can only be written when the DPLL is disabled.
Reserved	[15:13]	r	Reserved Read as zero, should be written as zero.
ACB_2	[20:16]	rw	Action Control Bits of ACTION_(j + 2) reflects ACT_D[j+2](52:48), j=4*i Note: This value can only be written when the DPLL is disabled.
Reserved	[23:21]	r	Reserved Read as zero, should be written as zero.
ACB_3	[28:24]	rw	Action Control Bits of ACTION_(j + 3) reflects ACT_D[j+3](52:48), j=4*i Note: This value can only be written when the DPLL is disabled.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:29]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)
26.16.11.98 Memory DPLL_RDT_T[i]

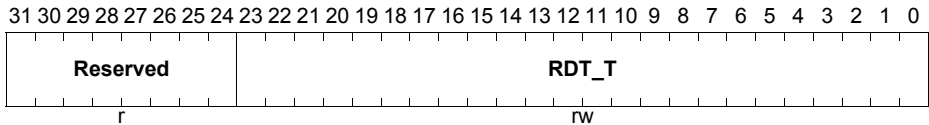
Reciprocal values of the nominal TRIGGER increment durations in FULL_SCALE.

DPLL_RDT_T_x (x=0-511)

DPLL TRIGGER x Nominal Increment Reciprocals in FULL_SCALE

(AOSV_2A + x * 04_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
RDT_T	[23:0]	rw	Reciprocal difference time of TRIGGER 2* (TNU+1- SYN_NT) stored values nominal reciprocal value of the number of time stamp clocks measured in the corresponding increment (which is divided by the number of nominal increments); multiplied by 2 ³² while only the lower 24 bits are used; the LSB is rounded up, when the next truncated bit is 1. Note: There are 2* (TNU+1- SYN_NT) entries. The maximum number of entries is restricted to a value corresponding to the OSS value in the DPLL_OSW register.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

26.16.11.99 Memory DPLL_TSF_T[i]

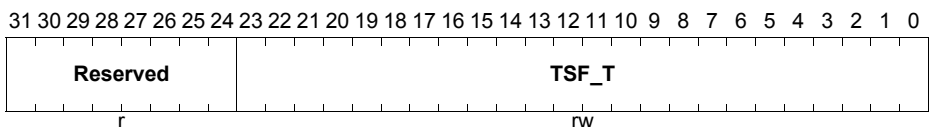
Time stamp values of the nominal TRIGGER increments in FULL_SCALE.

DPLL_TSF_T_x (x=0-511)

DPLL Time Stamp Field of TRIGGER x Events

(AOSV_2B + x * 04_H)

Reset Value: 00000000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
TSF_T	[23:0]	rw	Time stamp field of valid TRIGGER slopes Note: There are 2^* (TNU+1) entries. The maximum number of entries is restricted to a value corresponding to the OSS value in the DPLL_OSW register.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero.

Generic Timer Module (GTM)

26.16.11.100 Memory DPLL_ADT_T[i]

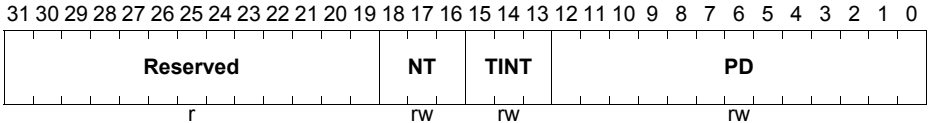
Adapt and profile values of the TRIGGER increments in FULL_SCALE.

DPLL_ADT_Tx (x=0-511)

DPLL Adapt Value x for All Increments

$$(AOSV_2C + x * 04_H)$$

Reset Value: 00000000_H



Field	Bits	Type	Description
PD	[12:0]	rw	Physical deviation Adapt values for each TRIGGER increment in FULL_SCALE (sint13); the PD value does mean the number of SUB_INC1 pulses to be added to NT*(MLT+1); the absolute value of a negative PD must not exceed nt*(MLT+1) or MLS1 respectively; systematic missing TRIGGER events must not be considered for the value of PD;
TINT	[15:13]	rw	TRIGGER Interrupt information depending on the value up to 7 different interrupts can be generated. In the current version the 5 interrupts TE0_IRQ ... TE4_IRQ are supported by TINT="001", "010", "011", "100", "101" respectively. For the values "000", "110" and "111" no interrupt is generated and no other reaction is performed. The corresponding interrupt is activated, when the TINT value is read by the DPLL together with the other values (PD, NT) according to the profile.
NT	[18:16]	rw	Number of TRIGGERS number of nominal TRIGGER parts in the corresponding increment. Note: There are 2* (TNU+1- SYN_NT) entries. The maximum number of entries is restricted to a value corresponding to the OSS value in the DPLL_OSW register.
Reserved	[31:19]	r	Reserved Read as zero, should be written as zero.

26.16.11.101 Memory DPLL_DT_T[i]

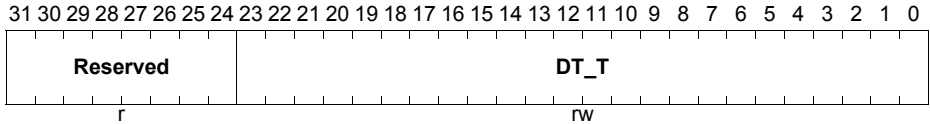
Nominal TRIGGER increment durations in FULL_SCALE.

DPLL_DT_Tx (x=0-511)

DPLL Nominal TRIGGER Increment Values for FULL_SCALE

$$(AOSV_2D + x * 04_H)$$

Reset Value: 00000000_H



Field	Bits	Type	Description
DT_T	[23:0]	rw	<p>Difference time of TRIGGER</p> <p>Increment duration values for each TRIGGER increment in FULL_SCALE divided by the number of nominal increments (nominal value).</p> <p>Note: There are 2*(TNU+1- SYN_NT) entries. The maximum number of entries is restricted to a value corresponding to the OSS value in the DPLL_OSW register.</p>
Reserved	[31:24]	r	<p>Reserved</p> <p>Read as zero, should be written as zero.</p>

26.16.12 Terms and Abbreviations

The following terms and abbreviations are used for DPLL:

- FULL_SCALE; Range in which all positions / values depend on the information of TRIGGER and STATE signals
- HALF_SCALE; Range in which all positions / values depend on the information of TRIGGER signal only, two consecutive HALF_SCALE periods from a FULL_SCALE period
- TS; Time stamp representation
- PS; Position (or value) stamp representation, common description

26.17 Sensor Pattern Evaluation (SPE)

26.17.1 Overview

The Sensor Pattern Evaluation (SPE) submodule can be used to evaluate three hall sensor inputs and together with the TOM module to support the drive of BLDC engines. Thus, the input signals are filtered already in the connected TIM channels. In addition, the SPE submodule can be used as an input stage to the MAP submodule if the DPLL should be used to calculate the rotation speed of one or two electric engine(s). The integration of the SPE submodule into the overall GTM architecture concept is shown in [Figure 26-77](#).

26.17.1.1 SPE Submodule integration concept into GTM

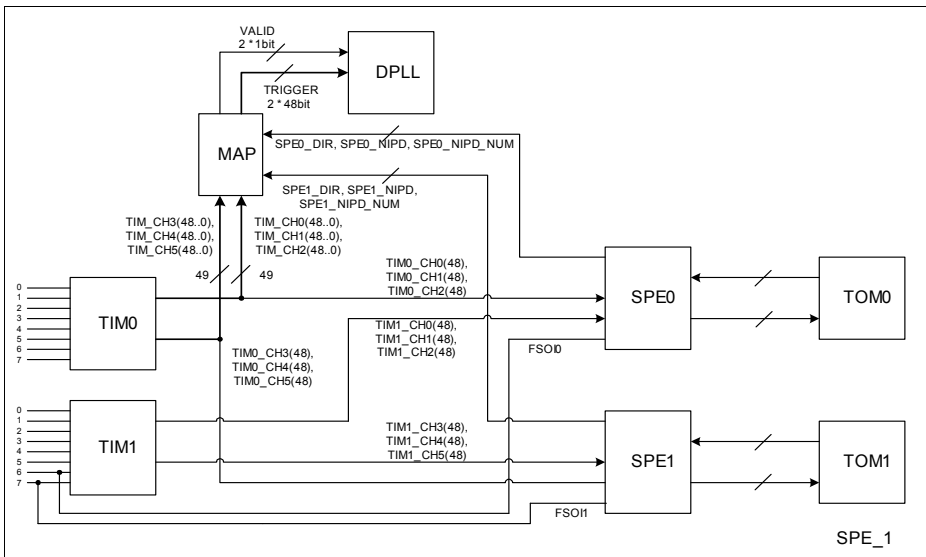


Figure 26-77 SPE integration

As mentioned above, the SPE submodule can determine a rotation direction out of the combined TIM[i]_CHx(48), TIM[i]_CHy(48) and TIM[i]_CHz(48) signals. On this input signals a pattern matching algorithm is applied to generate the SPE_x_DIR signal on behalf of the temporal relation between these input patterns. A possible sample pattern of the three input signals is shown in [Figure 26-78](#). In general, the input pattern is programmable within the SPE submodule.

26.17.1.2 SPE Sample input pattern for

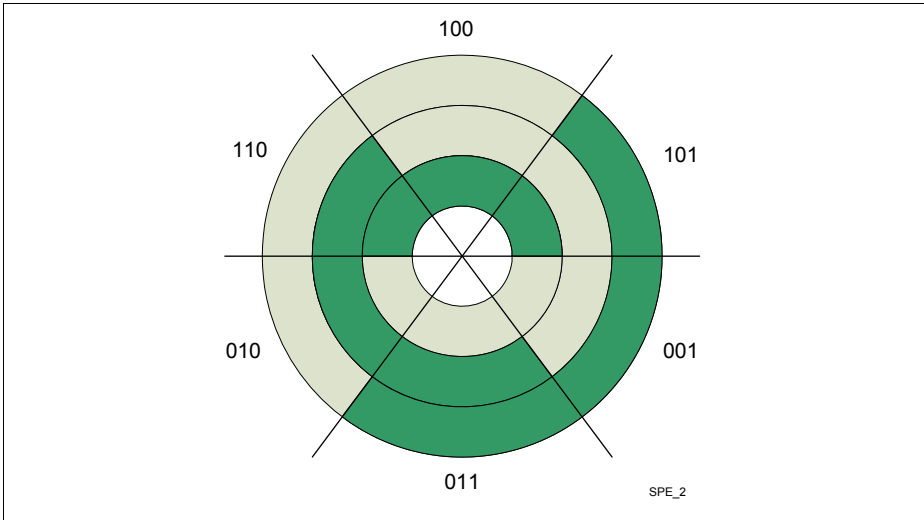


Figure 26-78 SPE sample input pattern

In **Figure 26-78** the input signals define the pattern from the input sensors which have a 50% high and 50% low phase. The pattern according to **Figure 26-79** is as follows:

100 – 110 – 010 – 011 – 001 – 101 – 100

where the first bit (smallest circle) represents $TIM[i]_{CH}[x](48)$, the second bit represents $TIM[i]_{CH}[y](48)$, and the third bit (greatest circle) represents $TIM[i]_{CH}[z](48)$.

Note that the SPE module expects that with every new pattern only one of the three input signals changes its value.

26.17.2 SPE Submodule description

The SPE submodule can handle sensor pattern inputs. Every time if one of the input signals $TIM[i]_{CH}[x](48)$, $TIM[i]_{CH}[y](48)$ or $TIM[i]_{CH}[z](48)$ changes its value, a sample of all three input signals is made. Derived from the sample of the three inputs the encoded rotation direction and the validity of the input pattern sequence can be detected and signalled. When a valid input pattern is detected, the SPE submodule can control the outputs of a dedicated connected TOM submodule. This connection is shown in **Figure 26-79**.

26.17.2.1 SPE to TOM Connections

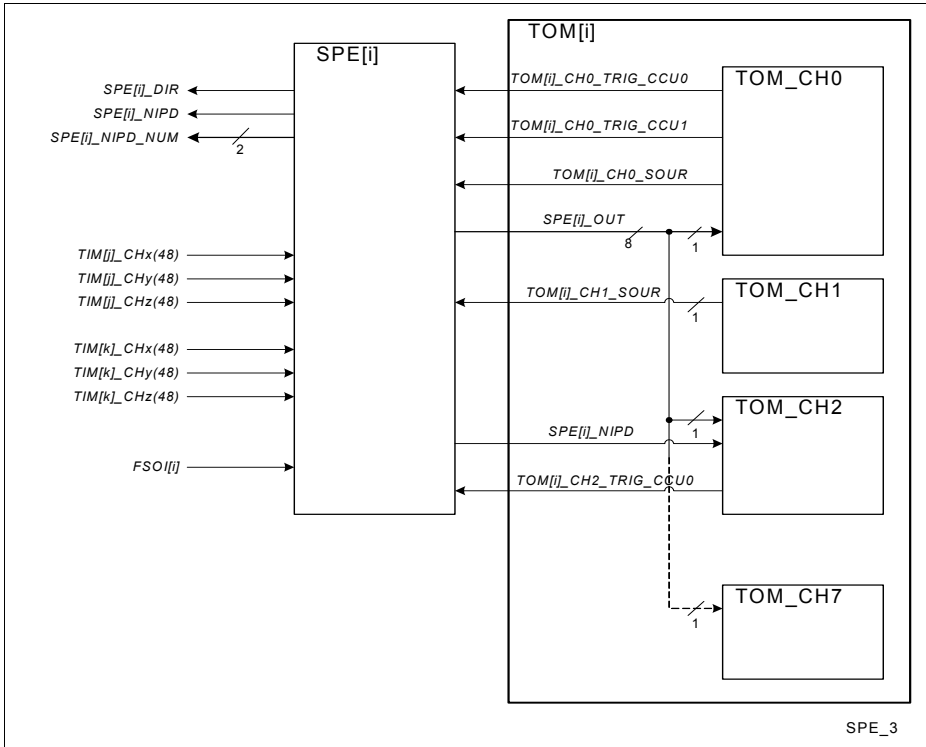


Figure 26-79 SPE to TOM Connections

The TOM[i]_CH0_TRIG_CCU[x] and TOM[i]_CH[x]_SOUR signal lines are used to evaluate the current state of the TOM outputs, whereas the SPE[i]_OUT output vector is used to control the TOM output depending on the new input pattern. The SPE[i]_OUT output vector is defined inside the SPE submodule in a pattern definition table SPEi_OUT_PATx. The internal SPE submodule architecture is shown in [Figure 26-80](#).

26.17.2.2 SPE Submodule architecture

Generic Timer Module (GTM)

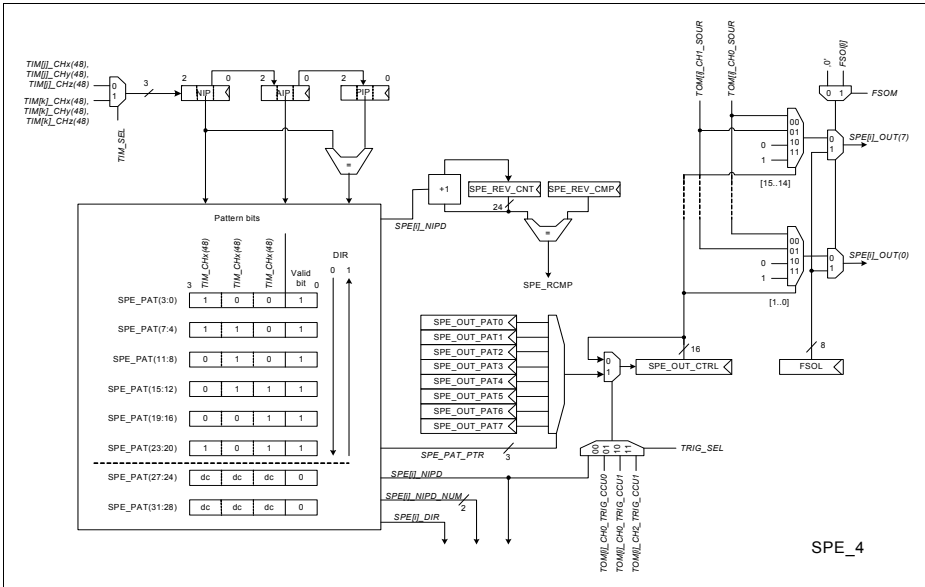


Figure 26-80 SPE Submodule architecture

The SPE_i_PAT register holds the valid input pattern for the three input patterns TIM[i]_CH[x](48), TIM[i]_CH[y](48) and TIM[i]_CH[z](48). The input pattern is programmable. The valid bit shows if the programmed pattern is a valid one. **Figure 26-80** shows the programming of the SPE_i_PAT register for the input pattern defined in **Figure 26-79**.

The rotation direction is determined by the order of the valid input pattern. This rotation direction defines if the SPE_PAT_PTR is incremented (DIR = 0) or decremented (DIR = 1). Whenever a valid input pattern is detected, the NIPD signal is raised, the SPE_PAT_PTR is increment/decremented and a new output control signal SPE_i_OUT(x) is sent to the corresponding TOM submodule.

The TOM[i]_CH2 with i=0...3 can be used together with the SPE module to trigger a delayed update of the SPE_OUT_CTRL register after new input pattern detected by SPE (signalled by SPE_i_NIPD).

To do this, the TOM[i]_CH2 has to be configured to work in one-shot mode (set bit OSM in register TOM[i]_CH2_CTRL). The SPE mode of this channel has to be enabled, too (set bit SPEM in register TOM[i]_CH2_CTRL). The SPE module has to be configured to update SPE_OUT_CTRL on TOM[i]_CH2_TRIG_CCUI (set in SPE_i_CTRL_STAT bits TRIG_SEL to '11'). Then, on new input detected by SPE, the signal SPE_i_NIPD triggers the start of the TOM channel 2 to generates one PWM period by resetting CN0 to 0. On

Generic Timer Module (GTM)

second PWM edge triggered by CCU1 of TOM channel 2, the signal TOM[i]_CH2_TRIG_CCU1 triggers the update of SPE_OUT_CTRL.

According to [Figure 26-80](#), the two input patterns "000" and "111" are not allowed combinations and will end in a SPE[i]_PERR interrupt. These two patterns can be used to determine a sensor input error. A SPE[i]_PERR interrupt will also be raised, if the input patterns occur in a wrong order, e.g. if the pattern "010" does not follow the pattern "110" or "011".

The register SPE[i]_IN_PAT bit field inside the SPEi_CTRL_STAT register is implemented, where the input pattern history is stored by the SPE submodule. The CPU can determine a broken sensor when the SPE[i]_PERR interrupt occurs by analyzing the bit pattern NIP inside the SPEi_CTRL_STAT register. The input pattern in the SPEi_CTRL_STAT register is updated whenever a valid edge is detected on one of the input lines TIM[i]_CH[x](48), TIM[i]_CH[y](48) or TIM[i]_CH[z](48). The pattern bit fields are then shifted. The input pattern history generation inside the SPEi_CTRL_STAT register is shown in [Figure 26-81](#).

Additionally to the sensor pattern evaluation the SPE module also provides the feature of fast shut-off for all TOM channels controlled by the SPE module. The feature is enabled by setting bit FSOM in register SPEi_CTRL_STAT. The fast shut-off level itself is defined in the bit field FSOL of register SPEi_CTRL_STAT. The TIM input used to trigger the fast shut-off is either TIM channel 6 or TIM channel 7 depending on the TIM instance connected to the SPE module. For details of connections please refer to [Figure 26-77](#).

26.17.2.3 SPE[i]_IN_PAT register representation

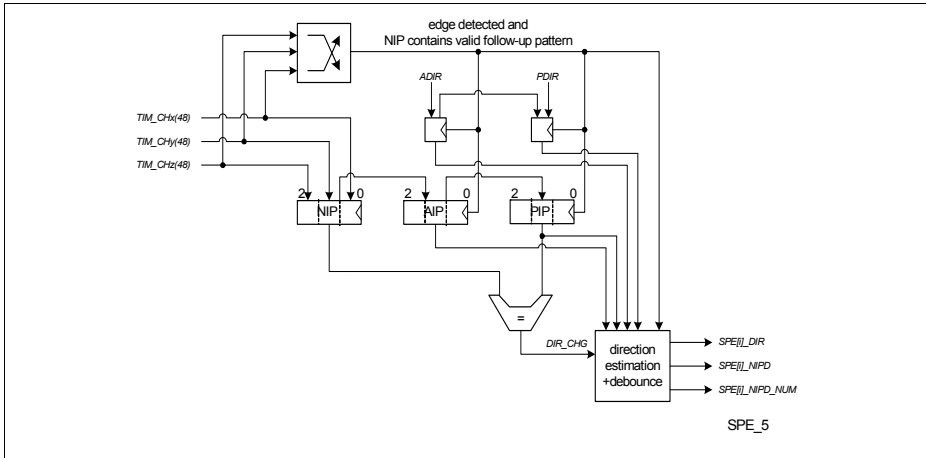


Figure 26-81 Register SPE_IN_PAT implementation

The CPU can disable one of the three input signals, e.g. when a broken input sensor was detected, by disabling the input with the three input enable bits SIE inside the SPE_i_CTRL_STAT register.

Whenever at least one of the input signal TIM_[i]_CH_[x](48), TIM_[i]_CH_[y](48) or TIM_[i]_CH_[z](48) changes the SPE submodule stores the new bit pattern in an internal register NIP (New Input Pattern). If the current input pattern in NIP is the same as in the Previous Input Pattern (PIP) the direction of the engine changed, the SPE_[i]_DCHG interrupt is raised, the direction change is stored internally and the pattern in the PIP bit field is filled with the AIP bit field and the AIP bit field is filled with the NIP bit field. The SPE_[i]_DIR bit inside the SPE_i_CTRL_STAT register is toggled and the SPE_[i]_DIR signal is changed.

If the SPE encounters that with the next input pattern detected new input pattern NIP the direction change again, the input signal is categorized as bouncing and the bouncing input signal interrupt SPE_[i]_BIS is raised.

Immediately after update of register NIP, when the new detected input pattern doesn't match the PIP pattern (i.e. no direction change was detected), the SPE shifts the value of register AIP to register PIP and the value of register NIP to register AIP. The SPE_[i]_NIPD interrupt is raised.

The number of the channel that has been changed and thus leads to the new input pattern is encoded in the signal SPE_[i]_NIPD_NUM.

Generic Timer Module (GTM)

If a sensor error was detected, the CPU has to define upon the pattern in the SPE_i_CTRL_STAT register, which input line comes from the broken sensor. The faulty signal line has to be masked by the CPU and the SPE submodule determines the rotation direction on behalf of the two remaining TIM_[i]_CH_[x] input lines.

The pattern history can be determined by the CPU by reading the two bit fields AIP and PIP of the SPE_i_CTRL_STAT register. The AIP register field holds the actual detected input pattern at TIM_[i]_CH_[x](48), TIM_[i]_CH_[y](48) and TIM_[i]_CH_[z](48) and the PIP holds the previous detected pattern.

After reset the register NIP, AIP and PIP as well as the register SPE_i_PAT_PTR and SPE_i_OUT_CTRL will not contain valid startup values which would allow correct behaviour after enabling SPE and detecting the first input patterns.

Thus, it is necessary to initialize these register to correct values.

To do this, before enabling the SPE, the bit field NIP of register SPE_i_CTRL_STAT can be read and depending on this value the initialization values for the register AIP, PIP, SPT_PAT_PTR and SPE_i_OUT_CTRL can be determined.

26.17.2.4 SPE Revolution detection

The SPE submodule is able to detect and count the number of valid input patterns detected at the specified input ports. This is done with a 24bit revolution counter SPE_CNT. The counter is incremented by a value of one (1) when a new valid input pattern indicating forward direction is detected. The counter is decremented by a value of one (1) when a new valid input pattern indicating backward direction is detected.

In addition there exists a 24 bit SPE_CMP register. The user can initialize this register with a compare value, where an interrupt SPE_i_RCMP is raised, when the revolution counter equals the compare value either in forward or backward direction.

Both register may be written by software at any time.

26.17.3 SPE Interrupt signals

The following table describes SPE interrupt signals:

Table 26-54 SPE Interrupt signals

Signal	Description
SPE _[i] _NIPD	SPE New valid input pattern detected.
SPE _[i] _DCHG	SPE Rotation direction change detected on behalf of input pattern.
SPE _[i] _PERR	SPE Invalid input pattern detected.
SPE _[i] _BIS	SPE Bouncing input signal detected at input.

26.17.4 SPE Register overview

The following table shows an overview about the SPE register set.

Table 26-55 SPE Register overview

Register name	Description	Details in Section
SPEi_CTRL_STAT	SPE Control status register	Section 26.17.5.1
SPEi_PAT	SPE Input pattern definition register	Section 26.17.5.2
SPEi_OUT_PATx	SPE Output definition registers. (x: 07)	Section 26.17.5.3
SPEi_OUT_CTRL	SPE output control register	Section 26.17.5.4
SPEi_CNT	SPE input revolution counter	Section 26.17.5.5
SPEi_CMP	SPE revolution counter compare value	Section 26.17.5.6
SPEi_IRQ_NOTIFY	SPE Interrupt notification register	Section 26.17.5.7
SPEi_IRQ_EN	SPE Interrupt enable register	Section 26.17.5.8
SPEi_EIRQ_EN	SPE Error interrupt enable register.	Section 26.17.5.11
SPEi_IRQ_FORCINT	SPE Interrupt generation by software	Section 26.17.5.9
SPEi_IRQ_MODE	IRQ mode configuration register	Section 26.17.5.10

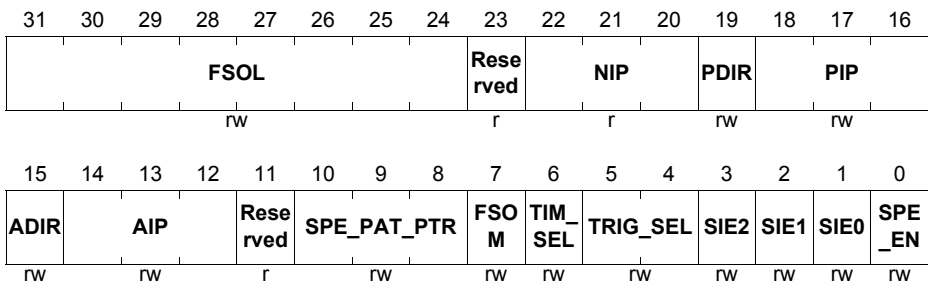
26.17.5 SPE Register description

All of the following registers are 32-bit only accessible.

26.17.5.1 Register SPEi_CTRL_STAT

GTM_SPEi_CTRL_STAT (i=0-1)

SPEi Control Status Register (00800_H+i*80_H) **Reset Value: 00000000_H**



Generic Timer Module (GTM)

Field	Bits	Type	Description
SPE_EN	0	rw	SPE Submodule enable 0 _B SPE disabled. 1 _B SPE enabled.
SIE0	1	rw	SPE Input enable for TIM_CHx(48) 0 _B SPE Input is disabled. 1 _B SPE Input is enabled. Note: When the input is disabled, a '0' signal is sampled for this input. However, the bit field NIP of this register shows the true value of the input signal.
SIE1	2	rw	SPE Input enable for TIM_CHy(48) See bit 1.
SIE2	3	rw	SPE Input enable for TIM_CHz(48) See bit 1.
TRIG_SEL	[5:4]	rw	Select trigger input signal 00 _B SPE[i]_NIPD selected. 01 _B TOM_CH0_TRIG_CCU0 selected. 10 _B TOM_CH0_TRIG_CCU1 selected. 11 _B TOM_CH2_TRIG_CCU1 selected.
TIM_SEL	6	rw	select TIM input signal SPE0: 0 = TIM0_CH0...2 1 = TIM1_CH0...2 SPE1: 0 = TIM0_CH3...5 1 = TIM1_CH3...5 SPE2: 0 = TIM2_CH0...2 1 = unused SPE3: 0 = TIM2_CH3...5 1 = unused
FSOM	7	rw	Fast Shut-Off Mode 0 _B Fast Shut-Off mode disabled 1 _B Fast Shut-Off mode enabled

Generic Timer Module (GTM)

Field	Bits	Type	Description
SPE_PAT_PTR	[10:8]	rw	Pattern selector for TOM output signals Actual index into the SPE _i _OUT_PAT _x register table. Each register SPE _i _OUT_PAT _x is fixed assigned to one bit field IP _x _PAT of register SPE _i _PAT. Thus, the pointer SPE _i _PAT_PTR represents an index to the selected SPE _i _OUT_PAT _x register as well as the actual detected input pattern IP _x _PAT. 000 _B SPE _[i] _OUT_PAT ₀ selected
Reserved	11	r	Reserved Read as zero, should be written as zero
AIP	[14:12]	rw	Actual input pattern that was detected by a regular input pattern change
ADIR	15	rw	Actual rotation direction 0 _B Rotation direction is 0 according to SPE _i _PAT register. 1 _B Rotation direction is 1 according to SPE _i _PAT register.
PIP	[18:16]	rw	Previous input pattern that was detected by a regular input pattern change
PDIR	19	rw	Previous rotation direction 0 _B Rotation direction is 0 according to SPE _i _PAT register. 1 _B Rotation direction is 1 according to SPE _i _PAT register.
NIP	[22:20]	r	New input pattern that was detected Note: This bit field mirrors the new input pattern. SPE internal functionality is triggered on each change of this bit field.
Reserved	23	r	Reserved Read as zero, should be written as zero
FSOL	[31:24]	rw	Fast Shut-Off Level for TOM_[i] channel 0 to 7

26.17.5.2 Register SPE_i_PAT

 GTM_SPE_i_PAT (i=0-1)

 SPE_i Input Pattern Definition Register

 (804_H+i*80_H)

 Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IP7_PAT			IP7_VAL	IP6_PAT			IP6_VAL	IP5_PAT			IP5_VAL	IP4_PAT			IP4_VAL
rw			rw	rw			rw	rw			rw	rw			rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IP3_PAT			IP3_VAL	IP2_PAT			IP2_VAL	IP1_PAT			IP1_VAL	IP0_PAT			IP0_VAL
rw			rw	rw			rw	rw			rw	rw			rw

Field	Bits	Type	Description
IP0_VAL	0	rw	Input pattern 0 is a valid pattern 0 _B Pattern invalid. 1 _B Pattern valid.
IP0_PAT	[3:1]	rw	Input pattern 0 Bit field defines the first input pattern of the SPE input signals. Bit 1 defines the TIM[i]_CHx(48) input signal. Bit 2 defines the TIM[i]_CHy(48) input signal. Bit 3 defines the TIM[i]_CHz(48) input signal.
IP1_VAL	4	rw	Input pattern 1 is a valid pattern See bit 0.
IP1_PAT	[7:5]	rw	Input pattern 1 See bits 3:1.
IP2_VAL	8	rw	Input pattern 2 is a valid pattern See bit 0.
IP2_PAT	[11:9]	rw	Input pattern 2 See bits 3:1.
IP3_VAL	12	rw	Input pattern 3 is a valid pattern See bit 0.
IP3_PAT	[15:13]	rw	Input pattern 3 See bits 3:1.

Generic Timer Module (GTM)

Field	Bits	Type	Description
IP4_VAL	16	rw	Input pattern 4 is a valid pattern See bit 0.
IP4_PAT	[19:17]	rw	Input pattern 4 See bits 3:1.
IP5_VAL	20	rw	Input pattern 5 is a valid pattern See bit 0.
IP5_PAT	[23:21]	rw	Input pattern 5 See bits 3:1.
IP6_VAL	24	rw	Input pattern 6 is a valid pattern See bit 0.
IP6_PAT	[27:25]	rw	Input pattern 6 See bits 3:1.
IP7_VAL	28	rw	Input pattern 7 is a valid pattern See bit 0.
IP7_PAT	[31:29]	rw	Input pattern 7 See bits 3:1.

Note: Only the first block of valid input patterns defines the commutator. All input pattern following the first marked invalid input pattern are ignored.

26.17.5.3 Register SPE_i_OUT_PAT_x (x: 07)

GTM_SPE0_OUT_PAT_x (x=0-7)

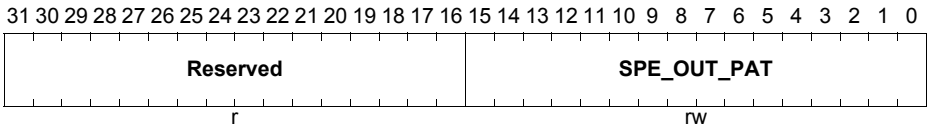
SPE0 Output Definition Register x (808_H+x*04_H)

Reset Value: 00000000_H

GTM_SPE1_OUT_PAT_x (x=0-7)

SPE1 Output Definition Register x (888_H+x*04_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
SPE_OUT_PAT	[15:0]	rw	SPE output control value for TOM_CH0 to TOM_CH7 SPE_OUT_PAT[n+1:n] defines output select signal of TOM[i]_CH[n] 00 _B set SPE_OUT(n) to TOM_CH0_SOUR 01 _B set SPE_OUT(n) to TOM_CH1_SOUR 10 _B set SPE_OUT(n) to '0' 11 _B set SPE_OUT(n) to '1' with n = 0...7
Reserved	[31:16]	r	Reserved Read as zero, should be written as zero

Note: Note: Register SPE_i_OUT_PAT_x defines the output selection for TOM[i]_CH0 to TOM[i]_CH7 depending on actual input pattern IP[x]_PAT with x=0-7.

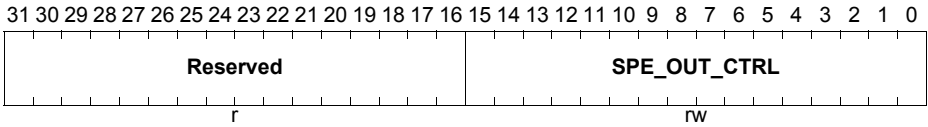
Generic Timer Module (GTM)

26.17.5.4 Register SPE_i_OUT_CTRL

GTM_SPE_i_OUT_CTRL (i=0-1)

SPE_i Output Control Register (828_H+i*80_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
SPE_OUT_CTRL	[15:0]	rw	<p>SPE output control value for TOM_CH0 to TOM_CH7 SPE_OUT_CTRL[n+1:n] defines output select signal of TOM_CHn 00_B set SPE_OUT(n) to TOM_CH0_SOUR 01_B set SPE_OUT(n) to TOM_CH1_SOUR 10_B set SPE_OUT(n) to '0' 11_B set SPE_OUT(n) to '1' with n = 0...7</p>
Reserved	[31:16]	r	<p>Read as zero, should be written as zero Note: Current output control selection for SPE[i]_OUT(0...7).</p>

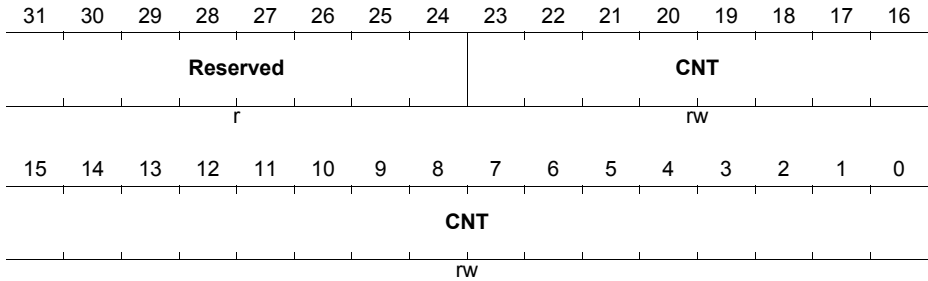
Generic Timer Module (GTM)

26.17.5.5 Register SPEi_CNT

GTM_SPEi_CNT (i=0-1)

SPEi Revolution Counter Register (840_H+i*80_H)

Reset Value: 00000000_H



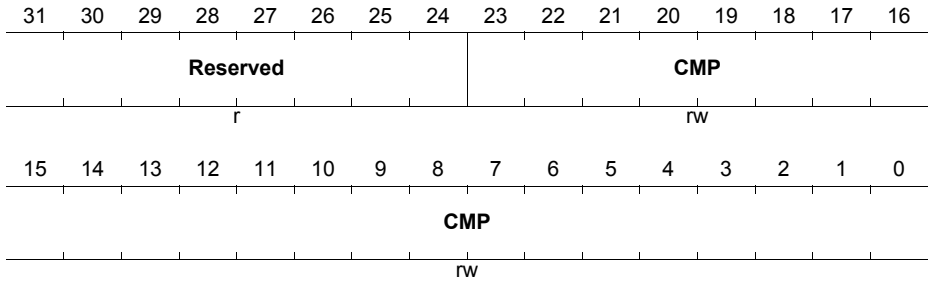
Field	Bits	Type	Description
CNT	[23:0]	rw	Input signal revolution counter The counter is running if SPE module is enabled (bit SPE_EN). CNT is incrementing if SPE_PAT_PTR is incrementing CNT is decrementing if SPE_PAT_PTR is decrementing
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

26.17.5.6 Register SPEi_CMP

GTM_SPEi_CMP (i=0-1)

SPEi Revolution Compare Register(844_H+i*80_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
CMP	[23:0]	rw	Input signal revolution counter compare value The interrupt SPEi_RCMP is raised when the SPEi_CNT value equals the SPEi_CMP register. It should be noted that SPEi_RCMP is only raised if an incrementation or decrementation of SPEi_CNT is applied, due to a input signal change. Any update of SPEi_CNT or SPEi_CMP via AEI does not raise an SPEi_RCMP interrupt.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

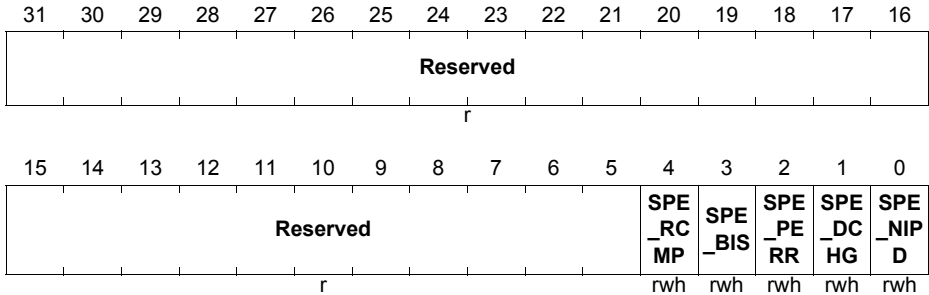
Generic Timer Module (GTM)

26.17.5.7 Register SPE_i_IRQ_NOTIFY

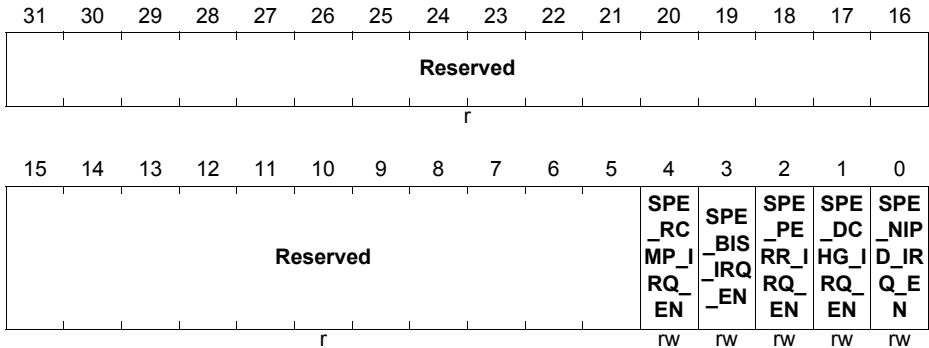
GTM_SPE_i_IRQ_NOTIFY (i=0-1)

SPE_i Interrupt Notification Register(82C_H+i*80_H)

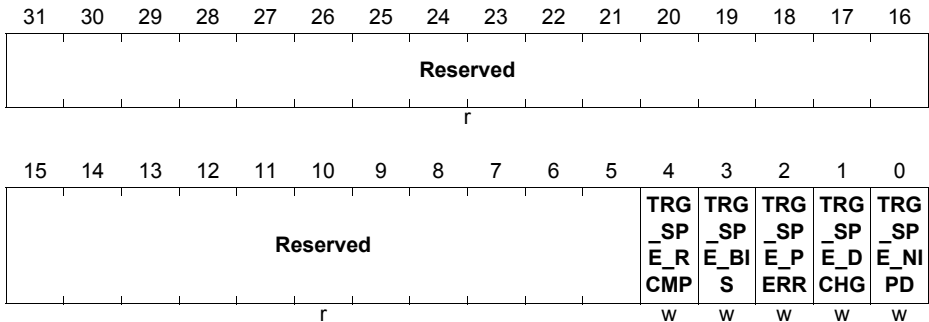
Reset Value: 00000000_H



Field	Bits	Type	Description
SPE_NIPD	0	rwh	New input pattern interrupt occurred 0 _B No interrupt occurred. 1 _B New input pattern detected interrupt occurred. Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
SPE_DCHG	1	rwh	SPE_DIR bit changed on behalf of new input pattern See bit 0.
SPE_PERR	2	rwh	Wrong or invalid pattern detected at input See bit 0.
SPE_BIS	3	rwh	Bouncing input signal detected See bit 0.
SPE_RCM_P	4	rwh	SPE revolution counter match event See bit 0.
Reserved	[31:5]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)
26.17.5.8 Register SPE_i_IRQ_EN
GTM_SPE_i_IRQ_EN (i=0-1)
SPE_i Interrupt Enable Register (830_H+i*80_H)
Reset Value: 00000000_H


Field	Bits	Type	Description
SPE_NIPD_IRQ_EN	0	rw	SPE_NIPD_IRQ interrupt enable 0 _B Disable interrupt, interrupt is not visible outside GTM. 1 _B Enable interrupt, interrupt is visible outside GTM.
SPE_DCHG_IRQ_EN	1	rw	SPE_DCHG_IRQ interrupt enable See bit 0.
SPE_PERR_IRQ_EN	2	rw	SPE_PERR_IRQ interrupt enable See bit 0.
SPE_BIS_IRQ_EN	3	rw	SPE_BIS_IRQ interrupt enable See bit 0.
SPE_RCOMP_IRQ_EN	4	rw	SPE_RCOMP_IRQ interrupt enable See bit 0.
Reserved	[31:5]	r	Reserved Read as zero, should be written as zero

26.17.5.9 Register SPE_i_IRQ_FORCINT
GTM_SPE_i_IRQ_FORCINT (i=0-1)
SPE_i Interrupt Generation by Software
(834_H+i*80_H)
Reset Value: 00000000_H


Field	Bits	Type	Description
TRG_SPE_NIPD	0	w	Force interrupt of SPE_NIPD 0 _B Corresponding bit in status register will not be forced. 1 _B Assert corresponding field in SPE_IRQ_NOTIFY register. Note: This bit is cleared automatically after interrupt is released Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
TRG_SPE_DCHG	1	w	Force interrupt of SPE_DCHG See bit 0.
TRG_SPE_PERR	2	w	Force interrupt of SPE_PERR See bit 0.
TRG_SPE_BIS	3	w	Force interrupt of SPE_BIS See bit 0.
TRG_SPE_RCMP	4	w	Force interrupt of SPE_RCMP See bit 0.
Reserved	[31:5]	r	Reserved Read as zero, should be written as zero

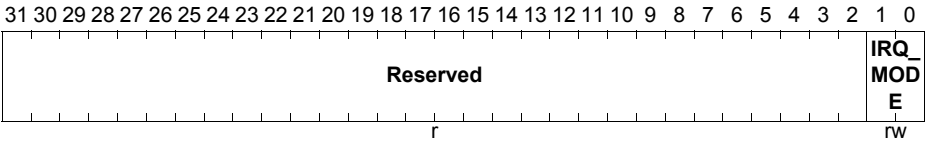
Generic Timer Module (GTM)

26.17.5.10 Register SPEi_IRQ_MODE

GTM_SPEi_IRQ_MODE (i=0-1)

SPEi IRQ Mode Configuration Register(838_H+i*80_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
IRQ_MOD E	[1:0]	rw	IRQ mode selection 00 _B Level mode 01 _B Pulse mode 10 _B Pulse-Notify mode 11 _B Single-Pulse mode Note: The interrupt modes are described in Section 26.2.5 .
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

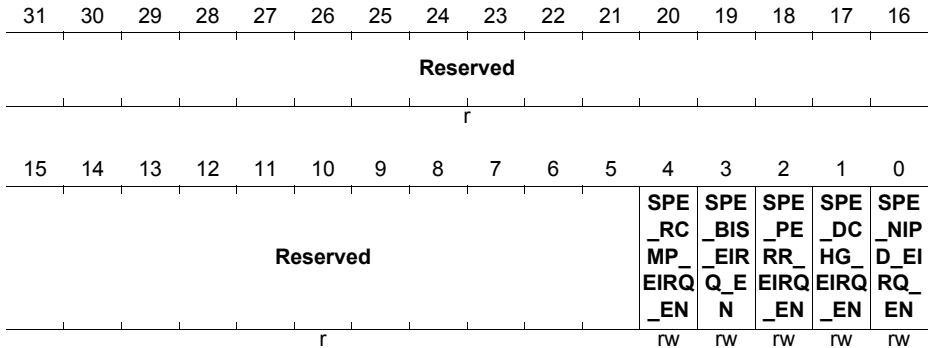
Generic Timer Module (GTM)

26.17.5.11 Register SPE_i_EIRQ_EN

GTM_SPE_i_EIRQ_EN (i=0-1)

SPE_i Error Interrupt Enable Register(83C_H+i*80_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
SPE_NIPD_EIRQ_EN	0	rw	SPE_NIPD_EIRQ interrupt enable 0 _B Disable error interrupt, error interrupt is not visible outside GTM 1 _B Enable error interrupt, error interrupt is visible outside GTM
SPE_DCHG_EIRQ_EN	1	rw	SPE_DCHG_EIRQ error interrupt enable See bit 0.
SPE_PERR_EIRQ_EN	2	rw	SPE_PERR_EIRQ error interrupt enable See bit 0.
SPE_BIS_EIRQ_EN	3	rw	SPE_BIS_EIRQ error interrupt enable See bit 0.
SPE_RCMP_EIRQ_EN	4	rw	SPE_RCMP_EIRQ error interrupt enable See bit 0.
Reserved	[31:5]	r	Reserved Read as zero, should be written as zero

26.18 Interrupt Concentrator Module (ICM)

26.18.1 Overview

The Interrupt Concentrator Module (ICM) is used to bundle the GTM interrupt lines of the individual submodules in a reasonable manner into interrupt groups. By this bundling a smaller amount of interrupt lines is visible at the outside of the GTM.

The individual interrupts of the GTM submodules and channels have to be enabled or disabled inside the submodules and channels.

The feed through architecture of bundled interrupt lines is used for the submodules AEI, ARU, BRC, CMP, SPE, PSM, TIM, DPLL, TOM, ATOM and MCS.

To determine the detailed interrupt source the microcontroller has to read the submodule/channel interrupt notification register NOTIFY and serve the channel individual interrupt.

Please note, that the interrupts are only visible inside the ICM and in consequence outside of the GTM, when the interrupt is enabled inside the submodules themselves.

26.18.2 Bundling

The GTM submodule individual interrupt sources are connected to the ICM. There, the individual interrupt lines are either feed through and signalled to the outside world or bundled a second time into groups and are then signalled to the outside world.

The ICM interrupt bundling is described in the following sections.

26.18.2.1 GTM Infrastructure Interrupt Bundling

The first interrupt group contains interrupts of the infrastructure and safety components of the GTM. This interrupt group includes therefore interrupt lines coming from the AEI, ARU, BRC, PSM, SPE and CMP submodules. In this interrupt group each individual channel of the submodules has its own interrupt line to the outside world.

Thus, the active interrupt line can be used by the CPU to determine the GTM submodule channel that raised the interrupt. The interrupts are also represented in the ICM_IRQG_0 register. This register is typically not read by the CPU, but it is readable.

26.18.2.2 DPLL Interrupt Bundling

The DPLL Interrupt group handles the interrupts coming from the DPLL submodule of the GTM. Each of the individual DPLL interrupt lines has its own dedicated interrupt line to the outside world. The interrupts are additionally identified in the ICM_IRQG_1 interrupt group register. This register is typically not read out by the CPU, but it is readable.

Generic Timer Module (GTM)**26.18.2.3 TIM Interrupt Bundling**

Inside this group submodules which handle GTM input signals are treated. This is the case for the TIM[i] submodules. Each TIM submodule channel is able to generate six (6) individual interrupts if enabled inside the TIM channel. This six interrupts are bundled into one interrupt per TIM channel connected to the ICM.

The ICM does no further bundling. Thus, for the GTM 24 interrupt lines TIM[i]_IRQ[y] are provided for the external microcontroller. The channel responsible for the interrupt can be determined by the raised interrupt line.

In addition, the ICM_IRQG_2 register a mirror for the TIM submodule channel interrupts and typically not read out by the CPU, but it is readable.

26.18.2.4 MCS Interrupt Bundling

For complex signal output generation, the MCS submodules are used inside the GTM. Each of these MCS submodules has eight channels with one interrupt line. This interrupt line is connected to the ICM submodule and is feed through directly to the outside world.

In addition the interrupt line status is shown in the ICM_IRQG_4 register. Typically, the interrupt source is determined by the corresponding interrupt line and the ICM_IRQ4 register is typically not read out by the CPU, but it is readable.

26.18.2.5 TOM and ATOM Interrupt Bundling

For the TOM and ATOM submodules, the interrupts are bundled within the ICM submodule a second time to reduce external interrupt lines. The interrupts are OR-ed in a manner that one GTM external interrupt line represents two adjacent TOM or ATOM channel interrupts. For TOM[i] and ATOM[i] the bundling is shown in **TOM and ATOM interrupt bundling within ICM**.

TOM and ATOM interrupt bundling within ICM

TOM[i]-input IRQs [i]=0..number of TOM's-1	TOM-output IRQs (OR-ed)	ATOM[i]-input IRQs [i]=0..number of ATOM's-1	ATOM-output IRQs (OR-ed)
TOM[i]_CH0_IRQ	GTM_TOM[i]_IRQ[0]	ATOM[i]_CH0_IRQ	GTM_ATOM[i]_IR Q[0]
TOM[i]_CH1_IRQ		ATOM[i]_CH1_IRQ	
TOM[i]_CH2_IRQ	GTM_TOM[i]_IRQ[1]	ATOM[i]_CH2_IRQ	GTM_ATOM[i]_IR Q[1]
TOM[i]_CH3_IRQ		ATOM[i]_CH3_IRQ	
TOM[i]_CH4_IRQ	GTM_TOM[i]_IRQ[2]	ATOM[i]_CH4_IRQ	GTM_ATOM[i]_IR Q[2]
TOM[i]_CH5_IRQ		ATOM[i]_CH5_IRQ	
TOM[i]_CH6_IRQ	GTM_TOM[i]_IRQ[3]	ATOM[i]_CH6_IRQ	GTM_ATOM[i]_IR Q[3]
TOM[i]_CH7_IRQ		ATOM[i]_CH7_IRQ	
TOM[i]_CH8_IRQ	GTM_TOM[i]_IRQ[4]		
TOM[i]_CH9_IRQ			
TOM[i]_CH10_IRQ	GTM_TOM[i]_IRQ[5]		
TOM[i]_CH11_IRQ			
TOM[i]_CH12_IRQ	GTM_TOM[i]_IRQ[6]		
TOM[i]_CH13_IRQ			
TOM[i]_CH14_IRQ	GTM_TOM[i]_IRQ[7]		
TOM[i]_CH15_IRQ			

ICM_1

Figure 26-82 TOM and ATOM interrupts

The interrupts coming from the TOM[i] submodules are registered in the ICM_IRQG_6 register. Always two TOM's are bundled in one ICM register, TOM0 and TOM1 are bundled in ICM_IRQG_6. To identify the TOM submodule channel where the interrupt occurred, the CPU has to read out the ICM_IRQG_6 register first before it goes to the TOM submodule channel itself.

The ICM_IRQG_6 register bits are cleared automatically, when their corresponding interrupt in the submodule channels is cleared.

The interrupts coming from the ATOM[i] submodules are registered in the ICM_IRQG_9 register. Always four ATOM's are bundled in one ICM register, ATOM0, ATOM1, ATOM2 and ATOM3 are bundled in ICM_IRQG_9. To identify the ATOM submodule channel where the interrupt occurred, the CPU has to read out the ICM_IRQG_9 register first before it goes to the ATOM submodule channel itself.

The interrupts coming from the ATOM[i] submodules are registered in the ICM_IRQG_9 register. ATOM0, ATOM1 and ATOM2 are bundled in ICM_IRQG_9. To identify the

Generic Timer Module (GTM)

ATOM submodule channel where the interrupt occurred, the CPU has to read out the ICM_IRQG_9 register first before it goes to the ATOM submodule channel itself.

The ICM_IRQG_9 register bits are cleared automatically, when their corresponding interrupt in the submodule channels is cleared.

Module Error Interrupt Bundling

The Module Error Interrupt group handles the error interrupts coming from the BRC, FIFO, TIM, MCS, SPE, CMP, DPLL submodules of the GTM. The Module Error interrupts are additionally identified in the ICM_IRQ_MEI error interrupt group register. This register is typically not read out by the CPU, but it is readable.

The ICM_IRQG_MEI register bits are cleared automatically, when their corresponding error interrupt in the submodule is cleared.

FIFO Channel Error Interrupt Bundling

The FIFO Channel Error Interrupt group handles the error interrupts coming from the FIFO channel of the GTM. The FIFO Channel Error interrupts are additionally identified in the ICM_IRQ_CEI0 error interrupt group register. This register is typically not read out by the CPU, but it is readable.

The ICM_IRQG_CEI0 register bits are cleared automatically, when their corresponding error interrupt in the submodule channel is cleared.

TIM Channel Error Interrupt Bundling

The TIM Channel Error Interrupt group handles the error interrupts coming from the TIM channel of the GTM. The TIM Channel Error interrupts are additionally identified for the submodules TIM0, TIM1, TIM2, and TIM3 in the ICM_IRQ_CEI1 error interrupt group register and for TIM4, TIM5, and TIM6 in the ICM_IRQ_CEI2 error interrupt group register. This register is typically not read out by the CPU, but it is readable.

The ICM_IRQG_CEI1 and ICM_IRQG_CEI2 register bits are cleared automatically, when their corresponding error interrupt in the submodule channel is cleared.

MCS Channel Error Interrupt Bundling

The MCS Channel Error Interrupt group handles the error interrupts coming from the MCS channel of the GTM. The MCS Channel Error interrupts are additionally identified for the submodules MCS0, MCS1, MCS2, and MCS3 in the ICM_IRQ_CEI3 error interrupt group register and for MCS4, MCS5, and MCS6 in the ICM_IRQ_CEI4 error interrupt group register. This register is typically not read out by the CPU, but it is readable.

The ICM_IRQG_CEI4 and ICM_IRQG_CEI4 register bits are cleared automatically, when their corresponding error interrupt in the submodule channel is cleared.

26.18.3 ICM Interrupt Signals

Following table shows the GTM interrupt lines that are visible at the outside of the GTM.

Table 26-56 ICM Interrupt Signals

Signal	Description
GTM_AEI_IRQ	AEI Shared interrupt
GTM_ARU_IRQ[2:0]	[0]: ARU_NEW_DATA0 Interrupt [1]: ARU_NEW_DATA1 Interrupt [2]: ARU_ACC_ACK Interrupt
GTM_BRC_IRQ	BRC Shared interrupt
GTM_CMP_IRQ	CMP Shared interrupt
GTM_SPE[i]_IRQ	SPE Shared interrupt (i: 0...number of SPE's-1)
GTM_PSM[i]_IRQ[x]	PSM Shared interrupts (x: 0...7) (i: 0...number of PSM's-1)
GTM_DPLL_IRQ[0]	DPLL_DCGI: DPLL TRIGGER direction change detected
GTM_DPLL_IRQ[1]	DPLL_EDI: DPLL enable/disable interrupt
GTM_DPLL_IRQ[2]	DPLL_TIN: DPLL TRIG. min. hold time (THMI) viol. detected
GTM_DPLL_IRQ[3]	DPLL_TAX: DPLL TRIG. max. hold time (THMA) viol. detected
GTM_DPLL_IRQ[4]	DPLL_SIS: DPLL STATE inactive slope detected
GTM_DPLL_IRQ[5]	DPLL_TIS: DPLL TRIGGER inactive slope detected
GTM_DPLL_IRQ[6]	DPLL_MSI: DPLL Missing STATE interrupt
GTM_DPLL_IRQ[7]	DPLL_MTI: DPLL Missing TRIGGER interrupt
GTM_DPLL_IRQ[8]	DPLL_SAS: DPLL STATE active slope detected
GTM_DPLL_IRQ[9]	DPLL_TAS: DPLL TRIG. active slope det. while NTI_CNT is 0
GTM_DPLL_IRQ[10]	DPLL_PWI: DPLL Plausibility window (PVT) viol. int. of TRIG
GTM_DPLL_IRQ[11]	DPLL_W2I: DPLL Write access to RAM region 2 interrupt
GTM_DPLL_IRQ[12]	DPLL_W1I: DPLL Write access to RAM region 1B or 1C int
GTM_DPLL_IRQ[13]	DPLL_GLI: DPLL Get of lock interrupt for SUB_INC1
GTM_DPLL_IRQ[14]	DPLL_LLI: DPLL Lost of lock interrupt for SUB_INC1
GTM_DPLL_IRQ[15]	DPLL_EI: DPLL Error interrupt
GTM_DPLL_IRQ[16]	DPLL_GL2I: DPLL Get of lock interrupt for SUB_INC1
GTM_DPLL_IRQ[17]	DPLL_LL2I: DPLL Lost of lock interrupt for SUB_INC2
GTM_DPLL_IRQ[18]	DPLL_TE0: DPLL TRIGGER event interrupt 0

Generic Timer Module (GTM)
Table 26-56 ICM Interrupt Signals (cont'd)

Signal	Description
GTM_DPLL_IRQ[19]	DPLL_TE0: DPLL TRIGGER event interrupt 1
GTM_DPLL_IRQ[20]	DPLL_TE0: DPLL TRIGGER event interrupt 2
GTM_DPLL_IRQ[21]	DPLL_TE0: DPLL TRIGGER event interrupt 3
GTM_DPLL_IRQ[22]	DPLL_TE0: DPLL TRIGGER event interrupt 4
GTM_DPLL_IRQ[23]	DPLL_CDIT: DPLL calculated duration interrupt for trigger
GTM_DPLL_IRQ[24]	DPLL_CDIS: DPLL calculated duration interrupt for state
GTM_DPLL_IRQ[25]	DPLL_TORI: trigger out of range interrupt
GTM_DPLL_IRQ[26]	DPLL_SORI: status out of range interrupt
GTM_TIM[i]_IRQ[x]	TIM Shared interrupts (i: 0...number of TIM's-1) (x: 0...7)
GTM_MCS[i]_IRQ[x]	MCS Interrupt for channel x (x: 0...7) (i: 0...number of MCS's-1)
GTM_TOM[i]_IRQ[x]	TOM Shared interrupts for x:0...7 = {ch0 ch1,,ch14 ch15} (i: 0...number of TOM's-1)
GTM_ATOM[i]_IRQ[x]	ATOM Shared interrupts for x:0...3 = {ch0 ch1,,ch6 ch7} (i: 0...number of ATOM's-1)
GTM_ERR_IRQ	GTM Error Interrupt

26.18.4 ICM Configuration Registers Overview

ICM contains following configuration registers:

Table 26-57 ICM Configuration Registers Overview

Register Name	Description	Details in Section
ICM_IRQG_0	ICM Interrupt group register covering infrastructural and safety components (ARU, BRC, AEI, PSM[i], MAP, CMP,SPE)	Section 26.18.5.1
ICM_IRQG_1	ICM Interrupt group register covering DPLL	Section 26.18.5.2
ICM_IRQG_2	ICM Interrupt group register covering TIM0, TIM1, TIM2	Section 26.18.5.3
ICM_IRQG_4	ICM Interrupt group register covering MCS0 to MCS3 submodules	Section 26.18.5.4
ICM_IRQG_6	ICM Interrupt group register covering GTM output submodules TOM0 and TOM1	Section 26.18.5.5
ICM_IRQG_9	ICM Interrupt group register covering GTM output submodules ATOM0, ATOM1, ATOM2 and ATOM3	Section 26.18.5.6
ICM_IRQG_MEI	ICM Interrupt group register for module error interrupt information	Section 26.18.5.7
ICM_IRQG_CEI0	ICM Interrupt group register 0 for channel error interrupt information	Section 26.18.5.8
ICM_IRQG_CEI1	ICM Interrupt group register 1 for channel error interrupt information	Section 26.18.5.9
ICM_IRQG_CEI3	ICM Interrupt group register 3 for channel error interrupt information	Section 26.18.5.10

26.18.5 ICM Configuration Registers Description

All of the following registers are 32-bit only accessible.

26.18.5.1 Register ICM_IRQG_0

GTM_ICM_IRQG_0

GTM Infrastructure Interrupt Group (600_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								PSM 0_C H7_I RQ	PSM 0_C H6_I RQ	PSM 0_C H5_I RQ	PSM 0_C H4_I RQ	PSM 0_C H3_I RQ	PSM 0_C H2_I RQ	PSM 0_C H1_I RQ	PSM 0_C H0_I RQ
r								rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								SPE 1_IR Q	SPE 0_IR Q	CMP _IRQ	AEL _IRQ	BRC _IRQ	ARU _AC _C_A CK_I RQ	ARU _NE _W_D ATA 1_IR	ARU _NE _W_D ATA 0_IR
r								rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ARU_NEW_DATA0_I RQ	0	rh	ARU_NEW_DATA0 interrupt 0 _B no interrupt occurred 1 _B interrupt was raised by the corresponding submodule Note: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding submodule. The bit is cleared automatically, when the corresponding interrupt in the submodule channels is cleared.
ARU_NEW_DATA1_I RQ	1	rh	ARU_NEW_DATA1 interrupt See bit 0.
ARU_ACC_ACK_IRQ	2	rh	ARU_ACC_ACK interrupt See bit 0.
BRC_IRQ	3	rh	BRC shared submodule interrupt See bit 0.

Generic Timer Module (GTM)

Field	Bits	Type	Description
AEI_IRQ	4	rh	AEI_IRQ interrupt See bit 0.
CMP_IRQ	5	rh	CMP shared submodule interrupt See bit 0.
SPE0_IRQ	6	rh	SPE0 shared submodule interrupt See bit 0.
SPE1_IRQ	7	rh	SPE1 shared submodule interrupt See bit 0.
Reserved	[15:8]	r	Reserved Read as zero, should be written as zero
PSM0_CH 0_IRQ	16	rh	PSM0 shared submodule channel 0 interrupt 0 _B no interrupt occurred 1 _B interrupt was raised by the corresponding submodule Note: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding submodule. Note: When set this bit represents one of the four interrupt sources FIFO_[x]_EMPTY, FIFO_[x]_FULL, FIFO_[x]_LOWER_WM or FIFO_[x]_UPPER_WM The bit is cleared automatically, when the corresponding interrupt in the submodule channels is cleared.
PSM0_CH 1_IRQ	17	rh	PSM0 shared submodule channel 1 interrupt See bit 16
PSM0_CH 2_IRQ	18	rh	PSM0 shared submodule channel 2 interrupt See bit 16
PSM0_CH 3_IRQ	19	rh	PSM0 shared submodule channel 3 interrupt See bit 16
PSM0_CH 4_IRQ	20	rh	PSM0 shared submodule channel 4 interrupt See bit 16
PSM0_CH 5_IRQ	21	rh	PSM0 shared submodule channel 5 interrupt See bit 16
PSM0_CH 6_IRQ	22	rh	PSM0 shared submodule channel 6 interrupt See bit 16
PSM0_CH 7_IRQ	23	rh	PSM0 shared submodule channel 7 interrupt See bit 16

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.18.5.2 Register ICM_IRQG_1 (DPLL Interrupt Group)

GTM_ICM_IRQG_1

GTM DPLL Interrupt Group

 (604_H)

 Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					DPL L_S ORI IRQ	DPL L_T ORI IRQ	DPL L_C DIS IRQ	DPL L_C DIT RQ	DPL L_TE 4_IR Q	DPL L_TE 3_IR Q	DPL L_TE 2_IR Q	DPL L_TE 1_IR Q	DPL L_TE 0_IR Q	DPL L_LL 2I_IR Q	DPL L_G L2I_I RQ
r					rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DPL L_EI _IRQ	DPL L_LL I_IR Q	DPL L_G LI_IR Q	DPL L_W 1I_IR Q	DPL L_W 2I_IR Q	DPL L_P WI_I RQ	DPL L_T AS_I RQ	DPL L_S AS_I RQ	DPL L_M TI_IR Q	DPL L_M SI_I RQ	DPL L_TI S_IR Q	DPL L_SI S_IR Q	DPL L_T AX_I RQ	DPL L_TI N_IR Q	DPL L_E DI_I RQ	DPL L_D CG_I RQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
DPLL_DC G_IRQ	0	rh	TRIGGER direction change detected. 0 _B no interrupt occurred 1 _B interrupt was raised by the corresponding submodule Note: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding submodule. The bit is cleared automatically, when the corresponding interrupt in the submodule channels is cleared.
DPLL_EDI _IRQ	1	rh	DPLL enable/disable interrupt See bit 0
DPLL_TIN _IRQ	2	rh	TRIGGER minimum hold time (THMI) violation detected interrupt See bit 0.
DPLL_TA X_IRQ	3	rh	TRIGGER maximum hold time (THMA) violation detected interrupt See bit 0.
DPLL_SIS _IRQ	4	rh	STATE inactive slope detected interrupt. See bit 0.

Generic Timer Module (GTM)

Field	Bits	Type	Description
DPLL_TIS_IRQ	5	rh	TRIGGER inactive slope detected interrupt. See bit 0.
DPLL_MSI_IRQ	6	rh	Missing STATE interrupt See bit 0.
DPLL_MTI_IRQ	7	rh	Missing TRIGGER interrupt See bit 0.
DPLL_SAS_IRQ	8	rh	STATE active slope detected See bit 0.
DPLL_TAS_IRQ	9	rh	TRIGGER active slope detected while NTI_CNT is zero See bit 0.
DPLL_PWI_IRQ	10	rh	Plausibility window (PVT) violation interrupt of TRIGGER See bit 0.
DPLL_W2I_IRQ	11	rh	Write access to RAM region 2 interrupt See bit 0
DPLL_W1I_IRQ	12	rh	Write access to RAM region 1B or 1C interrupt See bit 0
DPLL_GLI_IRQ	13	rh	Get of lock interrupt for SUB_INC1 See bit 0.
DPLL_LLI_IRQ	14	rh	Lost of lock interrupt for SUB_INC1 See bit 0.
DPLL_EI_IRQ	15	rh	Error interrupt See bit 0
DPLL_GL2I_IRQ	16	rh	Get of lock interrupt for SUB_INC2 See bit 0.
DPLL_LL2I_IRQ	17	rh	Lost of lock interrupt for SUB_INC2 See bit 0.
DPLL_TE0_IRQ	18	rh	TRIGGER event interrupt 0 See bit 0.
DPLL_TE1_IRQ	19	rh	TRIGGER event interrupt 1 See bit 0.
DPLL_TE2_IRQ	20	rh	TRIGGER event interrupt 2 See bit 0.
DPLL_TE3_IRQ	21	rh	TRIGGER event interrupt 3 See bit 0.

Generic Timer Module (GTM)

Field	Bits	Type	Description
DPLL_TE4 _IRQ	22	rh	TRIGGER event interrupt 4 See bit 0.
DPLL_CDI T_IRQ	23	rh	DPLL calculated duration interrupt for trigger See bit 0
DPLL_CDI S_IRQ	24	rh	DPLL calculated duration interrupt for state See bit 0
DPLL_TO RI_IRQ	25	rh	DPLL calculated duration interrupt for state See bit 0
DPLL_SO RI_IRQ	26	rh	DPLL calculated duration interrupt for state See bit 0
Reserved	[31:27]	r	Reserved Read as zero, should be written as zero

26.18.5.3 Register ICM_IRQG_2

GTM_ICM_IRQG_2

TIM Interrupt Group 0

(608_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TIM2 _CH 7_IR Q	TIM2 _CH 6_IR Q	TIM2 _CH 5_IR Q	TIM2 _CH 4_IR Q	TIM2 _CH 3_IR Q	TIM2 _CH 2_IR Q	TIM2 _CH 1_IR Q	TIM2 _CH 0_IR Q
r								rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIM1 _CH 7_IR Q	TIM1 _CH 6_IR Q	TIM1 _CH 5_IR Q	TIM1 _CH 4_IR Q	TIM1 _CH 3_IR Q	TIM1 _CH 2_IR Q	TIM1 _CH 1_IR Q	TIM1 _CH 0_IR Q	TIM0 _CH 7_IR Q	TIM0 _CH 6_IR Q	TIM0 _CH 5_IR Q	TIM0 _CH 4_IR Q	TIM0 _CH 3_IR Q	TIM0 _CH 2_IR Q	TIM0 _CH 1_IR Q	TIM0 _CH 0_IR Q
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TIM0_CH0_IRQ	0	rh	<p>TIM0 shared interrupt channel 0</p> <p>0_B no interrupt occurred</p> <p>1_B interrupt was raised by the corresponding submodule</p> <p>Note: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding submodule.</p> <p>Note: When set this bit represents one of the six interrupt sources NEWVALx_IRQ, ECNTOFLx_IRQ, CNTOFLx_IRQ, GPRzOFLx_IRQ, GLITCHDET_x_IRQ or TO_x_IRQ.</p> <p>The bit is cleared automatically, when the corresponding interrupt in the submodule channels is cleared.</p>
TIM0_CH1_IRQ	1	rh	<p>TIM0 shared interrupt channel 1</p> <p>See bit 0</p>
TIM0_CH2_IRQ	2	rh	<p>TIM0 shared interrupt channel 2</p> <p>See bit 0</p>
TIM0_CH3_IRQ	3	rh	<p>TIM0 shared interrupt channel 3</p> <p>See bit 0</p>
TIM0_CH4_IRQ	4	rh	<p>TIM0 shared interrupt channel 4</p> <p>See bit 0</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
TIM0_CH5_IRQ	5	rh	TIM0 shared interrupt channel 5 See bit 0
TIM0_CH6_IRQ	6	rh	TIM0 shared interrupt channel 6 See bit 0
TIM0_CH7_IRQ	7	rh	TIM0 shared interrupt channel 7 See bit 0
TIM1_CH0_IRQ	8	rh	TIM1 shared interrupt channel 0 See bit 0
TIM1_CH1_IRQ	9	rh	TIM1 shared interrupt channel 1 See bit 0
TIM1_CH2_IRQ	10	rh	TIM1 shared interrupt channel 2 See bit 0
TIM1_CH3_IRQ	11	rh	TIM1 shared interrupt channel 3 See bit 0
TIM1_CH4_IRQ	12	rh	TIM1 shared interrupt channel 4 See bit 0
TIM1_CH5_IRQ	13	rh	TIM1 shared interrupt channel 5 See bit 0
TIM1_CH6_IRQ	14	rh	TIM1 shared interrupt channel 6 See bit 0
TIM1_CH7_IRQ	15	rh	TIM1 shared interrupt channel 7 See bit 0
TIM2_CH0_IRQ	16	rh	TIM2 shared interrupt channel 0 See bit 0
TIM2_CH1_IRQ	17	rh	TIM2 shared interrupt channel 1 See bit 0
TIM2_CH2_IRQ	18	rh	TIM2 shared interrupt channel 2 See bit 0
TIM2_CH3_IRQ	19	rh	TIM2 shared interrupt channel 3 See bit 0
TIM2_CH4_IRQ	20	rh	TIM2 shared interrupt channel 4 See bit 0
TIM2_CH5_IRQ	21	rh	TIM2 shared interrupt channel 5 See bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
TIM2_CH6 _IRQ	22	rh	TIM2 shared interrupt channel 6 See bit 0
TIM2_CH7 _IRQ	23	rh	TIM2 shared interrupt channel 7 See bit 0
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

26.18.5.4 Register ICM_IRQG_4

GTM_ICM_IRQG_4

MCS Interrupt Group 0

 (610_H)

 Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								MCS 2_C H7_I RQ	MCS 2_C H6_I RQ	MCS 2_C H5_I RQ	MCS 2_C H4_I RQ	MCS 2_C H3_I RQ	MCS 2_C H2_I RQ	MCS 2_C H1_I RQ	MCS 2_C H0_I RQ
r								rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCS 1_C H7_I RQ	MCS 1_C H6_I RQ	MCS 1_C H5_I RQ	MCS 1_C H4_I RQ	MCS 1_C H3_I RQ	MCS 1_C H2_I RQ	MCS 1_C H1_I RQ	MCS 1_C H0_I RQ	MCS 0_C H7_I RQ	MCS 0_C H6_I RQ	MCS 0_C H5_I RQ	MCS 0_C H4_I RQ	MCS 0_C H3_I RQ	MCS 0_C H2_I RQ	MCS 0_C H1_I RQ	MCS 0_C H0_I RQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MCS0_CH 0_IRQ	0	rh	MCS0 channel 0 interrupt 0 _B no interrupt occurred 1 _B interrupt was raised by the corresponding submodule Note: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding submodule. The bit is cleared automatically, when the corresponding interrupt in the submodule channels is cleared.
MCS0_CH 1_IRQ	1	rh	MCS0 channel 1 interrupt See bit 0
MCS0_CH 2_IRQ	2	rh	MCS0 channel 2 interrupt See bit 0
MCS0_CH 3_IRQ	3	rh	MCS0 channel 3 interrupt See bit 0
MCS0_CH 4_IRQ	4	rh	MCS0 channel 4 interrupt See bit 0
MCS0_CH 5_IRQ	5	rh	MCS0 channel 5 interrupt See bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
MCS0_CH 6_IRQ	6	rh	MCS0 channel 6 interrupt See bit 0
MCS0_CH 7_IRQ	7	rh	MCS0 channel 7 interrupt See bit 0
MCS1_CH 0_IRQ	8	rh	MCS1 channel 0 interrupt See bit 0
MCS1_CH 1_IRQ	9	rh	MCS1 channel 1 interrupt See bit 0
MCS1_CH 2_IRQ	10	rh	MCS1 channel 2 interrupt See bit 0
MCS1_CH 3_IRQ	11	rh	MCS1 channel 3 interrupt See bit 0
MCS1_CH 4_IRQ	12	rh	MCS1 channel 4 interrupt See bit 0
MCS1_CH 5_IRQ	13	rh	MCS1 channel 5 interrupt See bit 0
MCS1_CH 6_IRQ	14	rh	MCS1 channel 6 interrupt See bit 0
MCS1_CH 7_IRQ	15	rh	MCS1 channel 7 interrupt See bit 0
MCS2_CH 0_IRQ	16	rh	MCS2 channel 0 interrupt
MCS2_CH 1_IRQ	17	rh	MCS2 channel 1 interrupt See bit 0
MCS2_CH 2_IRQ	18	rh	MCS2 channel 2 interrupt See bit 0
MCS2_CH 3_IRQ	19	rh	MCS2 channel 3 interrupt See bit 0
MCS2_CH 4_IRQ	20	rh	MCS2 channel 4 interrupt See bit 0
MCS2_CH 5_IRQ	21	rh	MCS2 channel 5 interrupt See bit 0
MCS2_CH 6_IRQ	22	rh	MCS2 channel 6 interrupt See bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
MCS2_CH 7_IRQ	23	rh	MCS2 channel 7 interrupt See bit 0
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

26.18.5.5 Register ICM_IRQG_6

GTM_ICM_IRQG_6

TOM Interrupt Group 0

 (618_H)

 Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C	TOM 1_C
H15_ IRQ	H14_ IRQ	H13_ IRQ	H12_ IRQ	H11_ IRQ	H10_ IRQ	H9_ RQ	H8_ RQ	H7_ RQ	H6_ RQ	H5_ RQ	H4_ RQ	H3_ RQ	H2_ RQ	H1_ RQ	H0_ RQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C	TOM 0_C
H15_ IRQ	H14_ IRQ	H13_ IRQ	H12_ IRQ	H11_ IRQ	H10_ IRQ	H9_ RQ	H8_ RQ	H7_ RQ	H6_ RQ	H5_ RQ	H4_ RQ	H3_ RQ	H2_ RQ	H1_ RQ	H0_ RQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TOM0_CH 0_IRQ	0	rh	TOM0 channel 0 shared interrupt 0 _B no interrupt occurred 1 _B interrupt was raised by the corresponding submodule Note: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding submodule. The bit is cleared automatically, when the corresponding interrupt in the submodule channels is cleared.
TOM0_CH 1_IRQ	1	rh	TOM0 channel 1 shared interrupt See bit 0
TOM0_CH 2_IRQ	2	rh	TOM0 channel 2 shared interrupt See bit 0
TOM0_CH 3_IRQ	3	rh	TOM0 channel 3 shared interrupt See bit 0
TOM0_CH 4_IRQ	4	rh	TOM0 channel 4 shared interrupt See bit 0
TOM0_CH 5_IRQ	5	rh	TOM0 channel 5 shared interrupt See bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
TOM0_CH 6_IRQ	6	rh	TOM0 channel 6 shared interrupt See bit 0
TOM0_CH 7_IRQ	7	rh	TOM0 channel 7 shared interrupt See bit 0
TOM0_CH 8_IRQ	8	rh	TOM0 channel 8 shared interrupt See bit 0
TOM0_CH 9_IRQ	9	rh	TOM0 channel 9 shared interrupt See bit 0
TOM0_CH 10_IRQ	10	rh	TOM0 channel 10 shared interrupt See bit 0
TOM0_CH 11_IRQ	11	rh	TOM0 channel 11 shared interrupt See bit 0
TOM0_CH 12_IRQ	12	rh	TOM0 channel 12 shared interrupt See bit 0
TOM0_CH 13_IRQ	13	rh	TOM0 channel 13 shared interrupt See bit 0
TOM0_CH 14_IRQ	14	rh	TOM0 channel 14 shared interrupt See bit 0
TOM0_CH 15_IRQ	15	rh	TOM0 channel 15 shared interrupt See bit 0
TOM1_CH 0_IRQ	16	rh	TOM1 channel 0 shared interrupt See bit 0
TOM1_CH 1_IRQ	17	rh	TOM1 channel 1 shared interrupt See bit 0
TOM1_CH 2_IRQ	18	rh	TOM1 channel 2 shared interrupt See bit 0
TOM1_CH 3_IRQ	19	rh	TOM1 channel 3 shared interrupt See bit 0
TOM1_CH 4_IRQ	20	rh	TOM1 channel 4 shared interrupt See bit 0
TOM1_CH 5_IRQ	21	rh	TOM1 channel 5 shared interrupt See bit 0
TOM1_CH 6_IRQ	22	rh	TOM1 channel 6 shared interrupt See bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
TOM1_CH 7_IRQ	23	rh	TOM1 channel 7 shared interrupt See bit 0
TOM1_CH 8_IRQ	24	rh	TOM1 channel 8 shared interrupt See bit 0
TOM1_CH 9_IRQ	25	rh	TOM1 channel 9 shared interrupt See bit 0
TOM1_CH 10_IRQ	26	rh	TOM1 channel 10 shared interrupt See bit 0
TOM1_CH 11_IRQ	27	rh	TOM1 channel 11 shared interrupt See bit 0
TOM1_CH 12_IRQ	28	rh	TOM1 channel 12 shared interrupt See bit 0
TOM1_CH 13_IRQ	29	rh	TOM1 channel 13 shared interrupt See bit 0
TOM1_CH 14_IRQ	30	rh	TOM1 channel 14 shared interrupt See bit 0
TOM1_CH 15_IRQ	31	rh	TOM1 channel 15 shared interrupt See bit 0

26.18.5.6 Register ICM_IRQG_9

GTM_ICM_IRQG_9

ATOM Interrupt Group 0

 (624_H)

 Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ATO M3_ CH7 _IRQ	ATO M3_ CH6 _IRQ	ATO M3_ CH5 _IRQ	ATO M3_ CH4 _IRQ	ATO M3_ CH3 _IRQ	ATO M3_ CH2 _IRQ	ATO M3_ CH1 _IRQ	ATO M3_ CH0 _IRQ	ATO M2_ CH7 _IRQ	ATO M2_ CH6 _IRQ	ATO M2_ CH5 _IRQ	ATO M2_ CH4 _IRQ	ATO M2_ CH3 _IRQ	ATO M2_ CH2 _IRQ	ATO M2_ CH1 _IRQ	ATO M2_ CH0 _IRQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ATO M1_ CH7 _IRQ	ATO M1_ CH6 _IRQ	ATO M1_ CH5 _IRQ	ATO M1_ CH4 _IRQ	ATO M1_ CH3 _IRQ	ATO M1_ CH2 _IRQ	ATO M1_ CH1 _IRQ	ATO M1_ CH0 _IRQ	ATO M0_ CH7 _IRQ	ATO M0_ CH6 _IRQ	ATO M0_ CH5 _IRQ	ATO M0_ CH4 _IRQ	ATO M0_ CH3 _IRQ	ATO M0_ CH2 _IRQ	ATO M0_ CH1 _IRQ	ATO M0_ CH0 _IRQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
ATOM0_C H0_IRQ	0	rh	ATOM0 channel 0 shared interrupt 0 _B no interrupt occurred 1 _B interrupt was raised by the corresponding submodule Note: This bit is only set, when the interrupt is enabled in the interrupt enable register of the corresponding submodule. The bit is cleared automatically, when the corresponding interrupt in the submodule channels is cleared.
ATOM0_C H1_IRQ	1	rh	ATOM0 channel 1 shared interrupt See bit 0
ATOM0_C H2_IRQ	2	rh	ATOM0 channel 2 shared interrupt See bit 0
ATOM0_C H3_IRQ	3	rh	ATOM0 channel 3 shared interrupt See bit 0
ATOM0_C H4_IRQ	4	rh	ATOM0 channel 4 shared interrupt See bit 0
ATOM0_C H5_IRQ	5	rh	ATOM0 channel 5 shared interrupt See bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
ATOM0_C H6_IRQ	6	rh	ATOM0 channel 6 shared interrupt See bit 0
ATOM0_C H7_IRQ	7	rh	ATOM0 channel 7 shared interrupt See bit 0
ATOM1_C H0_IRQ	8	rh	ATOM1 channel 0 shared interrupt See bit 0
ATOM1_C H1_IRQ	9	rh	ATOM1 channel 1 shared interrupt See bit 0
ATOM1_C H2_IRQ	10	rh	ATOM1 channel 2 shared interrupt See bit 0
ATOM1_C H3_IRQ	11	rh	ATOM1 channel 3 shared interrupt See bit 0
ATOM1_C H4_IRQ	12	rh	ATOM1 channel 4 shared interrupt See bit 0
ATOM1_C H5_IRQ	13	rh	ATOM1 channel 5 shared interrupt See bit 0
ATOM1_C H6_IRQ	14	rh	ATOM1 channel 6 shared interrupt See bit 0
ATOM1_C H7_IRQ	15	rh	ATOM1 channel 7 shared interrupt See bit 0
ATOM2_C H0_IRQ	16	rh	ATOM2 channel 0 shared interrupt See bit 0
ATOM2_C H1_IRQ	17	rh	ATOM2 channel 1 shared interrupt See bit 0
ATOM2_C H2_IRQ	18	rh	ATOM2 channel 2 shared interrupt See bit 0
ATOM2_C H3_IRQ	19	rh	ATOM2 channel 3 shared interrupt See bit 0
ATOM2_C H4_IRQ	20	rh	ATOM2 channel 4 shared interrupt See bit 0
ATOM2_C H5_IRQ	21	rh	ATOM2 channel 5 shared interrupt See bit 0
ATOM2_C H6_IRQ	22	rh	ATOM2 channel 6 shared interrupt See bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
ATOM2_C H7_IRQ	23	rh	ATOM2 channel 7 shared interrupt See bit 0
ATOM3_C H0_IRQ	24	rh	ATOM3 channel 0 shared interrupt See bit 0
ATOM3_C H1_IRQ	25	rh	ATOM3 channel 1 shared interrupt See bit 0
ATOM3_C H2_IRQ	26	rh	ATOM3 channel 2 shared interrupt See bit 0
ATOM3_C H3_IRQ	27	rh	ATOM3 channel 3 shared interrupt See bit 0
ATOM3_C H4_IRQ	28	rh	ATOM3 channel 4 shared interrupt See bit 0
ATOM3_C H5_IRQ	29	rh	ATOM3 channel 5 shared interrupt See bit 0
ATOM3_C H6_IRQ	30	rh	ATOM3 channel 6 shared interrupt See bit 0
ATOM3_C H7_IRQ	31	rh	ATOM3 channel 7 shared interrupt See bit 0

Generic Timer Module (GTM)
26.18.5.7 Register ICM_IRQG_MEI (Module Error Interrupt)
GTM_ICM_IRQG_MEI
ICM Module Error Interrupt Register (630_H)
Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Reserved						DPL L_EI RQ	CMP _EIR Q	Reserved			SPE 1_EI RQ	SPE 0_EI RQ	Reserved			
r						rh	rh	r			rh	rh	r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Reserved	MCS 2_EI RQ	MCS 1_EI RQ	MCS 0_EI RQ	Reserved				TIM2 _EIR Q	TIM1 _EIR Q	TIM0 _EIR Q	Reserved	FIFO 0_EI RQ	BRC _EIR Q	GTM _EIR Q		
r	rh	rh	rh	r				rh	rh	rh	r	rh	r	rh		

Field	Bits	Type	Description
GTM_EIR Q	0	rh	GTM Error interrupt request 0 _B no interrupt occurred 1 _B interrupt was raised by the corresponding submodule <i>Note: Note: This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding submodule.</i>
BRC_EIR Q	1	rh	BRC error interrupt See bit 0
FIFO0_EIR Q	2	rh	FIFO0 error interrupt See bit 0
Reserved	3	r	Reserved Read as zero, should be written as zero
TIM0_EIR Q	4	rh	TIM0 error interrupt See bit 0
TIM1_EIR Q	5	rh	TIM1 error interrupt See bit 0
TIM2_EIR Q	6	rh	TIM2 error interrupt See bit 0
Reserved	[11:7]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

Field	Bits	Type	Description
MCS0_EIR Q	12	rh	MCS0 error interrupt See bit 0
MCS1_EIR Q	13	rh	MCS1 error interrupt See bit 0
MCS2_EIR Q	14	rh	MCS2 error interrupt See bit 0
Reserved	[19:15]	r	Reserved Read as zero, should be written as zero
SPE0_EIR Q	20	rh	SPE0 error interrupt See bit 0
SPE1_EIR Q	21	rh	SPE1 error interrupt See bit 0
Reserved	[23:22]	r	Reserved Read as zero, should be written as zero
CMP_EIR Q	24	rh	CMP error interrupt See bit 0
DPLL_EIR Q	25	rh	DPLL error interrupt See bit 0
Reserved	[31:26]	r	Reserved Read as zero, should be written as zero

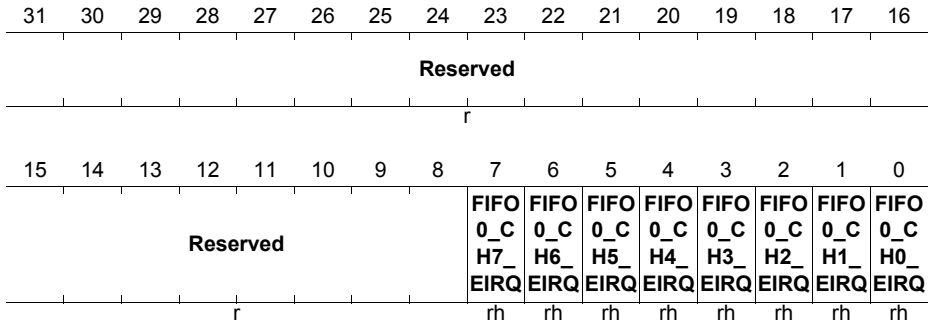
26.18.5.8 Register ICM_IRQG_CEI0 (Channel Error Interrupt 0)

GTM_ICM_IRQG_CEI0

ICM Channel Error Interrupt 0 Register

(634_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
FIFO0_CH 0_EIRQ	0	rh	<p>FIFO0 channel 0 error interrupt</p> <p>0_B no interrupt occurred</p> <p>1_B error interrupt was raised by the corresponding submodule</p> <p>Note: This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding submodule.</p>
FIFO0_CH 1_EIRQ	1	rh	<p>FIFO0 channel 1 shared interrupt</p> <p>See bit 0</p>
FIFO0_CH 2_EIRQ	2	rh	<p>FIFO0 channel 2 shared interrupt</p> <p>See bit 0</p>
FIFO0_CH 3_EIRQ	3	rh	<p>FIFO0 channel 3 shared interrupt</p> <p>See bit 0</p>
FIFO0_CH 4_EIRQ	4	rh	<p>FIFO0 channel 4 shared interrupt</p> <p>See bit 0</p>
FIFO0_CH 5_EIRQ	5	rh	<p>FIFO0 channel 5 shared interrupt</p> <p>See bit 0</p>
FIFO0_CH 6_EIRQ	6	rh	<p>FIFO0 channel 6 shared interrupt</p> <p>See bit 0</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
FIFO0_CH 7_EIRQ	7	rh	FIFO0 channel 7 shared interrupt See bit 0
Reserved	[31:8]	r	Reserved Read as zero, should be written as zero

Generic Timer Module (GTM)

26.18.5.9 Register ICM_IRQG_CEI1 (Channel Error Interrupt 1)

GTM_ICM_IRQG_CEI1

ICM Channel Error Interrupt 1 Register

(638_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TIM2 _CH 7_EI RQ	TIM2 _CH 6_EI RQ	TIM2 _CH 5_EI RQ	TIM2 _CH 4_EI RQ	TIM2 _CH 3_EI RQ	TIM2 _CH 2_EI RQ	TIM2 _CH 1_EI RQ	TIM2 _CH 0_EI RQ
r								rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIM1 _CH 7_EI RQ	TIM1 _CH 6_EI RQ	TIM1 _CH 5_EI RQ	TIM1 _CH 4_EI RQ	TIM1 _CH 3_EI RQ	TIM1 _CH 2_EI RQ	TIM1 _CH 1_EI RQ	TIM1 _CH 0_EI RQ	TIM0 _CH 7_EI RQ	TIM0 _CH 6_EI RQ	TIM0 _CH 5_EI RQ	TIM0 _CH 4_EI RQ	TIM0 _CH 3_EI RQ	TIM0 _CH 2_EI RQ	TIM0 _CH 1_EI RQ	TIM0 _CH 0_EI RQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TIM0_CH0_EIRQ	0	rh	<p>TIM0 channel 0 error interrupt</p> <p>0_B no error interrupt occurred</p> <p>1_B error interrupt was raised by the corresponding submodule</p> <p><i>Note: This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding submodule.</i></p>
TIM0_CH1_EIRQ	1	rh	<p>TIM0 channel 1 error interrupt</p> <p>See bit 0</p>
TIM0_CH2_EIRQ	2	rh	<p>TIM0 channel 2 error interrupt</p> <p>See bit 0</p>
TIM0_CH3_EIRQ	3	rh	<p>TIM0 channel 3 error interrupt</p> <p>See bit 0</p>
TIM0_CH4_EIRQ	4	rh	<p>TIM0 channel 4 error interrupt</p> <p>See bit 0</p>
TIM0_CH5_EIRQ	5	rh	<p>TIM0 channel 5 error interrupt</p> <p>See bit 0</p>
TIM0_CH6_EIRQ	6	rh	<p>TIM0 channel 6 error interrupt</p> <p>See bit 0</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
TIM0_CH7_EIRQ	7	rh	TIM0 channel 7 error interrupt See bit 0
TIM1_CH0_EIRQ	8	rh	TIM1 channel 0 error interrupt See bit 0
TIM1_CH1_EIRQ	9	rh	TIM1 channel 1 error interrupt See bit 0
TIM1_CH2_EIRQ	10	rh	TIM1 channel 2 error interrupt See bit 0
TIM1_CH3_EIRQ	11	rh	TIM1 channel 3 error interrupt See bit 0
TIM1_CH4_EIRQ	12	rh	TIM1 channel 4 error interrupt See bit 0
TIM1_CH5_EIRQ	13	rh	TIM1 channel 5 error interrupt See bit 0
TIM1_CH6_EIRQ	14	rh	TIM1 channel 6 error interrupt See bit 0
TIM1_CH7_EIRQ	15	rh	TIM1 channel 7 error interrupt See bit 0
TIM2_CH0_EIRQ	16	rh	TIM2 channel 0 error interrupt See bit 0
TIM2_CH1_EIRQ	17	rh	TIM2 channel 1 error interrupt See bit 0
TIM2_CH2_EIRQ	18	rh	TIM2 channel 2 error interrupt See bit 0
TIM2_CH3_EIRQ	19	rh	TIM2 channel 3 error interrupt See bit 0
TIM2_CH4_EIRQ	20	rh	TIM2 channel 4 error interrupt See bit 0
TIM2_CH5_EIRQ	21	rh	TIM2 channel 5 error interrupt See bit 0
TIM2_CH6_EIRQ	22	rh	TIM2 channel 6 error interrupt See bit 0
TIM2_CH7_EIRQ	23	rh	TIM2 channel 7 error interrupt See bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

26.18.5.10 Register ICM_IRQG_CEI3 (Channel Error Interrupt 3)
GTM_ICM_IRQG_CEI3
ICM Channel Error Interrupt 3 Register

 (640_H)

 Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								MCS 2_C H7_ EIRQ	MCS 2_C H6_ EIRQ	MCS 2_C H5_ EIRQ	MCS 2_C H4_ EIRQ	MCS 2_C H3_ EIRQ	MCS 2_C H2_ EIRQ	MCS 2_C H1_ EIRQ	MCS 2_C H0_ EIRQ
r								rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCS 1_C H7_ EIRQ	MCS 1_C H6_ EIRQ	MCS 1_C H5_ EIRQ	MCS 1_C H4_ EIRQ	MCS 1_C H3_ EIRQ	MCS 1_C H2_ EIRQ	MCS 1_C H1_ EIRQ	MCS 1_C H0_ EIRQ	MCS 0_C H7_ EIRQ	MCS 0_C H6_ EIRQ	MCS 0_C H5_ EIRQ	MCS 0_C H4_ EIRQ	MCS 0_C H3_ EIRQ	MCS 0_C H2_ EIRQ	MCS 0_C H1_ EIRQ	MCS 0_C H0_ EIRQ
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MCS0_CH 0_EIRQ	0	rh	MCS0 channel 0 error interrupt 0 _B no error interrupt occurred 1 _B error interrupt was raised by the corresponding submodule <i>Note: This bit is only set, when the error interrupt is enabled in the error interrupt enable register of the corresponding submodule.</i>
MCS0_CH 1_EIRQ	1	rh	MCS0 channel 1 error interrupt See bit 0
MCS0_CH 2_EIRQ	2	rh	MCS0 channel 2 error interrupt See bit 0
MCS0_CH 3_EIRQ	3	rh	MCS0 channel 3 error interrupt See bit 0
MCS0_CH 4_EIRQ	4	rh	MCS0 channel 4 error interrupt See bit 0
MCS0_CH 5_EIRQ	5	rh	MCS0 channel 5 error interrupt See bit 0
MCS0_CH 6_EIRQ	6	rh	MCS0 channel 6 error interrupt See bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
MCS0_CH 7_EIRQ	7	rh	MCS0 channel 7 error interrupt See bit 0
MCS1_CH 0_EIRQ	8	rh	MCS1 channel 0 error interrupt See bit 0
MCS1_CH 1_EIRQ	9	rh	MCS1 channel 1 error interrupt See bit 0
MCS1_CH 2_EIRQ	10	rh	MCS1 channel 2 error interrupt See bit 0
MCS1_CH 3_EIRQ	11	rh	MCS1 channel 3 error interrupt See bit 0
MCS1_CH 4_EIRQ	12	rh	MCS1 channel 4 error interrupt See bit 0
MCS1_CH 5_EIRQ	13	rh	MCS1 channel 5 error interrupt See bit 0
MCS1_CH 6_EIRQ	14	rh	MCS1 channel 6 error interrupt See bit 0
MCS1_CH 7_EIRQ	15	rh	MCS1 channel 7 error interrupt See bit 0
MCS2_CH 0_EIRQ	16	rh	MCS2 channel 0 error interrupt See bit 0
MCS2_CH 1_EIRQ	17	rh	MCS2 channel 1 error interrupt See bit 0
MCS2_CH 2_EIRQ	18	rh	MCS2 channel 2 error interrupt See bit 0
MCS2_CH 3_EIRQ	19	rh	MCS2 channel 3 error interrupt See bit 0
MCS2_CH 4_EIRQ	20	rh	MCS2 channel 4 error interrupt See bit 0
MCS2_CH 5_EIRQ	21	rh	MCS2 channel 5 error interrupt See bit 0
MCS2_CH 6_EIRQ	22	rh	MCS2 channel 6 error interrupt See bit 0
MCS2_CH 7_EIRQ	23	rh	MCS2 channel 7 error interrupt See bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

26.19 Output Compare Unit (CMP)

26.19.1 Overview

The Output Compare Unit (CMP) is designed for the use in safety relevant applications. The main idea is to have the possibility to duplicate outputs in order to be compared in this unit. Because of the simple EXOR function used it is necessary to ensure the total cycle accurate output behaviour of the output modules to be compared. This is given when two ATOM units produce output signals at the same time stamp or when two TOMs have the same configuration and start their output generation at the same time. This is possible by means of the trigger mechanisms TRIG_x provided by the TOMs as shown in the [Section 26.11.1.1](#) (TOM Block Diagram). It is not necessary to compare each output channel with each other.

The CMP enables the comparison of 2x24 channels of the TOM and ATOM units respectively and is restricted to neighbor channels. Thus, channel 0 is compared with channel 1, channel 2 with 3 and so on until the comparison of channel 22 with channel 23.

26.19.1.1 Architecture of the Compare Unit

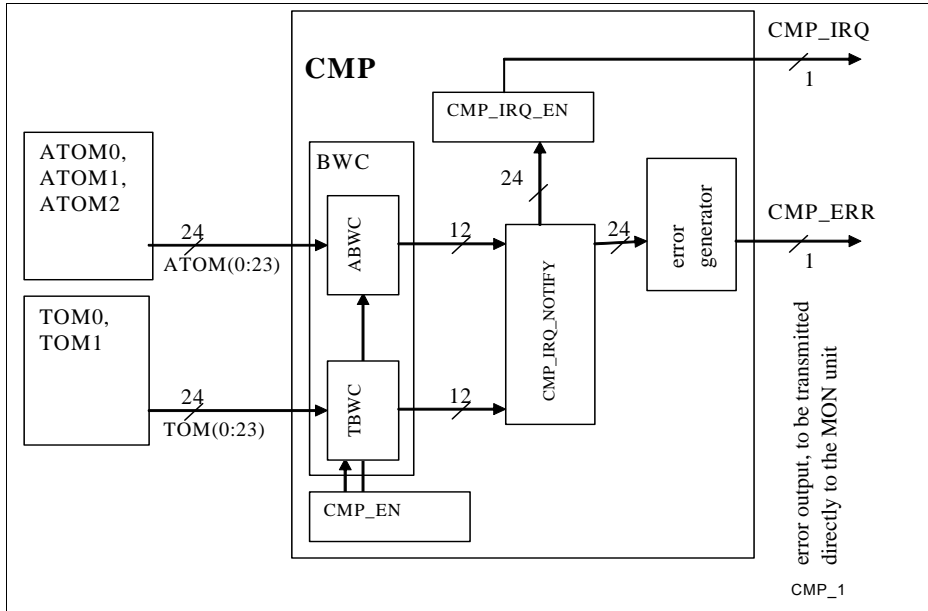


Figure 26-83 Architecture of the Compare Unit

26.19.2 Bitwise Compare Unit (BWC)

The Bitwise Compare Unit compares in pairs the combinations shown in following table

Table 26-58 Bitwise compare

TBWC/ABWC Comparator Number	Compare TOM/ATOM Bit Number one	Compare TOM/ATOM Bit Number Two	Output Number
0	0	1	0
1	2	3	1
2	4	5	2
3	6	7	3
4	8	9	4
5	10	11	5

Generic Timer Module (GTM)
Table 26-58 Bitwise compare (cont'd)

TBWC/ABWC Comparator Number	Compare TOM/ATOM Bit Number one	Compare TOM/ATOM Bit Number Two	Output Number
6	12	13	6
7	14	15	7
8	16	17	8
9	18	19	9
10	20	21	10
11	22	23	11

26.19.3 Configuration of the Compare Unit

Because of the restrictions described in the section above the Compare Unit consists of 24 antivalence (EXOR) elements, a select register CMP_EN which selects the corresponding comparisons and a status register CMP_IRQ_NOTIFY which shows and stores each mismatching result, when selected.

For each with CMP_IRQ_EN enabled mismatching error an interrupt signal on CMP_IRQ is generated.

For each with CMP_EIRQ_EN enabled mismatching error an interrupt signal on CMP_EIRQ is generated.

26.19.4 Error Generator

The error generator generates an error signal to be transmitted directly to the MON unit and independently from the CMP_IRQ and CMP_EIRQ. The error is set when in the CMP_IRQ_NOTIFY register at least one bit is set. The CMP_IRQ_NOTIFY bits are not maskable for this purpose.

The CMP_ERR output reflects its status in the status register of the Monitor Unit, which is to be polled by the CPU.

26.19.5 CMP Interrupt Signal

The CMP submodule has two interrupt signals, one normal interrupt and one error interrupt. The source of both interrupts can be determined by reading the CMP_IRQ_NOTIFY register for under consideration of CMP_IRQ_EN register and CMP_EIRQ_EN register. Each source can be forced separately for debug purposes using the interrupt force CMP_IRQ_FORCINT register. CMP_IRQ_MODE configures interrupt output characteristic. All interrupt modes are described in detail in [Section 26.2.5](#).

Table 26-59 CMP interrupt interface

Signal	Description
CMP_EIRQ	Mismatching interrupt of outputs to be compared, when enabled
CMP_IRQ	Mismatching interrupt of outputs to be compared, when enabled

26.19.6 CMP Configuration Registers Overview

CMP contains following configuration registers:

Table 26-60 CMP Configuration Registers Overview

Register Name	Description	Details in Section
CMP_EN	Comparator enable register	Section 26.19.7.1
CMP_IRQ_NOTIFY	Event notification register	Section 26.19.7.2
CMP_IRQ_EN	Interrupt enable register	Section 26.19.7.3
CMP_IRQ_FORCINT	Interrupt force register	Section 26.19.7.4
CMP_IRQ_MODE	IRQ mode configuration register	Section 26.19.7.5
CMP_EIRQ_EN	Error interrupt enable register	Section 26.19.7.6

26.19.7 CMP Configuration Registers Description

All of the following registers are 32-bit only accessible.

26.19.7.1 Register CMP_EN

GTM_CMP_EN

CMP Comparator Enable Register

(200_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TBW C11_ EN	TBW C10_ EN	TBW C9_ EN	TBW C8_ EN	TBW C7_ EN	TBW C6_ EN	TBW C5_ EN	TBW C4_ EN
r								r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBW C3_ EN	TBW C2_ EN	TBW C1_ EN	TBW C0_ EN	ABW C11_ EN	ABW C10_ EN	ABW C9_ EN	ABW C8_ EN	ABW C7_ EN	ABW C6_ EN	ABW C5_ EN	ABW C4_ EN	ABW C3_ EN	ABW C2_ EN	ABW C1_ EN	ABW C0_ EN
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
ABWC0_EN	0	rw	Enable comparator 0 in ABWC (see Section 26.19.2) 0 _B ABWC Comparator 0 is disabled 1 _B ABWC Comparator 0 is enabled
ABWC1_EN	1	rw	Enable comparator 1 in ABWC see bit 0
ABWC2_EN	2	rw	Enable comparator 2 in ABWC see bit 0
ABWC3_EN	3	rw	Enable comparator 3 in ABWC see bit 0
ABWC4_EN	4	rw	Enable comparator 4 in ABWC see bit 0
ABWC5_EN	5	rw	Enable comparator 5 in ABWC see bit 0
ABWC6_EN	6	rw	Enable comparator 6 in ABWC see bit 0
ABWC7_EN	7	rw	Enable comparator 7 in ABWC see bit 0
ABWC8_EN	8	rw	Enable comparator 8 in ABWC see bit 0
ABWC9_EN	9	rw	Enable comparator 9 in ABW see bit 0
ABWC10_EN	10	rw	Enable comparator 10 in ABWC see bit 0
ABWC11_EN	11	rw	Enable comparator 11 in ABWC see bit 0
TBWC0_EN	12	rw	Enable comparator 0 in TBWC 0 _B TBWC comparator 0 is disabled 1 _B TBWC comparator 0 is enabled
TBWC1_EN	13	rw	Enable comparator 1 in TBWC see bit 12
TBWC2_EN	14	rw	Enable comparator 2 in TBWC see bit 12
TBWC3_EN	15	rw	Enable comparator 3 in TBWC see bit 12

Generic Timer Module (GTM)

Field	Bits	Type	Description
TBWC4_EN	16	rw	Enable comparator 4 in TBWC see bit 12
TBWC5_EN	17	rw	Enable comparator 5 in TBWC see bit 12
TBWC6_EN	18	rw	Enable comparator 6 in TBWC see bit 12
TBWC7_EN	19	rw	Enable comparator 7 in TBWC see bit 12
TBWC8_EN	20	rw	Enable comparator 8 in TBWC see bit 12
TBWC9_EN	21	rw	Enable comparator 9 in TBWC see bit 12
TBWC10_EN	22	rw	Enable comparator 10 in TBWC see bit 12
TBWC11_EN	23	rw	Enable comparator 11 in TBWC see bit 12
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

26.19.7.2 Register CMP_IRQ_NOTIFY
GTM_CMP_IRQ_NOTIFY
CMP Event Notification Register (204_H)
Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TBW C11	TBW C10	TBW C9	TBW C8	TBW C7	TBW C6	TBW C5	TBW C4
r								rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBW C3	TBW C2	TBW C1	TBW C0	ABW C11	ABW C10	ABW C9	ABW C8	ABW C7	ABW C6	ABW C5	ABW C4	ABW C3	ABW C2	ABW C1	ABW C0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
ABWC0	0	rwh	Error indication for ABWC0 0_B no error recognized on ATOM sub modules bits 0 and 1 (see Section 26.19.2) 1_B an error was recognized on corresponding ATOM sub modules bits Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ABWC1	1	rwh	Error indication for ABWC1 see bit 0
ABWC2	2	rwh	Error indication for ABWC2 see bit 0
ABWC3	3	rwh	Error indication for ABWC3 see bit 0
ABWC4	4	rwh	Error indication for ABWC4 see bit 0
ABWC5	5	rwh	Error indication for ABWC5 see bit 0
ABWC6	6	rwh	Error indication for ABWC6 see bit 0
ABWC7	7	rwh	Error indication for ABWC7 see bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
ABWC8	8	rwh	Error indication for ABWC8 see bit 0
ABWC9	9	rwh	Error indication for ABWC9 see bit 0
ABWC10	10	rwh	Error indication for ABWC10 see bit 0
ABWC11	11	rwh	Error indication for ABWC11 see bit 0
TBWC0	12	rwh	TOM sub modules outputs bitwise comparator 0 error indication 0_B no error recognized on TOM sub modules bits 0 and 1 (see Section 26.19.2) 1_B an error was recognized on corresponding TOM sub modules bits Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
TBWC1	13	rwh	TOM sub modules outputs bitwise comparator 1 error indication see bit 12
TBWC2	14	rwh	TOM sub modules outputs bitwise comparator 2 error indication see bit 12
TBWC3	15	rwh	TOM sub modules outputs bitwise comparator 3 error indication see bit 12
TBWC4	16	rwh	TOM sub modules outputs bitwise comparator 4 error indication see bit 12
TBWC5	17	rwh	TOM sub modules outputs bitwise comparator 5 error indication see bit 12
TBWC6	18	rwh	TOM sub modules outputs bitwise comparator 6 error indication see bit 12
TBWC7	19	rwh	TOM sub modules outputs bitwise comparator 7 error indication see bit 12

Generic Timer Module (GTM)

Field	Bits	Type	Description
TBWC8	20	rwh	TOM sub modules outputs bitwise comparator 8 error indication see bit 12
TBWC9	21	rwh	TOM sub modules outputs bitwise comparator 9 error indication see bit 12
TBWC10	22	rwh	TOM sub modules outputs bitwise comparator 10 error indication see bit 12
TBWC11	23	rwh	TOM sub modules outputs bitwise comparator 11 error indication see bit 12
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

26.19.7.3 Register CMP_IRQ_EN

GTM_CMP_IRQ_EN

CMP Interrupt Enable Register

(208_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TBW C11_ EN_I RQ	TBW C10_ EN_I RQ	TBW C9_ EN_I RQ	TBW C8_ EN_I RQ	TBW C7_ EN_I RQ	TBW C6_ EN_I RQ	TBW C5_ EN_I RQ	TBW C4_ EN_I RQ
rw								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBW C3_ EN_I RQ	TBW C2_ EN_I RQ	TBW C1_ EN_I RQ	TBW C0_ EN_I RQ	ABW C11_ EN_I RQ	ABW C10_ EN_I RQ	ABW C9_ EN_I RQ	ABW C8_ EN_I RQ	ABW C7_ EN_I RQ	ABW C6_ EN_I RQ	ABW C5_ EN_I RQ	ABW C4_ EN_I RQ	ABW C3_ EN_I RQ	ABW C2_ EN_I RQ	ABW C1_ EN_I RQ	ABW C0_ EN_I RQ
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ABWC0_EN_IRQ	0	rw	Enable ABWC0 interrupt source for CMP_IRQ line 0 _B interrupt source ABWC0 is disabled 1 _B interrupt source ABWC0 is enabled
ABWC1_EN_IRQ	1	rw	Enable ABWC1 interrupt source for CMP_IRQ line see bit 0
ABWC2_EN_IRQ	2	rw	Enable ABWC2 interrupt source for CMP_IRQ line see bit 0
ABWC3_EN_IRQ	3	rw	Enable ABWC3 interrupt source for CMP_IRQ line see bit 0
ABWC4_EN_IRQ	4	rw	Enable ABWC4 interrupt source for CMP_IRQ line see bit 0
ABWC5_EN_IRQ	5	rw	Enable ABWC5 interrupt source for CMP_IRQ line see bit 0
ABWC6_EN_IRQ	6	rw	Enable ABWC6 interrupt source for CMP_IRQ line see bit 0
ABWC7_EN_IRQ	7	rw	Enable ABWC7 interrupt source for CMP_IRQ line see bit 0
ABWC8_EN_IRQ	8	rw	Enable ABWC8 interrupt source for CMP_IRQ line see bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
ABWC9_EN_IRQ	9	rw	Enable ABWC9 interrupt source for CMP_IRQ line see bit 0
ABWC10_EN_IRQ	10	rw	Enable ABWC10 interrupt source for CMP_IRQ line see bit 0
ABWC11_EN_IRQ	11	rw	Enable ABWC11 interrupt source for CMP_IRQ line see bit 0
TBWC0_EN_IRQ	12	rw	Enable TBWC0 interrupt source for CMP_IRQ line 0 _B interrupt source TBWC0 is disabled 1 _B interrupt source TBWC0 is enabled
TBWC1_EN_IRQ	13	rw	Enable TBWC1 interrupt source for CMP_IRQ line see bit 12
TBWC2_EN_IRQ	14	rw	Enable TBWC2 interrupt source for CMP_IRQ line see bit 12
TBWC3_EN_IRQ	15	rw	Enable TBWC3 interrupt source for CMP_IRQ line see bit 12
TBWC4_EN_IRQ	16	rw	Enable TBWC4 interrupt source for CMP_IRQ line see bit 12
TBWC5_EN_IRQ	17	rw	Enable TBWC5 interrupt source for CMP_IRQ line see bit 12
TBWC6_EN_IRQ	18	rw	Enable TBWC6 interrupt source for CMP_IRQ line see bit 12
TBWC7_EN_IRQ	19	rw	Enable TBWC7 interrupt source for CMP_IRQ line see bit 12
TBWC8_EN_IRQ	20	rw	Enable TBWC8 interrupt source for CMP_IRQ line see bit 12
TBWC9_EN_IRQ	21	rw	Enable TBWC9 interrupt source for CMP_IRQ line see bit 12
TBWC10_EN_IRQ	22	rw	Enable TBWC10 interrupt source for CMP_IRQ line see bit 12
TBWC11_EN_IRQ	23	rw	Enable TBWC11 interrupt source for CMP_IRQ line see bit 12
Reserved	[31:24]	rw	Reserved Read as zero, should be written as zero

26.19.7.4 Register CMP_IRQ_FORCINT

GTM_CMP_IRQ_FORCINT

CMP Interrupt Force Register

 (20C_H)

 Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TRG_TB_WC1	TRG_TB_WC1	TRG_TB_WC9	TRG_TB_WC8	TRG_TB_WC7	TRG_TB_WC6	TRG_TB_WC5	TRG_TB_WC4
								1	0						
								w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRG_TB_WC3	TRG_TB_WC2	TRG_TB_WC1	TRG_TB_WC0	TRG_AB_WC1	TRG_AB_WC1	TRG_AB_WC9	TRG_AB_WC8	TRG_AB_WC7	TRG_AB_WC6	TRG_AB_WC5	TRG_AB_WC4	TRG_AB_WC3	TRG_AB_WC2	TRG_AB_WC1	TRG_AB_WC0
				1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
TRG_ABW C0	0	w	Trigger ABWC0 bit in CMP_IRQ_NOTIFY register by software 0 _B No event triggering 1 _B Assert corresponding field in CMP_IRQ_NOTIFY register Note: This bit is cleared automatically after write. Note: This bit is write protected by bit GTM_CTRL.RF_PROT.
TRG_ABW C1	1	w	Trigger ABWC1 bit in CMP_IRQ_NOTIFY register by software see bit 0
TRG_ABW C2	2	w	Trigger ABWC2 bit in CMP_IRQ_NOTIFY register by software see bit 0
TRG_ABW C3	3	w	Trigger ABWC3 bit in CMP_IRQ_NOTIFY register by software see bit 0
TRG_ABW C4	4	w	Trigger ABWC4 bit in CMP_IRQ_NOTIFY register by software see bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG_ABW C5	5	w	Trigger ABWC5 bit in CMP_IRQ_NOTIFY register by software see bit 0
TRG_ABW C6	6	w	Trigger ABWC6 bit in CMP_IRQ_NOTIFY register by software see bit 0
TRG_ABW C7	7	w	Trigger ABWC7 bit in CMP_IRQ_NOTIFY register by software see bit 0
TRG_ABW C8	8	w	Trigger ABWC8 bit in CMP_IRQ_NOTIFY register by software see bit 0
TRG_ABW C9	9	w	Trigger ABWC9 bit in CMP_IRQ_NOTIFY register by software see bit 0
TRG_ABW C10	10	w	Trigger ABWC10 bit in CMP_IRQ_NOTIFY register by software see bit 0
TRG_ABW C11	11	w	Trigger ABWC11 bit in CMP_IRQ_NOTIFY register by software see bit 0
TRG_TBW C0	12	w	Trigger TBWC0 bit in CMP_IRQ_NOTIFY register by software 0 _B No event triggering 1 _B Assert corresponding field in CMP_IRQ_NOTIFY register Note: This bit is cleared automatically after write.
TRG_TBW C1	13	w	Trigger TBWC1 bit in CMP_IRQ_NOTIFY register by software see bit 12
TRG_TBW C2	14	w	Trigger TBWC2 bit in CMP_IRQ_NOTIFY register by software see bit 12
TRG_TBW C3	15	w	Trigger TBWC3 bit in CMP_IRQ_NOTIFY register by software see bit 12

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRG_TBWC4	16	w	Trigger TBWC4 bit in CMP_IRQ_NOTIFY register by software see bit 12
TRG_TBWC5	17	w	Trigger TBWC5 bit in CMP_IRQ_NOTIFY register by software see bit 12
TRG_TBWC6	18	w	Trigger TBWC6 bit in CMP_IRQ_NOTIFY register by software see bit 12
TRG_TBWC7	19	w	Trigger TBWC7 bit in CMP_IRQ_NOTIFY register by software see bit 12
TRG_TBWC8	20	w	Trigger TBWC8 bit in CMP_IRQ_NOTIFY register by software see bit 12
TRG_TBWC9	21	w	Trigger TBWC9 bit in CMP_IRQ_NOTIFY register by software see bit 12
TRG_TBWC10	22	w	Trigger TBWC10 bit in CMP_IRQ_NOTIFY register by software see bit 12
TRG_TBWC11	23	w	Trigger TBWC11 bit in CMP_IRQ_NOTIFY register by software see bit 12
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

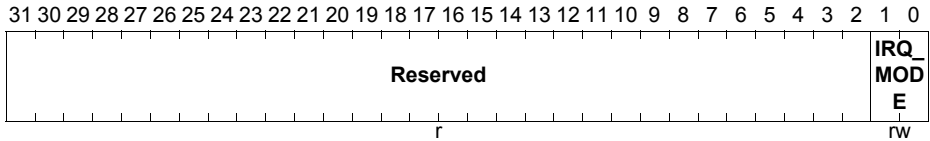
Generic Timer Module (GTM)

26.19.7.5 Register CMP_IRQ_MODE

GTM_CMP_IRQ_MODE

CMP IRQ Mode Configuration Register(210_H)

Reset Value: 00000000_H



Field	Bits	Type	Description
IRQ_MODE	[1:0]	rw	IRQ mode selection 00 _B Level mode 01 _B Pulse mode 10 _B Pulse-Notify mode 11 _B Single-Pulse mode Note: The interrupt modes are described in Section 26.2.5 .
Reserved	[31:2]	r	Reserved Read as zero, should be written as zero

26.19.7.6 Register CMP_EIRQ_EN

GTM_CMP_EIRQ_EN

CMP Error Interrupt Enable Register

(214_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TBW C11_ EN_ EIRQ	TBW C10_ EN_ EIRQ	TBW C9_ EN_ EIRQ	TBW C8_ EN_ EIRQ	TBW C7_ EN_ EIRQ	TBW C6_ EN_ EIRQ	TBW C5_ EN_ EIRQ	TBW C4_ EN_ EIRQ
rw								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBW C3_ EN_ EIRQ	TBW C2_ EN_ EIRQ	TBW C1_ EN_ EIRQ	TBW C0_ EN_ EIRQ	ABW C11_ EN_ EIRQ	ABW C10_ EN_ EIRQ	ABW C9_ EN_ EIRQ	ABW C8_ EN_ EIRQ	ABW C7_ EN_ EIRQ	ABW C6_ EN_ EIRQ	ABW C5_ EN_ EIRQ	ABW C4_ EN_ EIRQ	ABW C3_ EN_ EIRQ	ABW C2_ EN_ EIRQ	ABW C1_ EN_ EIRQ	ABW C0_ EN_ EIRQ
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ABWC0_EN_EIRQ	0	rw	enable ABWC0 interrupt source for CMP_EIRQ line 0 _B interrupt source ABWC0 is disabled 1 _B interrupt source ABWC0 is enabled
ABWC1_EN_EIRQ	1	rw	enable ABWC1 interrupt source for CMP_EIRQ line see bit 0
ABWC2_EN_EIRQ	2	rw	enable ABWC2 interrupt source for CMP_EIRQ line see bit 0
ABWC3_EN_EIRQ	3	rw	enable ABWC3 interrupt source for CMP_EIRQ line see bit 0
ABWC4_EN_EIRQ	4	rw	enable ABWC4 interrupt source for CMP_EIRQ line see bit 0
ABWC5_EN_EIRQ	5	rw	enable ABWC5 interrupt source for CMP_EIRQ line see bit 0
ABWC6_EN_EIRQ	6	rw	enable ABWC6 interrupt source for CMP_EIRQ line. see bit 0
ABWC7_EN_EIRQ	7	rw	enable ABWC7 interrupt source for CMP_EIRQ line see bit 0

Generic Timer Module (GTM)

Field	Bits	Type	Description
ABWC8_EN_EIRQ	8	rw	enable ABWC8 interrupt source for CMP_EIRQ line see bit 0
ABWC9_EN_EIRQ	9	rw	enable ABWC9 interrupt source for CMP_EIRQ line see bit 0
ABWC10_EN_EIRQ	10	rw	enable ABWC10 interrupt source for CMP_EIRQ line see bit 0
ABWC11_EN_EIRQ	11	rw	enable ABWC11 interrupt source for CMP_EIRQ line see bit 0
TBWC0_EN_EIRQ	12	rw	enable TBWC0 interrupt source for CMP_EIRQ line 0 _B interrupt source TBWC0 is disabled 1 _B interrupt source TBWC0 is enabled
TBWC1_EN_EIRQ	13	rw	enable TBWC1 interrupt source for CMP_EIRQ line see bit 12
TBWC2_EN_EIRQ	14	rw	enable TBWC2 interrupt source for CMP_EIRQ line see bit 12
TBWC3_EN_EIRQ	15	rw	enable TBWC3 interrupt source for CMP_EIRQ line see bit 12
TBWC4_EN_EIRQ	16	rw	enable TBWC4 interrupt source for CMP_EIRQ line see bit 12
TBWC5_EN_EIRQ	17	rw	enable TBWC5 interrupt source for CMP_EIRQ line see bit 12
TBWC6_EN_EIRQ	18	rw	enable TBWC6 interrupt source for CMP_EIRQ line see bit 12
TBWC7_EN_EIRQ	19	rw	enable TBWC7 interrupt source for CMP_EIRQ line see bit 12
TBWC8_EN_EIRQ	20	rw	enable TBWC8 interrupt source for CMP_EIRQ line see bit 12
TBWC9_EN_EIRQ	21	rw	enable TBWC9 interrupt source for CMP_EIRQ line see bit 12
TBWC10_EN_EIRQ	22	rw	enable TBWC10 interrupt source for CMP_EIRQ line see bit 12
TBWC11_EN_EIRQ	23	rw	enable TBWC11 interrupt source for CMP_EIRQ line see bit 12
Reserved	[31:24]	rw	Reserved Should be written as zero

26.20 Monitor Unit (MON)

26.20.1 Overview

The Monitor Unit (MON) is designed for the use in safety relevant applications. The main idea is to have a possibility to supervise common used circuitry and resources. In this way the activity of the clocks is supervised. In addition the characteristics of output signals can be checked in a MCS channel by a re-read-in via TIM and routing to the MCS. When the comparison fails an error signal is generated in MCS and sent to the monitor unit. One error signal per MCS summarizes the errors of all channels. By generating of an activity signal per channel for each such performed comparison, the activity of TIM, ARU and the used clocks is checked implicitly.

In addition the ARU cycle time could be also compared in a MCS channel to given values.

26.20.1.1 MON Block Diagram

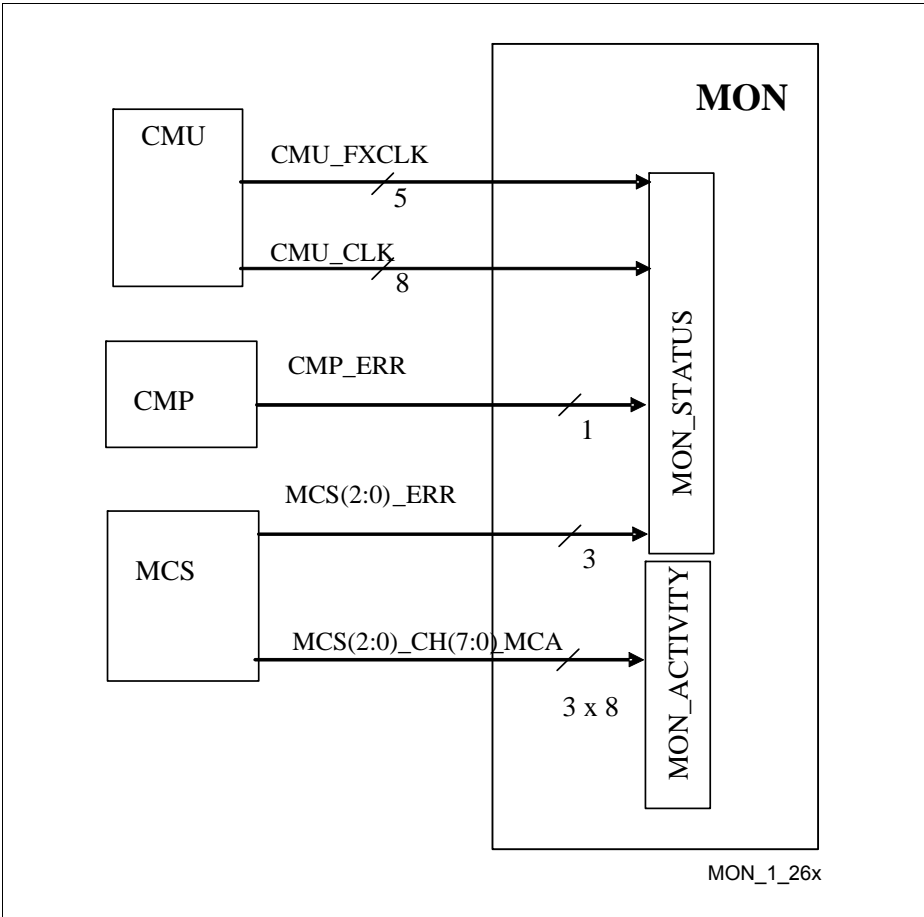


Figure 26-84 MON Block Diagram

26.20.1.2 Realization without Activity Checker of the clock signals

An activity checker of the clock signals used is not needed because these signals are only enables to be used in combination with the system clock. Therefore the clock enables are to be checked to have a high value.

26.20.2 Clock Monitoring

The monitor unit has a connection to each of the 8 clocks `CMU_CLK[x]` ($x=0\dots7$), provided by the CMU. Some of these clocks can be used for special tasks (see [Section 26.3](#)).

In addition the 5 clock inputs of the TOMs `CMU_FXCLK[y]` ($y=0\dots4$) are also connected to the MON unit.

The supervising of the clocks is done by scanning for activity of each clock.

A high value is defined as the state to be monitored.

When a high value of the clock enable is detected, the corresponding bit in the status register `MON_STATUS` is set.

The status register bits are reset by writing a one.

When the register is polled by the CPU and the time between two read accesses is higher than the period of the slowest clock, all bits of the corresponding clocks must have been set.

When polling in shorter time distances, not for all clocks an activity can be shown, although they are still working.

Because of the realization without a select register for the clock signals only the bits of the status register are to be considered for which the clock signal is enabled in the CMU.

26.20.3 CMP error Monitoring

The signal `CMP_ERR` is to be received directly from module `CMP` and is set if an error occurred.

26.20.4 Checking the Characteristics of Signals by MCS

By use of the MCS some given properties of signals can be checked. Such signals can be generated output signals of TOM or ATOM channels, which are re-read into a TIM and the time stamp information is routed via ARU to the MCS module.

The corresponding MCS signal performs the check according to given properties. In this way signal high or low time as well as signal periods can be checked, also taking into account tolerances. When the check fails a MCS internal error signal is generated and ORed with the error signals of the other channels of the MCS module to an summarized error signal `MCS[z]_ERR` ($z=0\dots235$).

For each MCS a summarized error signal is transmitted to MON and monitored in the `MON_STATUS` register.

In order to check the execution of the comparison for each MCS channel an activity signal is generated. In the `MCA_i_x` ($x=0\dots7$) vector always for each MCS 8 bits for each channels are combined. The activity signals are stored in the `MON_ACTIVITY_0` register

Generic Timer Module (GTM)

for MCS0 to MCS3. The bits are set by a one signal and reset by writing a one to it (preferably after polling the status of the register).

Because the activity signal shows the execution of a comparison, the involved units for providing the signals and execution of comparison (like TIM, ARU and MCS itself) are checked implicitly to work accordingly. Also the involved clocks and time bases are checked in this way.

26.20.5 Checking ARU Cycle Time

The cycle time of the ARU can be checked, when this is essential for safety purposes. This check can be performed by an MCS channel. It should be noted that the MCS program for measuring the ARU round trip time must add a tolerance value.

The resulting error is reported to the MON unit using the summarized error signal MCS[z]_ERR for each MCS module in addition to an interrupt, generated in MCS. The same signals and status bits are used as in the case of checking the signal characteristics.

The corresponding MCS is programmed to get a fixed data value at address 0x1FF. The data value is always zero and is not blocked. When getting the access the time stamp value TBU_TS0 is stored in a register. The next time getting the access the new TBU_TS0 value is stored and the difference between both values is compared with a given value. When the comparison fails, an error flag is set in the MCS internal status register, an interrupt is generated and the error signal MCS[z]_ERR is provided.

When the check is performed, an activity signal MCA_x (x=0...31) is provided for each channel x for each MCSi (i=0...6) together with an summarized interrupt MCS[z]_ERR for each MCS.

The activity signal sets a bit in the MON_ACTIVITY register.

The bits in the MON_ACTIVITY register are reset by writing a one.

When the check fails, an interrupt is generated and the error signal MCS[z]_ERR is provided for the MON unit.

Figure 26-84 shows the block diagram of the Monitor Unit.

26.20.6 MON Interrupt Signals

The MON submodule has no interrupt signals.

26.20.7 MON Registers Overview

Following configuration registers are considered in MON sub module

Table 26-61 MON Registers Overview

Register Name	Description	Details in Section
MON_STATUS	Monitor Status register	Section 26.20.8.1
MON_ACTIVITY_0	Monitor activity register 0	Section 26.20.8.2

26.20.8 MON Configuration Registers Description

All of the following registers are 32-bit only accessible.

26.20.8.1 Register MON_STATUS

GTM_MON_STATUS

Monitor Status Register

(180_H)

Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									MCS 2_E RR	MCS 1_E RR	MCS 0_E RR	Reserved			CM P _E R R
r									r	r	r	r			r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ACT _CM _UFX 4	ACT _CM _UFX 3	ACT _CM _UFX 2	ACT _CM _UFX 1	ACT _CM _UFX 0	ACT _CM _U7	ACT _CM _U6	ACT _CM _U5	ACT _CM _U4	ACT _CM _U3	ACT _CM _U2	ACT _CM _U1	ACT _CM _U0
r			rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
ACT_CMU 0	0	rwh	CMU_CLK0 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ACT_CMU 1	1	rwh	CMU_CLK1 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ACT_CMU 2	2	rwh	CMU_CLK2 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ACT_CMU 3	3	rwh	CMU_CLK3 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ACT_CMU 4	4	rwh	CMU_CLK4 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ACT_CMU 5	5	rwh	CMU_CLK5 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.

Generic Timer Module (GTM)

Field	Bits	Type	Description
ACT_CMU 6	6	rwh	CMU_CLK6 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ACT_CMU 7	7	rwh	CMU_CLK7 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ACT_CMU FX0	8	rwh	CMU_CLKFX0 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ACT_CMU FX1	9	rwh	CMU_CLKFX1 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ACT_CMU FX2	10	rwh	CMU_CLKFX2 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ACT_CMU FX3	11	rwh	CMU_CLKFX3 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
ACT_CMU FX4	12	rwh	CMU_CLKFX4 activity Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged. Note: Bits 0 to 12 are set, when a high low slope is detected at the considered clock
Reserved	[15:13]	r	Reserved Read as zero, should be written as zero
CMP_ERR	16	r	Error detected at CMP Note: This bit will be readable only.
Reserved	[19:17]	r	Reserved Read as zero, should be written as zero
MCS0_ER R	20	r	Error detected at MCS[0] Note: This bit will be readable only.
MCS1_ER R	21	r	Error detected at MCS[1] Note: This bit will be readable only.
MCS2_ER R	22	r	Error detected at MCS[2] Note: This bit will be readable only.

Generic Timer Module (GTM)

Field	Bits	Type	Description
Reserved	[31:23]	r	Reserved Read as zero, should be written as zero Note: Bits16 and 20 to 22 are set, when the corresponding unit reports an error Note: The MCS can be programmed to generate an error, when the comparison of signal values (duty time, cycle time) fails or also when the cycle time of the ARU (checking of the TBU_TS0 between two periodic accesses) is out of the expected range.

26.20.8.2 Register MON_ACTIVITY_0
GTM_MON_ACTIVITY_0
Monitor Activity Register 0
(184_H)
Reset Value: 00000000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								MCA	MCA	MCA	MCA	MCA	MCA	MCA	MCA
								_2_7	_2_6	_2_5	_2_4	_2_3	_2_2	_2_1	_2_0
r								rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCA	MCA	MCA	MCA	MCA	MCA	MCA	MCA	MCA	MCA	MCA	MCA	MCA	MCA	MCA	MCA
_1_7	_1_6	_1_5	_1_4	_1_3	_1_2	_1_1	_1_0	_0_7	_0_6	_0_5	_0_4	_0_3	_0_2	_0_1	_0_0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
MCA_0_0	0	rwh	Activity of check performed in module MCS0 at channel 0 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_0_1	1	rwh	Activity of check performed in module MCS0 at channel 1 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_0_2	2	rwh	Activity of check performed in module MCS0 at channel 2 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.

Generic Timer Module (GTM)

Field	Bits	Type	Description
MCA_0_3	3	rwh	Activity of check performed in module MCS0 at channel 3 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_0_4	4	rwh	Activity of check performed in module MCS0 at channel 4 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_0_5	5	rwh	Activity of check performed in module MCS0 at channel 5 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_0_6	6	rwh	Activity of check performed in module MCS0 at channel 6 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_0_7	7	rwh	Activity of check performed in module MCS0 at channel 7 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_1_0	8	rwh	Activity of check performed in module MCS1 at channel 0 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_1_1	9	rwh	Activity of check performed in module MCS1 at channel 1 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_1_2	10	rwh	Activity of check performed in module MCS1 at channel 2 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_1_3	11	rwh	Activity of check performed in module MCS1 at channel 3 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.

Generic Timer Module (GTM)

Field	Bits	Type	Description
MCA_1_4	12	rwh	Activity of check performed in module MCS1 at channel 4 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_1_5	13	rwh	Activity of check performed in module MCS1 at channel 5 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_1_6	14	rwh	Activity of check performed in module MCS1 at channel 6 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_1_7	15	rwh	Activity of check performed in module MCS1 at channel 7 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_2_0	16	rwh	Activity of check performed in module MCS2 at channel 0 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_2_1	17	rwh	Activity of check performed in module MCS2 at channel 1 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_2_2	18	rwh	Activity of check performed in module MCS2 at channel 2 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_2_3	19	rwh	Activity of check performed in module MCS2 at channel 3 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_2_4	20	rwh	Activity of check performed in module MCS2 at channel 4 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.

Generic Timer Module (GTM)

Field	Bits	Type	Description
MCA_2_5	21	rwh	Activity of check performed in module MCS2 at channel 5 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_2_6	22	rwh	Activity of check performed in module MCS2 at channel 6 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
MCA_2_7	23	rwh	Activity of check performed in module MCS2 at channel 7 Note: This bit will be cleared on a CPU write access of value '1'. A read access leaves the bit unchanged.
Reserved	[31:24]	r	Reserved Read as zero, should be written as zero

26.21 Appendix
26.21.1 ARU Write Address Overview
Table 26-62 ARU Write Address Table

Name	Address		Name	Address
ARU_ACCESS	0x000		ATOM [0...3]	
TIM [0...2]			ATOM0_WRA DDR[0...7]	0x11F...0x126
TIM0_WRADDR [0...7]	0x001...0x008		ATOM1_WRA DDR[0...7]	0x127...0x12E
TIM1_WRADDR [0...7]	0x009...0x010		ATOM2_WRA DDR[07]	0x12F0x136
TIM2_WRADDR [07]	0x0110x018		ATOM3_WRA DDR[07]	0x1370x13E
unused	0x0190x020		unused	0x13F0x146
unused	0x0210x028		unused	0x1470x14E
unused	0x0290x030		unused	0x14F0x156
unused	0x0310x038		unused	0x1570x15E
DPLL			unused	0x15F0x166
DPLL_WRADD R[023]	0x0390x050		unused	0x1670x16E
F2A [0]			unused	0x16F0x176
F2A0_WRADD R[07]	0x0510x058		unused	0x1770x17E
unused	0x0590x060		misc	
BRC			unused	0x17F0x1FD
BRC_WRADDR [021]	0x0610x076		ARU_EMPTY_ ADDR	0x1FE
MCS [0 2]			ARU_FULL_A DDR	0x1FF
MCS0_WRADD R[023]	0x0770x08E			

Generic Timer Module (GTM)

Table 26-62 ARU Write Address Table (cont'd)

Name	Address		Name	Address
MCS1_WRADDR[023]	0x08F0x0A6			
MCS2_WRADDR[023]	0x0A70x0BE			
unused	0x0BF0x0D6			
unused	0x0D70x0EE			
unused	0x0EF0x106			
unused	0x1070x11E			

Generic Timer Module (GTM)
26.22 GTM Implementation

This chapter describes product specific implementation of the Generic Timer Module (GTM).

Note: After a module reset the outputs of the (A)TOMs are high (see (A)TOMi_CHx_CRTL.SL) and could cause unexpected triggers in other modules.

26.22.1 GTM Registers
Table 26-63 Registers Address Space

Module	Base Address	End Address	Note
GTM	F010 0000 _H	F019 FFFF _H	

Table 26-64 Registers Overview - GTM Control Registers

Short Name	Description	Offset Addr.	Access Mode		Reset	Description See
			Read	Write		
CLC	Clock Control Register	9FD00 _H	U, SV	SV, E, P	Application	Page 26-73 5
TIM0INSEL	TIM0 Input Select Register	9FD10 _H	U, SV	U, SV, P	Application	Page 26-75 6
TIM1INSEL	TIM1 Input Select Register	9FD14 _H	U, SV	U, SV, P	Application	Page 26-75 6
TIM2INSEL	TIM2 Input Select Register	9FD18 _H	U, SV	U, SV, P	Application	Page 26-75 6
TOUTSEL0	Timer Output Select 0 Register	9FD30 _H	U, SV	U, SV, P	Application	Page 26-78 7
TOUTSEL1	Timer Output Select 1 Register	9FD34 _H	U, SV	U, SV, P	Application	Page 26-78 7
TOUTSEL2	Timer Output Select 2 Register	9FD38 _H	U, SV	U, SV, P	Application	Page 26-78 7
TOUTSEL3	Timer Output Select 3 Register	9FD3C _H	U, SV	U, SV, P	Application	Page 26-78 7
TOUTSEL4	Timer Output Select 4 Register	9FD40 _H	U, SV	U, SV, P	Application	Page 26-78 7
TOUTSEL5	Timer Output Select 5 Register	9FD44 _H	U, SV	U, SV, P	Application	Page 26-78 7

Generic Timer Module (GTM)
Table 26-64 Registers Overview - GTM Control Registers

Short Name	Description	Offset Addr.	Access Mode		Reset	Description See
			Read	Write		
TOUTSEL6	Timer Output Select 6 Register	9FD48 _H	U, SV	U, SV, P	Application	Page 26-78 7
TOUTSEL7	Timer Output Select 7 Register	9FD4C _H	U, SV	U, SV, P	Application	Page 26-78 7
TOUTSEL8	Timer Output Select 8 Register	9FD50 _H	U, SV	U, SV, P	Application	Page 26-78 7
TOUTSEL9	Timer Output Select 9 Register	9FD54 _H	U, SV	U, SV, P	Application	Page 26-78 7
MSCSET1CON0	MSC Set1 Control 0 Register	9FF00 _H	U, SV	U, SV, P	Application	Page 26-79 1
MSCSET1CON1	MSC Set1 Control 1 Register	9FF04 _H	U, SV	U, SV, P	Application	Page 26-79 2
MSCSET1CON2	MSC Set1 Control 2 Register	9FF08 _H	U, SV	U, SV, P	Application	Page 26-79 3
MSCSET1CON3	MSC Set1 Control 3 Register	9FF0C _H	U, SV	U, SV, P	Application	Page 26-79 4
MSCSET2CON0	MSC Set2 Control 0 Register	9FF10 _H	U, SV	U, SV, P	Application	Page 26-79 5
MSCSET2CON1	MSC Set2 Control 1 Register	9FF14 _H	U, SV	U, SV, P	Application	Page 26-79 6
MSCSET2CON2	MSC Set2 Control 2 Register	9FF18 _H	U, SV	U, SV, P	Application	Page 26-79 7
MSCSET2CON3	MSC Set2 Control 3 Register	9FF14C _H	U, SV	U, SV, P	Application	Page 26-79 8
MSCSET3CON0	MSC Set3 Control 0 Register	9FF20 _H	U, SV	U, SV, P	Application	Page 26-79 9
MSCSET3CON1	MSC Set3 Control 1 Register	9FF24 _H	U, SV	U, SV, P	Application	Page 26-80 0
MSCSET3CON2	MSC Set3 Control 2 Register	9FF28 _H	U, SV	U, SV, P	Application	Page 26-80 1
MSCSET3CON3	MSC Set3 Control 3 Register	9FF2C _H	U, SV	U, SV, P	Application	Page 26-80 2

Generic Timer Module (GTM)
Table 26-64 Registers Overview - GTM Control Registers

Short Name	Description	Offset Addr.	Access Mode		Reset	Description See
			Read	Write		
MSCSET4 CON0	MSC Set4 Control 0 Register	9FF30 _H	U, SV	U, SV, P	Application	Page 26-80 3
MSCSET4 CON1	MSC Set4 Control 1 Register	9FF34 _H	U, SV	U, SV, P	Application	Page 26-80 4
MSCSET4 CON2	MSC Set4 Control 2 Register	9FF38 _H	U, SV	U, SV, P	Application	Page 26-80 5
MSCSET4 CON3	MSC Set4 Control 3 Register	9FF3C _H	U, SV	U, SV, P	Application	Page 26-80 6
MSC0INL CON	MSC0 Input Low Control Register	9FF60 _H	U, SV	U, SV, P	Application	Page 26-80 7
MSC0INH CON	MSC0 Input High Control Register	9FF64 _H	U, SV	U, SV, P	Application	Page 26-80 7
MSC1INL CON	MSC1 Input Low Control Register	9FF68 _H	U, SV	U, SV, P	Application	Page 26-80 9
MSC1INH CON	MSC1 Input High Control Register	9FF6C _H	U, SV	U, SV, P	Application	Page 26-80 9
MSC0INL EXTCON	MSC0 Input Low Extended Control Register	9FF70 _H	U, SV	U, SV, P	Application	Page 26-80 8
DSADCIN SEL0	DSADC Input Select 0 Register	9FD7C _H	U, SV	U, SV, P	Application	Page 26-81 1
DSADCIN SEL1	DSADC Input Select 1 Register	9FD80 _H	U, SV	U, SV, P	Application	Page 26-81 5
DSADCIN SEL2	DSADC Input Select 2 Register	9FD84 _H	U, SV	U, SV, P	Application	Page 26-82 0
DSADCO UTSEL00	DSADC Output Select 00 Register	9FD88 _H	U, SV	U, SV, P	Application	Page 26-82 2
DSADCO UTSEL10	DSADC Output Select 10 Register	9FD90 _H	U, SV	U, SV, P	Application	Page 26-82 4
ADCTRIG 0OUT0	ADC Trigger 0 Output 0 Register	9FDB0 _H	U, SV	U, SV, P	Application	Page 26-82 7
ADCTRIG 1OUT0	ADC Trigger 1 Output 0 Register	9FDB8 _H	U, SV	U, SV, P	Application	Page 26-82 8

Generic Timer Module (GTM)
Table 26-64 Registers Overview - GTM Control Registers

Short Name	Description	Offset Addr.	Access Mode		Reset	Description See
			Read	Write		
CANOUTS EL	CAN Output Select Register	9FDA0 _H	U, SV	U, SV, P	Application	Page 26-829
PSI5OUTS EL0	PSI5 Output Select 0 Register	9FDA4 _H	U, SV	U, SV, P	Application	Page 26-835
PSI5SOUT SEL0	PSI5-S Output Select 0 Register	9FDA8 _H	U, SV	U, SV, P	Application	Page 26-836
OTBU0T	OCDS TBU0 Trigger Register	9FDC4 _H	U, SV	U, SV, P	Debug	Page 26-863
OTBU1T	OCDS TBU1 Trigger Register	9FDC8 _H	U, SV	U, SV, P	Debug	Page 26-863
OTBU2T	OCDS TBU2 Trigger Register	9FDCC _H	U, SV	U, SV, P	Debug	Page 26-864
OTSS	OCDS Trigger Set Select Register	9FDD0 _H	U, SV	U, SV, P	Debug	Page 26-858
OTSC0	OCDS Trigger Set Control 0 Register	9FDD4 _H	U, SV	U, SV, P	Debug	Page 26-859
OTSC1	OCDS Trigger Set Control 1 Register	9FDD8 _H	U, SV	U, SV, P	Debug	Page 26-861
ODA	OCDS Debug Access Register	9FDDC _H	U, SV	U, SV, P	Debug	Page 26-862
OCS	OCDS Control and Status Register	9FDE8 _H	U, SV	U, SV, P	Debug	Page 26-736
KRSTCLR	Reset Status Clear Register	9FDEC _H	U, SV	SV, E, P	Application	Page 26-740
KRST1	Reset Control Register 1	9FDF0 _H	U, SV	SV, E, P	Application	Page 26-740
KRST0	Reset Control Register 0	9FDF4 _H	U, SV	SV, E, P	Application	Page 26-739
ACCEN1	Access Enable Register 1	9FDF8 _H	U, SV	SV, SE	Application	Page 26-738
ACCEN0	Access Enable Register 0	9FDFC _H	U, SV	SV, SE	Application	Page 26-737

GTM Memory Addresses
Table 26-65 GTM Memories

Memory	Words	Word Size	Start Address Offset	End Address Offset
FIFO0 RAM	1024	29 Bit	0x19000	0x19FFF
DPLL RAM1A	96	24 Bit	0x28200	0x2837F ¹⁾
DPLL RAM1B	128	24 Bit	0x28400	0x285FF
DPLL RAM1C	256	24 Bit	0x28600	0x289FF
DPLL RAM2	1024	24 Bit	0x2C000	0x2CFFF
MCS0 RAM0	1024	32 Bit	0x38000	0x38FFF
MCS0 RAM1	512	32 Bit	0x39000	0x397FF
MCS1 RAM0	1024	32 Bit	0x40000	0x40FFF
MCS1 RAM1	512	32 Bit	0x41000	0x417FF
MCS2 RAM0	1024	32 Bit	0x48000	0x48FFF
MCS2 RAM1	512	32 Bit	0x49000	0x497FF

1) accesses to address range 0x28380 to 0x283FF cause access errors.

GTM Address Ordering

The GTM address area start at F010 0000_H and ends at F019 FFFF_H. All given address for registers (beside for the DPLL part) are offset addresses to the GTM base address of F010 0000_H and need to be added. DPLL addresses for the CPU access are offset addresses to the DPLL base address of 0x00028000_H, independent if with address is located in a DPLL RAM or not. For the DPLL base address of course the GTM base address have to be added.

Table 26-66 Address Overview

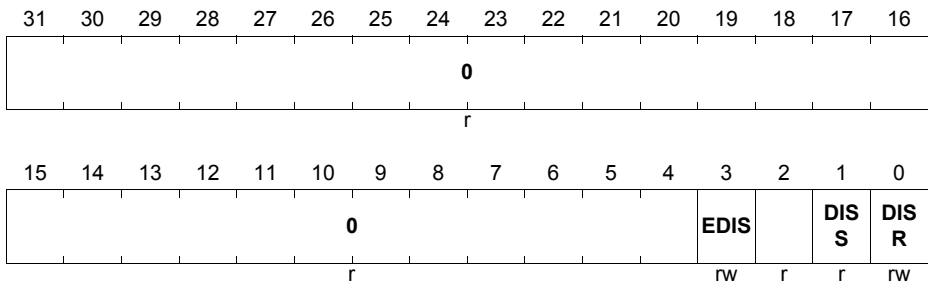
Submodule	Base address (to be added to GTM based address)
BRIDGE	0x00000030
MAP	0x00000F00
MCFG	0x00000F40
TBU	0x00000100
MON	0x00000180
CMP	0x00000200

Generic Timer Module (GTM)
Table 26-66 Address Overview (cont'd)

Submodule	Base address (to be added to GTM based address)
ARU	0x00000280
CMU	0x00000300
BRC	0x00000400
ICM	0x00000600
SPE0	0x00000800
SPE1	0x00000880
TIM0	0x00001000
TIM1	0x00001800
TIM2	0x00002000
TOM0	0x00008000
TOM1	0x00008800
ATOM0	0x0000D000
ATOM1	0x0000D800
ATOM2	0x0000E000
ATOM3	0x0000E800
F2A0	0x00018000
AFD0	0x00018080
AFD1	0x0001C080
FIFO0	0x00018400
FIFO0_MEMORY	0x00019000
DPLL	0x00028000
DPLL_RAM1A	0x00028200
DPLL_RAM1BC	0x00028400
DPLL_RAM2	0x0002C000
MCS0	0x00030000
MCS0_MEMORY	0x00038000
MCS1	0x00031000
MCS1_MEMORY	0x00040000
MCS2	0x00032000
MCS2_MEMORY	0x00048000

Generic Timer Module (GTM)

The clock control register is used to switch the GTM on or off and to control its input clock rate. The GTM can be disabled by setting bit DISR to 1.

CLC
Clock Control Register
(9FD00_H)
Reset Value: 0000 0003_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the GTM module. 0 _B No disable requested 1 _B Disable requested
DISS	1	r	Module Disable Status Bit Bit indicates the current status of the GTM module. 0 _B GTM module is enabled 1 _B GTM module is disabled
EDIS	3	rw	Sleep Mode Enable Control Used for module sleep mode control.
0	2, [31:4]	r	Reserved Read as 0; should be written with 0.

OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset and by each System Reset when OCDS is disabled. It is not touched by System Reset when OCDS is enabled.

The OCS register includes the module related control bits for the ODDS Trigger Bus (OTGB).

The OCS control register bits are only effective while the system is in debug mode. While not in debug mode, OCS reset values/modes are effective.

Write access is 32 bit wide only and requires Supervisor Mode.

Generic Timer Module (GTM)

OCS

OCDS Control and Status

 (9FDE8_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUS STA	SUS _P	SUS				0							
r		rh	w	rw				r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
								r							

Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately. In Hard suspend no registers beside the registers documented in the Implementation section of this chapter could be read or written. 2 _H Soft suspend (GTM Halt Mode). In Soft suspend registers could be read or written, for details see Chapter 26.2.6 . others , reserved
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:0], [31:30]	r	Reserved Read as 0; must be written with 0.

The Access Enable Register 0 controls write access for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Generic Timer Module (GTM)

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, EN31 -> TAG ID 011111_B.

All registers and memories of the GTM are protected beside the following registers:
ACCEN0 and ACCEN1.

ACCEN0
Access Enable Register 0
(9FDFC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

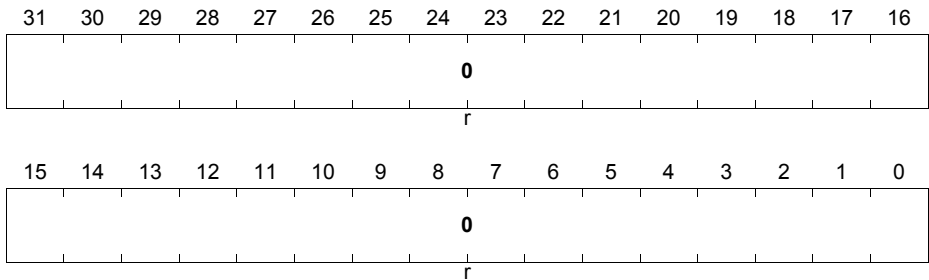
Field	Bits	Type	Description
ENn (n = 0-31)	n	rW	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register (ACCEN1)

The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID master peripheral mapping).

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000_B, EN1 -> TAG ID 100001_B, EN31 -> TAG ID 111111_B.

All registers and memories of the GTM are protected beside the following registers:
ACCEN0 and ACCEN1.

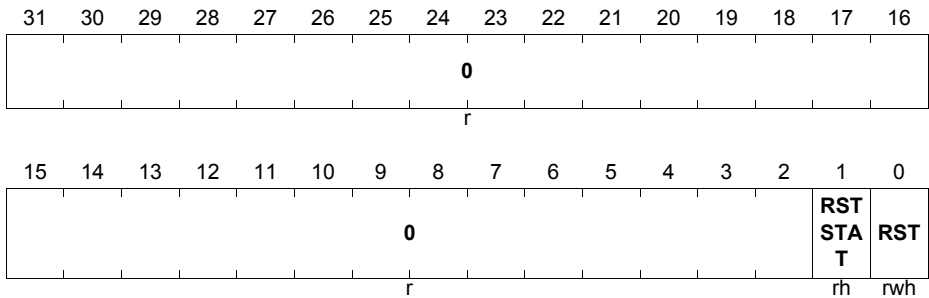
Generic Timer Module (GTM)
ACCEN1
Access Enable Register 1
(9FDF8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

The Kernel Reset Register 0 is used to reset the related module kernel. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers related to the module kernel that should be reset (kernel 0 or kernel 1). In order support modules with two kernel the BPI_FPI provides two set of kernel reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing to it with '1'.

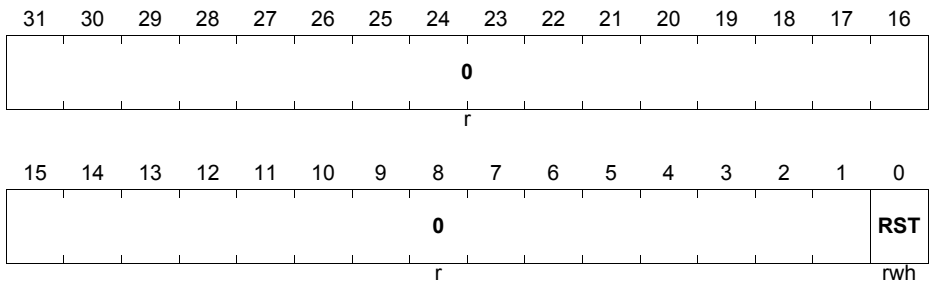
Note: This reset function has the effect as bit RST.RST.

Generic Timer Module (GTM)
KRST0
Kernel Reset Register 0
(9FDF4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
RSTSTAT	1	rw	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits. 0 _B No kernel reset was executed 1 _B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

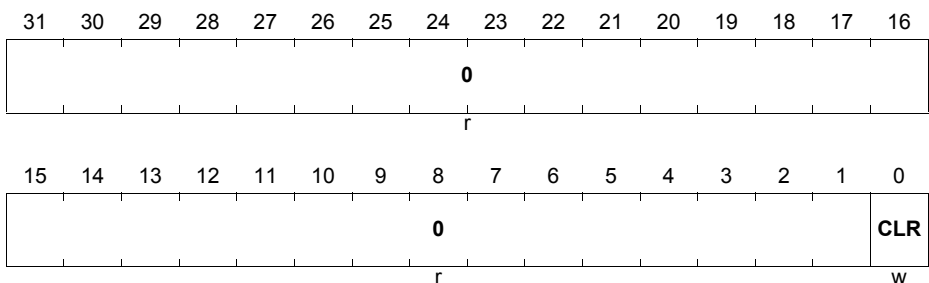
Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the GTM kernel. GTM kernel registers related to the Debug Reset (Class 1) are not influenced. To reset the GTM kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be cleared with the end of the BPI kernel reset sequence.

Generic Timer Module (GTM)
KRST1
Kernel Reset Register 1
(9FDF0_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. 0_B No kernel reset was requested 1_B A kernel reset was requested The RST bit will be cleared (re-set to '0') after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

The Kernel Reset Register Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

KRSTCLR
Kernel Reset Status Clear Register (9FDEC_H)
Reset Value: 0000 0000_H


Generic Timer Module (GTM)

Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

26.22.2 Port Connections

The sections summarize the port connections which defined the boundary of the GTM for the TC26x B-Step.

Generic Timer Module (GTM)
Table 26-67 GTM to Port Mapping for QFP-144

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P00.0	TIN9	TOUT9	TIM2_0	TIM0_0	TOM0_8	TOM1_0	ATOM_0_0	ATOM_1_0
P00.1	TIN10	TOUT10	TIM2_1	TIM0_1	TOM0_9	TOM1_1	ATOM_0_1	ATOM_1_1
P00.2	TIN11	TOUT11	TIM2_1	TIM0_1	TOM0_9	TOM1_1	ATOM_0_1	ATOM_1_1
P00.3	TIN12	TOUT12	TIM2_2	TIM1_2	TOM0_10	TOM1_2	ATOM_0_2	ATOM_1_2
P00.4	TIN13	TOUT13	TIM2_3	TIM1_3	TOM0_11	TOM1_3	ATOM_0_3	ATOM_1_3
P00.5	TIN14	TOUT14	TIM2_4	TIM1_4	TOM0_12	TOM1_4	ATOM_0_4	ATOM_1_4
P00.6	TIN15	TOUT15	TIM2_5	TIM1_5	TOM0_13	TOM1_5	ATOM_0_5	ATOM_1_5
P00.7	TIN16	TOUT16	TIM2_6	TIM1_6	TOM0_14	TOM1_6	ATOM_0_6	ATOM_1_6
P00.8	TIN17	TOUT17	TIM2_7	TIM1_7	TOM0_15	TOM1_7	ATOM_0_7	ATOM_1_7
P00.9	TIN18	TOUT18	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM_2_0	ATOM_3_0
P00.12	TIN21	TOUT21	TIM0_3	TIM1_3	TOM0_3	TOM1_3	ATOM_2_3	ATOM_3_3
P02.0	TIN0	TOUT0	TIM0_0	TIM1_0	TOM0_8	TOM1_8	ATOM_0_0	ATOM_1_0
P02.1	TIN1	TOUT1	TIM0_1	TIM1_1	TOM0_9	TOM1_9	ATOM_0_1	ATOM_1_1
P02.2	TIN2	TOUT2	TIM0_2	TIM1_2	TOM0_10	TOM1_10	ATOM_0_2	ATOM_1_2
P02.3	TIN3	TOUT3	TIM0_3	TIM1_3	TOM0_11	TOM1_11	ATOM_0_3	ATOM_1_3

Generic Timer Module (GTM)
Table 26-67 GTM to Port Mapping for QFP-144

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P02.4	TIN4	TOUT4	TIM0_4	TIM1_4	TOM0_12	TOM1_12	ATOM 0_4	ATOM 1_4
P02.5	TIN5	TOUT5	TIM0_5	TIM1_5	TOM0_13	TOM1_13	ATOM 0_5	ATOM 1_5
P02.6	TIN6	TOUT6	TIM0_6	TIM1_6	TOM0_14	TOM1_14	ATOM 0_6	ATOM 1_6
P02.7	TIN7	TOUT7	TIM0_7	TIM1_7	TOM0_15	TOM1_15	ATOM 0_7	ATOM 1_7
P02.8	TIN8	TOUT8	TIM2_0	TIM0_0	TOM0_8	TOM1_0	ATOM 0_0	ATOM 1_0
P10.1	TIN103	TOUT103	TIM0_1	TIM1_1	TOM0_1	TOM1_9	ATOM 1_1	ATOM 0_1
P10.2	TIN104	TOUT104	TIM0_2	TIM1_2	TOM0_2	TOM1_10	ATOM 1_2	ATOM 0_2
P10.3	TIN105	TOUT105	TIM0_3	TIM1_3	TOM0_3	TOM1_11	ATOM 1_3	ATOM 0_3
P10.5	TIN107	TOUT107	TIM0_2	TIM1_2	TOM0_2	TOM1_10	ATOM 1_2	ATOM 0_2
P10.6	TIN108	TOUT108	TIM0_3	TIM1_3	TOM0_3	TOM1_11	ATOM 1_3	ATOM 0_3
P11.2	TIN95	TOUT95	TIM2_1	TIM1_1	TOM0_8	TOM1_1	ATOM 2_1	ATOM 3_1
P11.3	TIN96	TOUT96	TIM2_2	TIM1_2	TOM0_10	TOM1_2	ATOM 2_2	ATOM 3_2
P11.6	TIN97	TOUT97	TIM2_3	TIM1_3	TOM0_11	TOM1_3	ATOM 2_3	ATOM 3_3
P11.9	TIN98	TOUT98	TIM2_4	TIM1_4	TOM0_12	TOM1_4	ATOM 2_4	ATOM 3_4
P11.10	TIN99	TOUT99	TIM2_5	TIM1_5	TOM0_13	TOM1_5	ATOM 2_5	ATOM 3_5
P11.11	TIN100	TOUT100	TIM2_6	TIM1_6	TOM0_14	TOM1_6	ATOM 2_6	ATOM 3_6

Generic Timer Module (GTM)
Table 26-67 GTM to Port Mapping for QFP-144

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P11.12	TIN101	TOUT101	TIM2_7	TIM1_7	TOM0_15	TOM1_7	ATOM 2_7	ATOM 3_7
P13.0	TIN91	TOUT91	TIM2_5	TIM1_5	TOM0_5	TOM1_5	ATOM 2_5	ATOM 3_5
P13.1	TIN92	TOUT92	TIM2_6	TIM1_6	TOM0_6	TOM1_6	ATOM 2_6	ATOM 3_6
P13.2	TIN93	TOUT93	TIM2_7	TIM1_7	TOM0_7	TOM1_7	ATOM 2_7	ATOM 3_7
P13.3	TIN94	TOUT94	TIM2_0	TIM1_0	TOM0_8	TOM1_0	ATOM 2_0	ATOM 3_0
P14.0	TIN80	TOUT80	TIM0_3	TIM1_3	TOM0_3	TOM1_3	ATOM 1_2	ATOM 3_2
P14.1	TIN81	TOUT81	TIM0_4	TIM1_4	TOM0_4	TOM1_4	ATOM 0_4	ATOM 3_4
P14.2	TIN82	TOUT82	TIM0_5	TIM1_5	TOM0_5	TOM1_5	ATOM 0_3	ATOM 3_3
P14.3	TIN83	TOUT83	TIM0_6	TIM1_6	TOM0_6	TOM1_6	ATOM 0_2	ATOM 3_2
P14.4	TIN84	TOUT84	TIM0_7	TIM1_7	TOM0_7	TOM1_7	ATOM 0_1	ATOM 3_1
P14.5	TIN85	TOUT85	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM 0_0	ATOM 3_0
P14.6	TIN86	TOUT86	TIM0_1	TIM1_1	TOM0_1	TOM1_1	ATOM 1_1	ATOM 3_1
P15.0	TIN71	TOUT71	TIM2_3	TIM1_3	TOM1_3	TOM0_11	ATOM 1_3	ATOM 2_3
P15.1	TIN72	TOUT72	TIM2_4	TIM1_4	TOM1_4	TOM0_12	ATOM 1_4	ATOM 2_4
P15.2	TIN73	TOUT73	TIM2_5	TIM1_5	TOM1_5	TOM0_13	ATOM 1_5	ATOM 2_5
P15.3	TIN74	TOUT74	TIM2_6	TIM1_6	TOM1_6	TOM0_14	ATOM 1_6	ATOM 2_6

Generic Timer Module (GTM)
Table 26-67 GTM to Port Mapping for QFP-144

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P15.4	TIN75	TOUT75	TIM2_7	TIM1_7	TOM1_7	TOM0_15	ATOM 1_7	ATOM 2_7
P15.5	TIN76	TOUT76	TIM2_0	TIM1_0	TOM0_0	TOM1_0	ATOM 1_0	ATOM 3_0
P15.6	TIN77	TOUT77	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM 1_0	ATOM 3_0
P15.7	TIN78	TOUT78	TIM0_1	TIM1_1	TOM0_1	TOM1_1	ATOM 1_1	ATOM 3_1
P15.8	TIN79	TOUT79	TIM0_2	TIM1_2	TOM0_2	TOM1_2	ATOM 1_1	ATOM 3_1
P20.0	TIN59	TOUT59	TIM0_6	TIM1_6	TOM0_6	TOM1_6	ATOM 0_6	ATOM 1_6
P20.3	TIN61	TOUT61	TIM2_4	TIM0_4	TOM1_12	TOM0_4	ATOM 0_4	ATOM 1_4
P20.6	TIN62	TOUT62	TIM2_6	TIM0_6	TOM1_10	TOM0_10	ATOM 2_6	ATOM 3_6
P20.7	TIN63	TOUT63	TIM2_7	TIM0_7	TOM1_11	TOM0_11	ATOM 2_7	ATOM 3_7
P20.8	TIN64	TOUT64	TIM0_7	TIM1_7	TOM1_7	TOM0_7	ATOM 0_7	ATOM 2_7
P20.9	TIN65	TOUT65	TIM2_5	TIM1_5	TOM1_13	TOM0_13	ATOM 1_5	ATOM 2_5
P20.10	TIN66	TOUT66	TIM2_6	TIM1_6	TOM1_14	TOM0_14	ATOM 1_6	ATOM 2_6
P20.11	TIN67	TOUT67	TIM2_7	TIM1_7	TOM1_15	TOM0_15	ATOM 1_7	ATOM 2_7
P20.12	TIN68	TOUT68	TIM2_0	TIM1_0	TOM1_0	TOM0_8	ATOM 1_0	ATOM 2_0
P20.13	TIN69	TOUT69	TIM2_1	TIM1_1	TOM1_1	TOM0_9	ATOM 1_1	ATOM 2_1
P20.14	TIN70	TOUT70	TIM2_2	TIM1_2	TOM1_2	TOM0_10	ATOM 1_2	ATOM 2_2

Generic Timer Module (GTM)
Table 26-67 GTM to Port Mapping for QFP-144

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P21.2	TIN53	TOUT53	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM 0_0	ATOM 1_0
P21.3	TIN54	TOUT54	TIM0_1	TIM1_1	TOM0_1	TOM1_1	ATOM 0_1	ATOM 1_1
P21.4	TIN55	TOUT55	TIM0_2	TIM1_2	TOM0_2	TOM1_2	ATOM 0_2	ATOM 1_2
P21.5	TIN56	TOUT56	TIM0_3	TIM1_3	TOM0_3	TOM1_3	ATOM 0_3	ATOM 1_3
P21.6	TIN57	TOUT57	TIM0_4	TIM1_4	TOM0_4	TOM1_4	ATOM 0_4	ATOM 1_4
P21.7	TIN58	TOUT58	TIM0_5	TIM1_5	TOM0_5	TOM1_5	ATOM 0_5	ATOM 1_5
P22.0	TIN47	TOUT47	TIM0_1	TIM1_1	TOM0_9	TOM1_1	ATOM 0_1	ATOM 1_1
P22.1	TIN48	TOUT48	TIM0_0	TIM1_0	TOM0_8	TOM1_0	ATOM 0_0	ATOM 1_0
P22.2	TIN49	TOUT49	TIM0_3	TIM1_3	TOM0_11	TOM1_3	ATOM 0_3	ATOM 1_3
P22.3	TIN50	TOUT50	TIM0_4	TIM1_4	TOM0_12	TOM1_4	ATOM 0_4	ATOM 1_4
P23.1	TIN42	TOUT42	TIM0_6	TIM1_6	TOM0_6	TOM0_15	ATOM 0_6	ATOM 1_6
P32.0	TIN36	TOUT36	TIM2_2	TIM0_2	TOM1_14	TOM0_14	ATOM 2_6	ATOM 3_6
P32.4	TIN40	TOUT40	TIM0_5	TIM1_5	TOM0_5	TOM1_5	ATOM 0_5	ATOM 1_5
P33.4	TIN26	TOUT26	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM 2_0	ATOM 3_0
P33.5	TIN27	TOUT27	TIM0_1	TIM1_1	TOM0_1	TOM1_1	ATOM 2_1	ATOM 3_1
P33.6	TIN28	TOUT28	TIM0_2	TIM1_2	TOM0_2	TOM1_2	ATOM 2_2	ATOM 3_2

Generic Timer Module (GTM)
Table 26-67 GTM to Port Mapping for QFP-144

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P33.7	TIN29	TOUT29	TIM0_3	TIM1_3	TOM0_3	TOM1_3	ATOM 2_3	ATOM 3_3
P33.8	TIN30	TOUT30	TIM0_4	TIM1_4	TOM0_4	TOM1_4	ATOM 2_4	ATOM 3_4
P33.9	TIN31	TOUT31	TIM0_1	TIM1_1	TOM0_1	TOM1_1	ATOM 0_1	ATOM 1_1
P33.10	TIN32	TOUT32	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM 2_0	ATOM 3_0
P33.11	TIN33	TOUT33	TIM0_2	TIM1_2	TOM0_2	TOM1_2	ATOM 0_2	ATOM 1_2
P33.12	TIN34	TOUT34	TIM2_0	TIM0_0	TOM1_12	TOM0_12	ATOM 2_4	ATOM 3_4
P33.13	TIN35	TOUT35	TIM2_1	TIM0_1	TOM1_13	TOM0_13	ATOM 2_5	ATOM 3_5

Generic Timer Module (GTM)
Table 26-68 GTM to Port Mapping for QFP-176 / BGA-292

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P00.0	TIN9	TOUT9	TIM2_0	TIM0_0	TOM0_8	TOM1_0	ATOM_0_0	ATOM_1_0
P00.1	TIN10	TOUT10	TIM2_1	TIM0_1	TOM0_9	TOM1_1	ATOM_0_1	ATOM_1_1
P00.2	TIN11	TOUT11	TIM2_1	TIM0_1	TOM0_9	TOM1_1	ATOM_0_1	ATOM_1_1
P00.3	TIN12	TOUT12	TIM2_2	TIM1_2	TOM0_10	TOM1_2	ATOM_0_2	ATOM_1_2
P00.4	TIN13	TOUT13	TIM2_3	TIM1_3	TOM0_11	TOM1_3	ATOM_0_3	ATOM_1_3
P00.5	TIN14	TOUT14	TIM2_4	TIM1_4	TOM0_12	TOM1_4	ATOM_0_4	ATOM_1_4
P00.6	TIN15	TOUT15	TIM2_5	TIM1_5	TOM0_13	TOM1_5	ATOM_0_5	ATOM_1_5
P00.7	TIN16	TOUT16	TIM2_6	TIM1_6	TOM0_14	TOM1_6	ATOM_0_6	ATOM_1_6
P00.8	TIN17	TOUT17	TIM2_7	TIM1_7	TOM0_15	TOM1_7	ATOM_0_7	ATOM_1_7
P00.9	TIN18	TOUT18	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM_2_0	ATOM_3_0
P00.10	TIN19	TOUT19	TIM0_1	TIM1_1	TOM0_1	TOM1_1	ATOM_2_1	ATOM_3_1
P00.11	TIN20	TOUT20	TIM0_2	TIM1_2	TOM0_2	TOM1_2	ATOM_2_2	ATOM_3_2
P00.12	TIN21	TOUT21	TIM0_3	TIM1_3	TOM0_3	TOM1_3	ATOM_2_3	ATOM_3_3
P02.0	TIN0	TOUT0	TIM0_0	TIM1_0	TOM0_8	TOM1_8	ATOM_0_0	ATOM_1_0
P02.1	TIN1	TOUT1	TIM0_1	TIM1_1	TOM0_9	TOM1_9	ATOM_0_1	ATOM_1_1

Generic Timer Module (GTM)
Table 26-68 GTM to Port Mapping for QFP-176 / BGA-292

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P02.2	TIN2	TOUT2	TIM0_2	TIM1_2	TOM0_10	TOM1_10	ATOM_0_2	ATOM_1_2
P02.3	TIN3	TOUT3	TIM0_3	TIM1_3	TOM0_11	TOM1_11	ATOM_0_3	ATOM_1_3
P02.4	TIN4	TOUT4	TIM0_4	TIM1_4	TOM0_12	TOM1_12	ATOM_0_4	ATOM_1_4
P02.5	TIN5	TOUT5	TIM0_5	TIM1_5	TOM0_13	TOM1_13	ATOM_0_5	ATOM_1_5
P02.6	TIN6	TOUT6	TIM0_6	TIM1_6	TOM0_14	TOM1_14	ATOM_0_6	ATOM_1_6
P02.7	TIN7	TOUT7	TIM0_7	TIM1_7	TOM0_15	TOM1_15	ATOM_0_7	ATOM_1_7
P02.8	TIN8	TOUT8	TIM2_0	TIM0_0	TOM0_8	TOM1_0	ATOM_0_0	ATOM_1_0
P10.0	TIN102	TOUT102	TIM0_4	TIM1_4	TOM0_4	TOM1_12	ATOM_1_4	ATOM_0_4
P10.1	TIN103	TOUT103	TIM0_1	TIM1_1	TOM0_1	TOM1_9	ATOM_1_1	ATOM_0_1
P10.2	TIN104	TOUT104	TIM0_2	TIM1_2	TOM0_2	TOM1_10	ATOM_1_2	ATOM_0_2
P10.3	TIN105	TOUT105	TIM0_3	TIM1_3	TOM0_3	TOM1_11	ATOM_1_3	ATOM_0_3
P10.4	TIN106	TOUT106	TIM0_6	TIM1_6	TOM0_6	TOM1_6	ATOM_0_6	ATOM_1_6
P10.5	TIN107	TOUT107	TIM0_2	TIM1_2	TOM0_2	TOM1_10	ATOM_1_2	ATOM_0_2
P10.6	TIN108	TOUT108	TIM0_3	TIM1_3	TOM0_3	TOM1_11	ATOM_1_3	ATOM_0_3
P10.7	TIN109	TOUT109	TIM0_0	TIM1_0	TOM0_0	TOM1_8	ATOM_1_0	ATOM_0_0
P10.8	TIN110	TOUT110	TIM0_5	TIM1_5	TOM0_5	TOM1_13	ATOM_1_5	ATOM_0_5

Generic Timer Module (GTM)
Table 26-68 GTM to Port Mapping for QFP-176 / BGA-292

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P11.2	TIN95	TOUT95	TIM2_1	TIM1_1	TOM0_8	TOM1_1	ATOM 2_1	ATOM 3_1
P11.3	TIN96	TOUT96	TIM2_2	TIM1_2	TOM0_10	TOM1_2	ATOM 2_2	ATOM 3_2
P11.6	TIN97	TOUT97	TIM2_3	TIM1_3	TOM0_11	TOM1_3	ATOM 2_3	ATOM 3_3
P11.9	TIN98	TOUT98	TIM2_4	TIM1_4	TOM0_12	TOM1_4	ATOM 2_4	ATOM 3_4
P11.10	TIN99	TOUT99	TIM2_5	TIM1_5	TOM0_13	TOM1_5	ATOM 2_5	ATOM 3_5
P11.11	TIN100	TOUT100	TIM2_6	TIM1_6	TOM0_14	TOM1_6	ATOM 2_6	ATOM 3_6
P11.12	TIN101	TOUT101	TIM2_7	TIM1_7	TOM0_15	TOM1_7	ATOM 2_7	ATOM 3_7
P13.0	TIN91	TOUT91	TIM2_5	TIM1_5	TOM0_5	TOM1_5	ATOM 2_5	ATOM 3_5
P13.1	TIN92	TOUT92	TIM2_6	TIM1_6	TOM0_6	TOM1_6	ATOM 2_6	ATOM 3_6
P13.2	TIN93	TOUT93	TIM2_7	TIM1_7	TOM0_7	TOM1_7	ATOM 2_7	ATOM 3_7
P13.3	TIN94	TOUT94	TIM2_0	TIM1_0	TOM0_8	TOM1_0	ATOM 2_0	ATOM 3_0
P14.0	TIN80	TOUT80	TIM0_3	TIM1_3	TOM0_3	TOM1_3	ATOM 1_2	ATOM 3_2
P14.1	TIN81	TOUT81	TIM0_4	TIM1_4	TOM0_4	TOM1_4	ATOM 0_4	ATOM 3_4
P14.2	TIN82	TOUT82	TIM0_5	TIM1_5	TOM0_5	TOM1_5	ATOM 0_3	ATOM 3_3
P14.3	TIN83	TOUT83	TIM0_6	TIM1_6	TOM0_6	TOM1_6	ATOM 0_2	ATOM 3_2
P14.4	TIN84	TOUT84	TIM0_7	TIM1_7	TOM0_7	TOM1_7	ATOM 0_1	ATOM 3_1

Generic Timer Module (GTM)
Table 26-68 GTM to Port Mapping for QFP-176 / BGA-292

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P14.5	TIN85	TOUT85	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM 0_0	ATOM 3_0
P14.6	TIN86	TOUT86	TIM0_1	TIM1_1	TOM0_1	TOM1_1	ATOM 1_1	ATOM 3_1
P14.7	TIN87	TOUT87	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM 1_0	ATOM 3_0
P14.8	TIN88	TOUT88	TIM2_2	TIM2_3	TOM0_2	TOM1_2	ATOM 2_2	ATOM 3_2
P14.9	TIN89	TOUT89	TIM2_3	TIM2_1	TOM0_3	TOM1_3	ATOM 2_3	ATOM 3_3
P14.10	TIN90	TOUT90	TIM2_4	TIM2_2	TOM0_4	TOM1_4	ATOM 2_4	ATOM 3_4
P15.0	TIN71	TOUT71	TIM2_3	TIM1_3	TOM1_3	TOM0_11	ATOM 1_3	ATOM 2_3
P15.1	TIN72	TOUT72	TIM2_4	TIM1_4	TOM1_4	TOM0_12	ATOM 1_4	ATOM 2_4
P15.2	TIN73	TOUT73	TIM2_5	TIM1_5	TOM1_5	TOM0_13	ATOM 1_5	ATOM 2_5
P15.3	TIN74	TOUT74	TIM2_6	TIM1_6	TOM1_6	TOM0_14	ATOM 1_6	ATOM 2_6
P15.4	TIN75	TOUT75	TIM2_7	TIM1_7	TOM1_7	TOM0_15	ATOM 1_7	ATOM 2_7
P15.5	TIN76	TOUT76	TIM2_0	TIM1_0	TOM0_0	TOM1_0	ATOM 1_0	ATOM 3_0
P15.6	TIN77	TOUT77	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM 1_0	ATOM 3_0
P15.7	TIN78	TOUT78	TIM0_1	TIM1_1	TOM0_1	TOM1_1	ATOM 1_1	ATOM 3_1
P15.8	TIN79	TOUT79	TIM0_2	TIM1_2	TOM0_2	TOM1_2	ATOM 1_1	ATOM 3_1
P20.0	TIN59	TOUT59	TIM0_6	TIM1_6	TOM0_6	TOM1_6	ATOM 0_6	ATOM 1_6

Generic Timer Module (GTM)
Table 26-68 GTM to Port Mapping for QFP-176 / BGA-292

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P20.1	TIN60	TOUT60	TIM2_3	TIM0_3	TOM1_11	TOM0_3	ATOM_0_3	ATOM_1_3
P20.3	TIN61	TOUT61	TIM2_4	TIM0_4	TOM1_12	TOM0_4	ATOM_0_4	ATOM_1_4
P20.6	TIN62	TOUT62	TIM2_6	TIM0_6	TOM1_10	TOM0_10	ATOM_2_6	ATOM_3_6
P20.7	TIN63	TOUT63	TIM2_7	TIM0_7	TOM1_11	TOM0_11	ATOM_2_7	ATOM_3_7
P20.8	TIN64	TOUT64	TIM0_7	TIM1_7	TOM1_7	TOM0_7	ATOM_0_7	ATOM_2_7
P20.9	TIN65	TOUT65	TIM2_5	TIM1_5	TOM1_13	TOM0_13	ATOM_1_5	ATOM_2_5
P20.10	TIN66	TOUT66	TIM2_6	TIM1_6	TOM1_14	TOM0_14	ATOM_1_6	ATOM_2_6
P20.11	TIN67	TOUT67	TIM2_7	TIM1_7	TOM1_15	TOM0_15	ATOM_1_7	ATOM_2_7
P20.12	TIN68	TOUT68	TIM2_0	TIM1_0	TOM1_0	TOM0_8	ATOM_1_0	ATOM_2_0
P20.13	TIN69	TOUT69	TIM2_1	TIM1_1	TOM1_1	TOM0_9	ATOM_1_1	ATOM_2_1
P20.14	TIN70	TOUT70	TIM2_2	TIM1_2	TOM1_2	TOM0_10	ATOM_1_2	ATOM_2_2
P21.0	TIN51	TOUT51	TIM2_4	TIM0_3	TOM0_8	TOM1_8	ATOM_2_4	ATOM_3_4
P21.1	TIN52	TOUT52	TIM2_5	TIM0_4	TOM0_9	TOM1_9	ATOM_2_5	ATOM_3_5
P21.2	TIN53	TOUT53	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM_0_0	ATOM_1_0
P21.3	TIN54	TOUT54	TIM0_1	TIM1_1	TOM0_1	TOM1_1	ATOM_0_1	ATOM_1_1
P21.4	TIN55	TOUT55	TIM0_2	TIM1_2	TOM0_2	TOM1_2	ATOM_0_2	ATOM_1_2

Generic Timer Module (GTM)
Table 26-68 GTM to Port Mapping for QFP-176 / BGA-292

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P21.5	TIN56	TOUT56	TIM0_3	TIM1_3	TOM0_3	TOM1_3	ATOM_0_3	ATOM_1_3
P21.6	TIN57	TOUT57	TIM0_4	TIM1_4	TOM0_4	TOM1_4	ATOM_0_4	ATOM_1_4
P21.7	TIN58	TOUT58	TIM0_5	TIM1_5	TOM0_5	TOM1_5	ATOM_0_5	ATOM_1_5
P22.0	TIN47	TOUT47	TIM0_1	TIM1_1	TOM0_9	TOM1_1	ATOM_0_1	ATOM_1_1
P22.1	TIN48	TOUT48	TIM0_0	TIM1_0	TOM0_8	TOM1_0	ATOM_0_0	ATOM_1_0
P22.2	TIN49	TOUT49	TIM0_3	TIM1_3	TOM0_11	TOM1_3	ATOM_0_3	ATOM_1_3
P22.3	TIN50	TOUT50	TIM0_4	TIM1_4	TOM0_12	TOM1_4	ATOM_0_4	ATOM_1_4
P23.0	TIN41	TOUT41	TIM0_5	TIM1_5	TOM0_10	TOM1_5	ATOM_0_5	ATOM_1_5
P23.1	TIN42	TOUT42	TIM0_6	TIM1_6	TOM0_6	TOM0_15	ATOM_0_6	ATOM_1_6
P23.2	TIN43	TOUT43	TIM0_6	TIM1_6	TOM0_11	TOM1_6	ATOM_2_1	ATOM_3_1
P23.3	TIN44	TOUT44	TIM0_7	TIM1_7	TOM0_12	TOM1_7	ATOM_2_2	ATOM_3_2
P23.4	TIN45	TOUT45	TIM0_7	TIM1_7	TOM0_7	TOM1_7	ATOM_0_7	ATOM_1_7
P23.5	TIN46	TOUT46	TIM0_2	TIM1_2	TOM0_10	TOM1_2	ATOM_0_2	ATOM_1_2
P32.0	TIN36	TOUT36	TIM2_2	TIM0_2	TOM1_14	TOM0_14	ATOM_2_6	ATOM_3_6
P32.2	TIN38	TOUT38	TIM0_3	TIM1_3	TOM0_3	TOM1_3	ATOM_0_3	ATOM_1_3
P32.3	TIN39	TOUT39	TIM0_4	TIM1_4	TOM0_4	TOM1_4	ATOM_0_4	ATOM_1_4

Generic Timer Module (GTM)
Table 26-68 GTM to Port Mapping for QFP-176 / BGA-292

Port	Input	Output	Input Timer Mapped		Output Timer Mapped			
			A	B	A	B	C	D
P32.4	TIN40	TOUT40	TIM0_5	TIM1_5	TOM0_5	TOM1_5	ATOM 0_5	ATOM 1_5
P33.0	TIN22	TOUT22	TIM0_4	TIM1_4	TOM0_4	TOM1_4	ATOM 2_4	ATOM 3_4
P33.1	TIN23	TOUT23	TIM0_5	TIM1_5	TOM0_5	TOM1_5	ATOM 0_5	ATOM 1_5
P33.2	TIN24	TOUT24	TIM0_6	TIM1_6	TOM0_6	TOM1_6	ATOM 0_6	ATOM 1_6
P33.3	TIN25	TOUT25	TIM0_7	TIM1_7	TOM0_7	TOM1_7	ATOM 0_7	ATOM 1_7
P33.4	TIN26	TOUT26	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM 2_0	ATOM 3_0
P33.5	TIN27	TOUT27	TIM0_1	TIM1_1	TOM0_1	TOM1_1	ATOM 2_1	ATOM 3_1
P33.6	TIN28	TOUT28	TIM0_2	TIM1_2	TOM0_2	TOM1_2	ATOM 2_2	ATOM 3_2
P33.7	TIN29	TOUT29	TIM0_3	TIM1_3	TOM0_3	TOM1_3	ATOM 2_3	ATOM 3_3
P33.8	TIN30	TOUT30	TIM0_4	TIM1_4	TOM0_4	TOM1_4	ATOM 2_4	ATOM 3_4
P33.9	TIN31	TOUT31	TIM0_1	TIM1_1	TOM0_1	TOM1_1	ATOM 0_1	ATOM 1_1
P33.10	TIN32	TOUT32	TIM0_0	TIM1_0	TOM0_0	TOM1_0	ATOM 2_0	ATOM 3_0
P33.11	TIN33	TOUT33	TIM0_2	TIM1_2	TOM0_2	TOM1_2	ATOM 0_2	ATOM 1_2
P33.12	TIN34	TOUT34	TIM2_0	TIM0_0	TOM1_12	TOM0_12	ATOM 2_4	ATOM 3_4
P33.13	TIN35	TOUT35	TIM2_1	TIM0_1	TOM1_13	TOM0_13	ATOM 2_5	ATOM 3_5

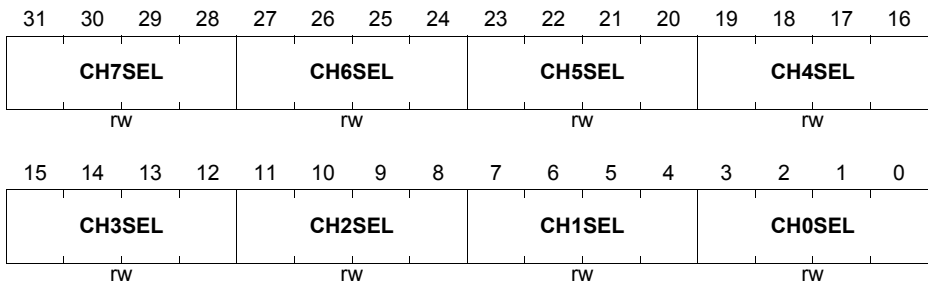
The GTM outputs *cmu_eclk[2:0]* are connected directly to dedicated ports.

Table 26-69 GTM clock to Port Mapping

GTM Clock Output	Alternate Output of Pin
<i>cmu_eclk0</i>	P23.1
<i>cmu_eclk1</i>	P32.4
<i>cmu_eclk2</i>	P11.12

Generic Timer Module (GTM)
26.22.2.1 Port to GTM Control Registers

The inputs to the GTM TIM modules (*gtm_timx_in[7:0]*) are not connected directly to the port input path. Each timed GPIO is connectable to two TIMs via an input multiplexer. In addition the inputs from the on chip peripherals are connected here too.

TIMnINSEL (n = 0-2)
TIMn Input Select Register (9FD10_H+n*4_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
CHxSEL (x = 0-7)	[x*4+3: x*4]	rw	TIM Channel x Input Selection This bit defines which input is connected for TIMn channel x of the GTM. The input is either derived from the a port pad or from a on-chip module.

Generic Timer Module (GTM)
Table 26-70 TIM 0 Mapping for QFP-144

CH0SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.9	TIN18
0010 _B	P02.0	TIN0
0011 _B	Reserved	-
0100 _B	P14.5	TIN85
0101 _B	Reserved	-
0110 _B	P15.6	TIN77
0111 _B	P21.2	TIN53
1000 _B	P22.1	TIN48
1001 _B	P33.10	TIN32
1010 _B	P33.4	TIN26
1011 _B	P02.8	TIN8
1100 _B	P33.12	TIN34
1101 _B	P00.0	TIN9
1110 _B	<i>tim0_muxout_0</i>	DSADC trigger
1111 _B	<i>vadc_COSR0</i>	ADC
CH1SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	Reserved	-
0010 _B	P02.1	TIN1
0011 _B	P10.1	TIN103
0100 _B	P14.6	TIN86
0101 _B	P15.7	TIN78
0110 _B	P21.3	TIN54
0111 _B	P22.0	TIN47
1000 _B	P33.5	TIN27
1001 _B	P33.9	TIN31
1010 _B	P00.1	TIN10
1011 _B	P00.2	TIN11
1100 _B	P33.13	TIN35

Generic Timer Module (GTM)

Table 26-70 TIM 0 Mapping for QFP-144 (cont'd)

1101 _B	<i>can_int[12]</i>	CAN
1110 _B	<i>tim0_muxout_1</i>	DSADC trigger
1111 _B	<i>vadc_CISR0</i>	ADC
CH2SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	Reserved	-
0010 _B	P02.2	TIN2
0011 _B	P10.2	TIN104
0100 _B	P10.5	TIN107
0101 _B	P15.8	TIN79
0110 _B	P21.4	TIN55
0111 _B	Reserved	-
1000 _B	P33.11	TIN33
1001 _B	P33.6	TIN28
1010 _B	Reserved	-
1011 _B	P32.0	TIN36
1100 _B	<i>eru_pdout2</i>	SCU
1101 _B	<i>can_int[13]</i>	CAN
1110 _B	<i>tim0_muxout_2</i>	DSADC trigger
1111 _B	<i>vadc_COSR1</i>	ADC
CH3SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.12	TIN21
0010 _B	P02.3	TIN3
0011 _B	P10.3	TIN105
0100 _B	P10.6	TIN108
0101 _B	P14.0	TIN80
0110 _B	P21.5	TIN56
0111 _B	P22.2	TIN49
1000 _B	Reserved	-
1001 _B	P33.7	TIN29

Generic Timer Module (GTM)

Table 26-70 TIM 0 Mapping for QFP-144 (cont'd)

1010 _B	Reserved	-
1011 _B	Reserved	-
1100 _B	<i>eru_pdout3</i>	SCU
1101 _B	<i>can_int[14]</i>	CAN
1110 _B	<i>tim0_muxout_3</i>	DSADC trigger
1111 _B	<i>vadc_CISR1</i>	ADC
CH4SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.4	TIN4
0010 _B	Reserved	-
0011 _B	P14.1	TIN81
0100 _B	P22.3	TIN50
0101 _B	Reserved	-
0110 _B	Reserved	-
0111 _B	P33.8	TIN30
1000 _B	P21.6	TIN57
1001 _B	P20.3	TIN61
1010 _B	Reserved	-
1011 _B	Reserved	-
1100 _B	<i>eru_pdout4</i>	SCU
1101 _B	<i>can_int[15]</i>	CAN
1110 _B	<i>tim0_muxout_4</i>	DSADC trigger
1111 _B	<i>vadc_COSR2</i>	ADC
CH5SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.5	TIN5
0010 _B	Reserved	-
0011 _B	P14.2	TIN82
0100 _B	Reserved	-
0101 _B	P32.4	TIN40
0110 _B	Reserved	-

Generic Timer Module (GTM)

Table 26-70 TIM 0 Mapping for QFP-144 (cont'd)

0111 _B	P21.7	TIN58
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	Reserved	-
1011 _B	Reserved	-
1100 _B	<i>eru_pdout5</i>	SCU
1101 _B	Reserved	-
1110 _B	<i>tim0_muxout_5</i>	DSADC trigger
1111 _B	<i>vadc_CISR2</i>	ADC
CH6SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.6	TIN6
0010 _B	Reserved	-
0011 _B	P14.3	TIN83
0100 _B	P23.1	TIN42
0101 _B	Reserved	-
0110 _B	Reserved	-
0111 _B	P20.0	TIN59
1000 _B	Reserved	-
1001 _B	P20.6	TIN62
1010 _B	Reserved	-
1011 _B	Reserved	-
1100 _B	<i>eru_pdout6</i>	SCU
1101 _B	Reserved	-
1110 _B	<i>tim0_muxout_6</i>	DSADC trigger
1111 _B	<i>vadc_COSR3</i>	ADC
CH7SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.7	TIN7
0010 _B	P14.4	TIN84
0011 _B	P20.8	TIN64

Generic Timer Module (GTM)

Table 26-70 TIM 0 Mapping for QFP-144 (cont'd)

0100 _B	Reserved	-
0101 _B	Reserved	-
0110 _B	Reserved	-
0111 _B	P20.7	TIN63
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	Reserved	-
1011 _B	Reserved	-
1100 _B	<i>eru_pdout7</i>	SCU
1101 _B	<i>eray_mt</i>	ERAY
1110 _B	<i>tim0_muxout_7</i>	DSADC trigger
1111 _B	<i>vadc_CISR3</i>	ADC

Generic Timer Module (GTM)
Table 26-71 TIM 0 Mapping for QFP-176 / BGA-292

CH0SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.9	TIN18
0010 _B	P02.0	TIN0
0011 _B	P10.7	TIN109
0100 _B	P14.5	TIN85
0101 _B	P14.7	TIN87
0110 _B	P15.6	TIN77
0111 _B	P21.2	TIN53
1000 _B	P22.1	TIN48
1001 _B	P33.10	TIN32
1010 _B	P33.4	TIN26
1011 _B	P02.8	TIN8
1100 _B	P33.12	TIN34
1101 _B	P00.0	TIN9
1110 _B	<i>tim0_muxout_0</i>	DSADC trigger
1111 _B	<i>vadc_COSR0</i>	ADC
CH1SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.10	TIN19
0010 _B	P02.1	TIN1
0011 _B	P10.1	TIN103
0100 _B	P14.6	TIN86
0101 _B	P15.7	TIN78
0110 _B	P21.3	TIN54
0111 _B	P22.0	TIN47
1000 _B	P33.5	TIN27
1001 _B	P33.9	TIN31
1010 _B	P00.1	TIN10
1011 _B	P00.2	TIN11
1100 _B	P33.13	TIN35

Generic Timer Module (GTM)

Table 26-71 TIM 0 Mapping for QFP-176 (cont'd)/ BGA-292

1101 _B	<i>can_int[12]</i>	CAN
1110 _B	<i>tim0_muxout_1</i>	DSADC trigger
1111 _B	<i>vadc_CISR0</i>	ADC
CH2SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.11	TIN20
0010 _B	P02.2	TIN2
0011 _B	P10.2	TIN104
0100 _B	P10.5	TIN107
0101 _B	P15.8	TIN79
0110 _B	P21.4	TIN55
0111 _B	P23.5	TIN46
1000 _B	P33.11	TIN33
1001 _B	P33.6	TIN28
1010 _B	Reserved	-
1011 _B	P32.0	TIN36
1100 _B	<i>eru_pdout2</i>	SCU
1101 _B	<i>can_int[13]</i>	CAN
1110 _B	<i>tim0_muxout_2</i>	DSADC trigger
1111 _B	<i>vadc_COSR1</i>	ADC
CH3SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.12	TIN21
0010 _B	P02.3	TIN3
0011 _B	P10.3	TIN105
0100 _B	P10.6	TIN108
0101 _B	P14.0	TIN80
0110 _B	P21.5	TIN56
0111 _B	P22.2	TIN49
1000 _B	P32.2	TIN38
1001 _B	P33.7	TIN29

Generic Timer Module (GTM)

Table 26-71 TIM 0 Mapping for QFP-176 (cont'd)/ BGA-292

1010 _B	P20.1	TIN60
1011 _B	P21.0	TIN51
1100 _B	<i>eru_pdout3</i>	SCU
1101 _B	<i>can_int[14]</i>	CAN
1110 _B	<i>tim0_muxout_3</i>	DSADC trigger
1111 _B	<i>vadc_CISR1</i>	ADC
CH4SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.4	TIN4
0010 _B	P10.0	TIN102
0011 _B	P14.1	TIN81
0100 _B	P22.3	TIN50
0101 _B	P32.3	TIN39
0110 _B	P33.0	TIN22
0111 _B	P33.8	TIN30
1000 _B	P21.6	TIN57
1001 _B	P20.3	TIN61
1010 _B	P21.1	TIN52
1011 _B	Reserved	-
1100 _B	<i>eru_pdout4</i>	SCU
1101 _B	<i>can_int[15]</i>	CAN
1110 _B	<i>tim0_muxout_4</i>	DSADC trigger
1111 _B	<i>vadc_COSR2</i>	ADC
CH5SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.5	TIN5
0010 _B	P10.8	TIN110
0011 _B	P14.2	TIN82
0100 _B	P23.0	TIN41
0101 _B	P32.4	TIN40
0110 _B	P31.1	TIN23

Generic Timer Module (GTM)

Table 26-71 TIM 0 Mapping for QFP-176 (cont'd)/ BGA-292

0111 _B	P21.7	TIN58
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	Reserved	-
1011 _B	Reserved	-
1100 _B	<i>eru_pdout5</i>	SCU
1101 _B	Reserved	-
1110 _B	<i>tim0_muxout_5</i>	DSADC trigger
1111 _B	<i>vadc_CISR2</i>	ADC
CH6SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P10.4	TIN106
0010 _B	P20.6	TIN62
0011 _B	P14.3	TIN83
0100 _B	P23.1	TIN42
0101 _B	P23.2	TIN43
0110 _B	P33.2	TIN24
0111 _B	P20.0	TIN59
1000 _B	Reserved	-
1001 _B	P20.6	TIN62
1010 _B	Reserved	-
1011 _B	Reserved	-
1100 _B	<i>eru_pdout6</i>	SCU
1101 _B	Reserved	-
1110 _B	<i>tim0_muxout_6</i>	DSADC trigger
1111 _B	<i>vadc_COSR3</i>	ADC
CH7SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.7	TIN7
0010 _B	P14.4	TIN84
0011 _B	P20.8	TIN64

Generic Timer Module (GTM)
Table 26-71 TIM 0 Mapping for QFP-176 (cont'd)/ BGA-292

0100 _B	P23.3	TIN44
0101 _B	P23.4	TIN45
0110 _B	P33.3	TIN25
0111 _B	P20.7	TIN63
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	Reserved	-
1011 _B	Reserved	-
1100 _B	<i>eru_pdout7</i>	SCU
1101 _B	<i>eray_mt</i>	ERAY
1110 _B	<i>tim0_muxout_7</i>	DSADC trigger
1111 _B	<i>vadc_CISR3</i>	ADC

Generic Timer Module (GTM)
Table 26-72 TIM 1 Mapping for QFP-144

CH0SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.9	TIN18
0010 _B	P02.0	TIN0
0011 _B	P13.3	TIN94
0100 _B	P14.5	TIN85
0101 _B	P15.5	TIN76
0110 _B	P15.6	TIN77
0111 _B	P21.2	TIN53
1000 _B	P22.1	TIN48
1001 _B	P33.10	TIN32
1010 _B	P33.4	TIN26
1011 _B	<i>tim1_muxout_0</i>	DSADC trigger
1100 _B	<i>vadc_C0SR2</i>	ADC
1101 _B	Reserved	-
1110 _B	Reserved	-
1111 _B	Reserved	-
CH1SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P11.2	TIN95
0010 _B	P02.1	TIN1
0011 _B	P10.1	TIN103
0100 _B	P14.6	TIN86
0101 _B	P15.7	TIN78
0110 _B	P21.3	TIN54
0111 _B	P22.0	TIN47
1000 _B	P33.5	TIN27
1001 _B	P33.9	TIN31
1010 _B	P20.13	TIN69
1011 _B	<i>tim1_muxout_1</i>	DSADC trigger
1100 _B	<i>vadc_C1SR2</i>	ADC

Generic Timer Module (GTM)

Table 26-72 TIM 1 Mapping for QFP-144 (cont'd)

1101 _B	<i>can_int[12]</i>	CAN
1110 _B	Reserved	-
1111 _B	<i>vadc_GOARBCNT</i>	ADC
CH2SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.3	TIN12
0010 _B	P02.2	TIN2
0011 _B	P10.2	TIN104
0100 _B	P10.5	TIN107
0101 _B	P15.8	TIN79
0110 _B	P21.4	TIN55
0111 _B	P11.3	TIN96
1000 _B	P33.11	TIN33
1001 _B	P33.6	TIN28
1010 _B	P20.14	TIN70
1011 _B	<i>tim1_muxout_2</i>	DSADC trigger
1100 _B	<i>vadc_COSR3</i>	ADC
1101 _B	<i>can_int[13]</i>	CAN
1110 _B	Reserved	-
1111 _B	Reserved	-
CH3SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.4	TIN13
0010 _B	P02.3	TIN3
0011 _B	P10.3	TIN105
0100 _B	P10.6	TIN108
0101 _B	P14.0	TIN80
0110 _B	P21.5	TIN56
0111 _B	P22.2	TIN49
1000 _B	P32.2	TIN38
1001 _B	P33.7	TIN29

Generic Timer Module (GTM)
Table 26-72 TIM 1 Mapping for QFP-144 (cont'd)

1010 _B	P15.0	TIN71
1011 _B	P00.12	TIN21
1100 _B	P11.6	TIN97
1101 _B	<i>can_int[14]</i>	CAN
1110 _B	<i>tim1_muxout_3</i>	DSADC trigger
1111 _B	<i>vadc_CISR3</i>	ADC
CH4SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.4	TIN4
0010 _B	Reserved	-
0011 _B	P14.1	TIN81
0100 _B	P22.3	TIN50
0101 _B	Reserved	-
0110 _B	Reserved	-
0111 _B	P33.8	TIN30
1000 _B	P21.6	TIN57
1001 _B	P00.5	TIN14
1010 _B	<i>vadc_COSR0</i>	ADC
1011 _B	P11.9	TIN98
1100 _B	P15.1	TIN72
1101 _B	<i>can_int[15]</i>	CAN
1110 _B	<i>tim1_muxout_4</i>	DSADC trigger
1111 _B	<i>vadc_G2ARBCNT</i>	ADC
CH5SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.5	TIN5
0010 _B	Reserved	-
0011 _B	P14.2	TIN82
0100 _B	Reserved	-
0101 _B	P32.4	TIN40
0110 _B	P13.0	TIN91

Generic Timer Module (GTM)

Table 26-72 TIM 1 Mapping for QFP-144 (cont'd)

0111 _B	P21.7	TIN58
1000 _B	P15.2	TIN73
1001 _B	P20.9	TIN65
1010 _B	<i>vadc_CISR0</i>	ADC
1011 _B	P00.6	TIN15
1100 _B	P11.10	TIN99
1101 _B	Reserved	-
1110 _B	<i>tim1_muxout_5</i>	DSADC trigger
1111 _B	Reserved	-
CH6SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.6	TIN6
0010 _B	Reserved	-
0011 _B	P14.3	TIN83
0100 _B	P23.1	TIN42
0101 _B	Reserved	-
0110 _B	Reserved	-
0111 _B	P20.0	TIN59
1000 _B	P15.3	TIN74
1001 _B	P20.10	TIN66
1010 _B	<i>vadc_COSR1</i>	ADC
1011 _B	P00.7	TIN16
1100 _B	P11.11	TIN100
1101 _B	P13.1	TIN92
1110 _B	<i>tim1_muxout_6</i>	DSADC trigger
1111 _B	Reserved	-
CH7SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.7	TIN7
0010 _B	P14.4	TIN84
0011 _B	P20.8	TIN64

Generic Timer Module (GTM)

Table 26-72 TIM 1 Mapping for QFP-144 (cont'd)

0100 _B	Reserved	-
0101 _B	Reserved	-
0110 _B	Reserved	-
0111 _B	P15.4	TIN75
1000 _B	P20.11	TIN67
1001 _B	P13.2	TIN93
1010 _B	<i>vadc_CISR1</i>	ADC
1011 _B	P00.8	TIN17
1100 _B	P11.12	TIN101
1101 _B	<i>eray_mt</i>	ERAY
1110 _B	<i>tim1_muxout_7</i>	DSADC trigger
1111 _B	Reserved	-

Generic Timer Module (GTM)
Table 26-73 TIM 1 Mapping for QFP-176 / BGA-292

CH0SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.9	TIN18
0010 _B	P02.0	TIN0
0011 _B	P13.3	TIN94
0100 _B	P14.5	TIN85
0101 _B	P15.5	TIN76
0110 _B	P15.6	TIN77
0111 _B	P21.2	TIN53
1000 _B	P22.1	TIN48
1001 _B	P33.10	TIN32
1010 _B	P33.4	TIN26
1011 _B	<i>tim1_muxout_0</i>	DSADC trigger
1100 _B	<i>vadc_C0SR2</i>	ADC
1101 _B	P14.7	TIN87
1110 _B	P10.7	TIN109
1111 _B	P20.12	TIN68
CH1SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P11.2	TIN95
0010 _B	P02.1	TIN1
0011 _B	P10.1	TIN103
0100 _B	P14.6	TIN86
0101 _B	P15.7	TIN78
0110 _B	P21.3	TIN54
0111 _B	P22.0	TIN47
1000 _B	P33.5	TIN27
1001 _B	P33.9	TIN31
1010 _B	P20.13	TIN69
1011 _B	<i>tim1_muxout_1</i>	DSADC trigger
1100 _B	<i>vadc_C1SR2</i>	ADC

Generic Timer Module (GTM)

Table 26-73 TIM 1 Mapping for QFP-176 / BGA-292 (cont'd)

1101 _B	<i>can_int[12]</i>	CAN
1110 _B	P00.10	TIN19
1111 _B	<i>vadc_GOARBCNT</i>	ADC
CH2SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.3	TIN12
0010 _B	P02.2	TIN2
0011 _B	P10.2	TIN104
0100 _B	P10.5	TIN107
0101 _B	P15.8	TIN79
0110 _B	P21.4	TIN55
0111 _B	P11.3	TIN96
1000 _B	P33.11	TIN33
1001 _B	P33.6	TIN28
1010 _B	P20.14	TIN70
1011 _B	<i>tim1_muxout_2</i>	DSADC trigger
1100 _B	<i>vadc_COSR3</i>	ADC
1101 _B	<i>can_int[13]</i>	CAN
1110 _B	P00.11	TIN20
1111 _B	P23.5	TIN46
CH3SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.4	TIN13
0010 _B	P02.3	TIN3
0011 _B	P10.3	TIN105
0100 _B	P10.6	TIN108
0101 _B	P14.0	TIN80
0110 _B	P21.5	TIN56
0111 _B	P22.2	TIN49
1000 _B	P32.2	TIN38
1001 _B	P33.7	TIN29

Generic Timer Module (GTM)
Table 26-73 TIM 1 Mapping for QFP-176 / BGA-292 (cont'd)

1010 _B	P15.0	TIN71
1011 _B	P00.12	TIN21
1100 _B	P11.6	TIN97
1101 _B	<i>can_int[14]</i>	CAN
1110 _B	<i>tim1_muxout_3</i>	DSADC trigger
1111 _B	<i>vadc_CISR3</i>	ADC
CH4SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.4	TIN4
0010 _B	P10.0	TIN102
0011 _B	P14.1	TIN81
0100 _B	P22.3	TIN50
0101 _B	P32.3	TIN39
0110 _B	P33.0	TIN22
0111 _B	P33.8	TIN30
1000 _B	P21.6	TIN57
1001 _B	P00.5	TIN14
1010 _B	<i>vadc_COSR0</i>	ADC
1011 _B	P11.9	TIN98
1100 _B	P15.1	TIN72
1101 _B	<i>can_int[15]</i>	CAN
1110 _B	<i>tim1_muxout_04</i>	DSADC trigger
1111 _B	<i>vadc_G2ARBCNT</i>	ADC
CH5SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.5	TIN5
0010 _B	P10.8	TIN110
0011 _B	P14.2	TIN82
0100 _B	P23.0	TIN41
0101 _B	P32.4	TIN40
0110 _B	P13.0	TIN91

Generic Timer Module (GTM)
Table 26-73 TIM 1 Mapping for QFP-176 / BGA-292 (cont'd)

0111 _B	P21.7	TIN58
1000 _B	P15.2	TIN73
1001 _B	P20.9	TIN65
1010 _B	<i>vadc_CISR0</i>	ADC
1011 _B	P00.6	TIN15
1100 _B	P11.10	TIN99
1101 _B	Reserved	-
1110 _B	<i>tim1_muxout_5</i>	DSADC trigger
1111 _B	Reserved	-
CH6SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.6	TIN6
0010 _B	P10.4	TIN106
0011 _B	P14.3	TIN83
0100 _B	P23.1	TIN42
0101 _B	P23.2	TIN43
0110 _B	P33.2	TIN24
0111 _B	P20.0	TIN59
1000 _B	P15.3	TIN74
1001 _B	P20.10	TIN66
1010 _B	<i>vadc_COSR1</i>	ADC
1011 _B	P00.7	TIN16
1100 _B	P11.11	TIN100
1101 _B	P13.1	TIN92
1110 _B	<i>tim1_muxout_6</i>	DSADC trigger
1111 _B	Reserved	-
CH7SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P02.7	TIN7
0010 _B	P14.4	TIN84
0011 _B	P20.8	TIN64

Generic Timer Module (GTM)
Table 26-73 TIM 1 Mapping for QFP-176 / BGA-292 (cont'd)

0100 _B	P23.3	TIN44
0101 _B	P23.4	TIN45
0110 _B	P33.3	TIN25
0111 _B	P15.4	TIN75
1000 _B	P20.11	TIN67
1001 _B	P13.2	TIN93
1010 _B	<i>vadc_CISR1</i>	ADC
1011 _B	P00.8	TIN17
1100 _B	P11.12	TIN101
1101 _B	<i>eray_mt</i>	ERAY
1110 _B	<i>tim1_muxout_7</i>	DSADC trigger
1111 _B	Reserved	-

Generic Timer Module (GTM)

Table 26-74 TIM 2 Mapping for QFP-144

CH0SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.0	TIN9
0010 _B	P02.8	TIN8
0011 _B	P13.3	TIN94
0100 _B	P15.5	TIN76
0101 _B	P20.12	TIN68
0110 _B	P33.12	TIN34
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_COSR0</i>	ADC
1011 _B	<i>tim2_muxout_0</i>	DSADC trigger
1100 _B	<i>vadc_GIARBCNT</i>	ADC
1101 _B	Reserved	-
1110 _B	Reserved	-
1111 _B	Reserved	-
CH1SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.1	TIN10
0010 _B	P00.2	TIN11
0011 _B	P11.2	TIN95
0100 _B	P20.13	TIN69
0101 _B	P33.13	TIN35
0110 _B	Reserved	-
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_CISR0</i>	ADC
1011 _B	<i>tim2_muxout_1</i>	DSADC trigger
1100 _B	<i>vadc_G3ARBCNT</i>	ADC

Generic Timer Module (GTM)

Table 26-74 TIM 2 Mapping for QFP-144 (cont'd)

1101 _B	<i>can_int[12]</i>	CAN
1110 _B	Reserved	-
1111 _B	Reserved	-
CH2SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.3	TIN12
0010 _B	P11.3	TIN96
0011 _B	Reserved	-
0100 _B	P20.14	TIN70
0101 _B	P32.0	TIN36
0110 _B	Reserved	-
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_COSR1</i>	ADC
1011 _B	<i>tim2_muxout_2</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	<i>can_int[13]</i>	CAN
1110 _B	Reserved	-
1111 _B	Reserved	-
CH3SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.4	TIN13
0010 _B	P11.6	TIN97
0011 _B	Reserved	-
0100 _B	P15.0	TIN71
0101 _B	Reserved	-
0110 _B	Reserved	-
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-

Generic Timer Module (GTM)

Table 26-74 TIM 2 Mapping for QFP-144 (cont'd)

1010 _B	<i>vadc_CISR1</i>	ADC
1011 _B	<i>tim2_muxout_3</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	<i>can_int[14]</i>	CAN
1110 _B	Reserved	-
1111 _B	Reserved	-
CH4SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.5	TIN14
0010 _B	P11.9	TIN98
0011 _B	Reserved	-
0100 _B	P15.1	TIN72
0101 _B	P20.3	TIN61
0110 _B	Reserved	-
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_COSR2</i>	ADC
1011 _B	<i>tim2_muxout_4</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	<i>can_int[15]</i>	CAN
1110 _B	Reserved	-
1111 _B	Reserved	-
CH5SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.6	TIN15
0010 _B	P11.10	TIN99
0011 _B	P13.0	TIN91
0100 _B	P15.2	TIN73
0101 _B	P20.9	TIN65
0110 _B	Reserved	-

Generic Timer Module (GTM)

Table 26-74 TIM 2 Mapping for QFP-144 (cont'd)

0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_CISR2</i>	ADC
1011 _B	<i>tim2_muxout_5</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	Reserved	-
1110 _B	Reserved	-
1111 _B	Reserved	-
CH6SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.7	TIN16
0010 _B	P11.11	TIN100
0011 _B	P13.1	TIN92
0100 _B	P15.3	TIN74
0101 _B	P20.6	TIN62
0110 _B	P20.10	TIN66
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_COSR3</i>	ADC
1011 _B	<i>tim2_muxout_6</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	Reserved	-
1110 _B	Reserved	-
1111 _B	Reserved	-
CH7SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.8	TIN17
0010 _B	P11.12	TIN101
0011 _B	P13.2	TIN93

Generic Timer Module (GTM)

Table 26-74 TIM 2 Mapping for QFP-144 (cont'd)

0100 _B	P15.4	TIN75
0101 _B	P20.7	TIN63
0110 _B	P20.11	TIN67
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_CISR3</i>	ADC
1011 _B	<i>tim2_muxout_7</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	<i>eray_mt</i>	ERAY
1110 _B	Reserved	-
1111 _B	Reserved	-

Generic Timer Module (GTM)

Table 26-75 TIM 2 Mapping for QFP-176 / BGA-292

CH0SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.0	TIN9
0010 _B	P02.8	TIN8
0011 _B	P13.3	TIN94
0100 _B	P15.5	TIN76
0101 _B	P20.12	TIN68
0110 _B	P33.12	TIN34
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_COSR0</i>	ADC
1011 _B	<i>tim2_muxout_0</i>	DSADC trigger
1100 _B	<i>vadc_GIARBCNT</i>	ADC
1101 _B	Reserved	-
1110 _B	Reserved	-
1111 _B	Reserved	-
CH1SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.1	TIN10
0010 _B	P00.2	TIN11
0011 _B	P11.2	TIN95
0100 _B	P20.13	TIN69
0101 _B	P33.13	TIN35
0110 _B	P14.9	TIN151
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_CISR0</i>	ADC
1011 _B	<i>tim2_muxout_1</i>	DSADC trigger
1100 _B	<i>vadc_G3ARBCNT</i>	ADC

Generic Timer Module (GTM)

Table 26-75 TIM 2 Mapping for QFP-176 (cont'd)/ BGA-292 (cont'd)

1101 _B	<i>can_int[12]</i>	CAN
1110 _B	Reserved	-
1111 _B	Reserved	-
CH2SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.3	TIN12
0010 _B	P11.3	TIN96
0011 _B	P14.8	TIN88
0100 _B	P20.14	TIN70
0101 _B	P32.0	TIN36
0110 _B	Reserved	-
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_COSR1</i>	ADC
1011 _B	<i>tim2_muxout_2</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	<i>can_int[13]</i>	CAN
1110 _B	Reserved	-
1111 _B	Reserved	-
CH3SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.4	TIN13
0010 _B	P11.6	TIN97
0011 _B	P14.9	TIN89
0100 _B	P15.0	TIN71
0101 _B	P20.1	TIN60
0110 _B	Reserved	-
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-

Generic Timer Module (GTM)
Table 26-75 TIM 2 Mapping for QFP-176 (cont'd)/ BGA-292 (cont'd)

1010 _B	<i>vadc_CISR1</i>	ADC
1011 _B	<i>tim2_muxout_3</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	<i>can_int[14]</i>	CAN
1110 _B	Reserved	-
1111 _B	Reserved	-
CH4SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.5	TIN14
0010 _B	P11.9	TIN98
0011 _B	P14.10	TIN90
0100 _B	P15.1	TIN72
0101 _B	P20.3	TIN61
0110 _B	P21.0	TIN51
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_COSR2</i>	ADC
1011 _B	<i>tim2_muxout_4</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	<i>can_int[15]</i>	CAN
1110 _B	Reserved	-
1111 _B	Reserved	-
CH5SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.6	TIN15
0010 _B	P11.10	TIN99
0011 _B	P13.0	TIN91
0100 _B	P15.2	TIN73
0101 _B	P20.9	TIN65
0110 _B	P21.1	TIN52

Generic Timer Module (GTM)

Table 26-75 TIM 2 Mapping for QFP-176 (cont'd)/ BGA-292 (cont'd)

0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_CISR2</i>	ADC
1011 _B	<i>tim2_muxout_5</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	Reserved	-
1110 _B	Reserved	-
1111 _B	Reserved	-
CH6SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.7	TIN16
0010 _B	P11.11	TIN100
0011 _B	P13.1	TIN92
0100 _B	P15.3	TIN74
0101 _B	P20.6	TIN62
0110 _B	P20.10	TIN66
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_COSR3</i>	ADC
1011 _B	<i>tim2_muxout_6</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	Reserved	-
1110 _B	Reserved	-
1111 _B	Reserved	-
CH7SEL	Pad / Input	Name
0000 _B	'0'	-
0001 _B	P00.8	TIN17
0010 _B	P11.12	TIN101
0011 _B	P13.2	TIN93

Generic Timer Module (GTM)
Table 26-75 TIM 2 Mapping for QFP-176 (cont'd)/ BGA-292 (cont'd)

0100 _B	P15.4	TIN75
0101 _B	P20.7	TIN63
0110 _B	P20.11	TIN67
0111 _B	Reserved	-
1000 _B	Reserved	-
1001 _B	Reserved	-
1010 _B	<i>vadc_CISR3</i>	ADC
1011 _B	<i>tim2_muxout_7</i>	DSADC trigger
1100 _B	Reserved	-
1101 _B	<i>eray_mt</i>	ERAY
1110 _B	Reserved	-
1111 _B	Reserved	-

Generic Timer Module (GTM)

26.22.2.2 GTM to Port Control Registers

The outputs of the GTM TOM and ATOM modules are not connected directly to the port output path. Each TOM and ATOM is connectable to several port alternate inputs via an output multiplexer.

The output signals to the mux are called *gtm_tout_x*, where x is running number.

TOUTSELn (n = 0-9)

Timer Output Select Register (9FD30_H+n*4_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x = 0-15)	[x*2+1: x*2]	rw	<p>TOUT(n*16+x) Output Selection</p> <p>This bit defines which timer out is connected as TOUT(n*16+x). The mapping for each pin is defined by Table 26-67Table 26-68.</p> <p>00_B Timer A form Table 26-67Table 26-68 is connected as TOUT(n*16+x) to the ports</p> <p>01_B Timer B form Table 26-67Table 26-68 is connected as TOUT(n*16+x) to the ports</p> <p>10_B Timer C form Table 26-67Table 26-68 is connected as TOUT(n*16+x) to the ports</p> <p>11_B Timer D form Table 26-67Table 26-68 is connected as TOUT(n*16+x) to the ports</p> <p><i>Note: If TOUT(n*16+x) is not defined in Table 26-67Table 26-68 this bit field has to be treated as reserved.</i></p>

26.22.3 MSC Connections

The sections summarize the connections to the MSC modules for the TC26x B-Step. The MSC interfaces (MSC0 and MSC1) provide a serial communication link typically used to connect power switches or other peripheral devices.

The outputs of the signal names are defined in [Table 26-77](#) and [Table 26-78](#).

Table 26-76 GTM to MSC Output Multiplexer Configuration

control	Assigned GTM Output Set 1	Assigned GTM Output Set 2	Assigned GTM Output Set 3	Assigned GTM Output Set 4
00000 _B	TOM0_0	TOM1_0	Reserved	ATOM0_0
00001 _B	TOM0_1	TOM1_1	Reserved	ATOM0_1
00010 _B	TOM0_2	TOM1_2	Reserved	ATOM0_2
00011 _B	TOM0_3	TOM1_3	Reserved	ATOM0_3
00100 _B	TOM0_4	TOM1_4	Reserved	ATOM0_4
00101 _B	TOM0_5	TOM1_5	Reserved	ATOM0_5
00110 _B	TOM0_6	TOM1_6	Reserved	ATOM0_6
00111 _B	TOM0_7	TOM1_7	Reserved	ATOM0_7
01000 _B	TOM0_8	TOM1_8	Reserved	ATOM1_0
01001 _B	TOM0_9	TOM1_9	Reserved	ATOM1_1
01010 _B	TOM0_10	TOM1_10	Reserved	ATOM1_2
01011 _B	TOM0_11	TOM1_11	Reserved	ATOM1_3
01100 _B	TOM0_12	TOM1_12	Reserved	ATOM1_4
01101 _B	TOM0_13	TOM1_13	Reserved	ATOM1_5
01110 _B	TOM0_14	TOM1_14	Reserved	ATOM1_6
01111 _B	TOM0_15	TOM1_15	Reserved	ATOM1_7
10000 _B	ATOM0_0	ATOM1_0	Reserved	ATOM2_0
10001 _B	ATOM0_1	ATOM1_1	Reserved	ATOM2_1
10010 _B	ATOM0_2	ATOM1_2	Reserved	ATOM2_2
10011 _B	ATOM0_3	ATOM1_3	Reserved	ATOM2_3
10100 _B	ATOM0_4	ATOM1_4	Reserved	ATOM2_4
10101 _B	ATOM0_5	ATOM1_5	Reserved	ATOM2_5
10110 _B	ATOM0_6	ATOM1_6	Reserved	ATOM2_6
10111 _B	ATOM0_7	ATOM1_7	Reserved	ATOM2_7

Generic Timer Module (GTM)
Table 26-76 GTM to MSC Output Multiplexer Configuration (cont'd)

control	Assigned GTM Output Set 1	Assigned GTM Output Set 2	Assigned GTM Output Set 3	Assigned GTM Output Set 4
11000 _B	ATOM2_0	ATOM3_0	ATOM3_0	Reserved
11001 _B	ATOM2_1	ATOM3_1	ATOM3_1	Reserved
11010 _B	ATOM2_2	ATOM3_2	ATOM3_2	Reserved
11011 _B	ATOM2_3	ATOM3_3	ATOM3_3	Reserved
11100 _B	ATOM2_4	ATOM3_4	ATOM3_4	Reserved
11101 _B	ATOM2_5	ATOM3_5	ATOM3_5	Reserved
11110 _B	ATOM2_6	ATOM3_6	ATOM3_6	Reserved
11111 _B	ATOM2_7	ATOM3_7	ATOM3_7	Reserved

Table 26-77 GTM to MSC0 Connections

MSC Input	Mux output signal
ALTINL[15:0]	gtm_msc0altinl[15:0]
ALTINLE[15:0]	gtm_msc0altinh[15:0]
ALTINH[15:0]	gtm_msc0altinh[15:0]
ALTINHE[15:0]	gtm_msc0altinl[15:0]

Table 26-78 GTM to MSC1 Connections

MSC Input	Mux output signal
ALTINL[15:0]	gtm_msc1altinl[15:0]
ALTINLE[15:0]	gtm_msc1altinh[15:0]
ALTINH[15:0]	gtm_msc1altinh[15:0]
ALTINHE[15:0]	gtm_msc1altinl[15:0]

Generic Timer Module (GTM)

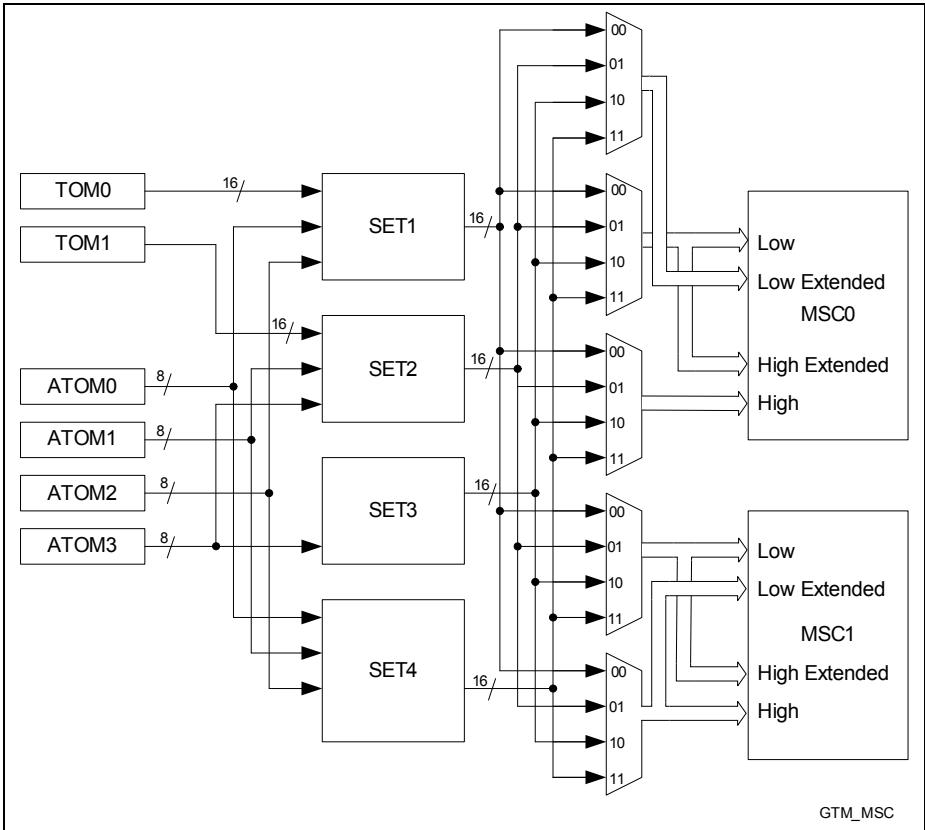


Figure 26-85 GTM to MSC connections

Generic Timer Module (GTM)

26.22.3.1 GTM to MSC Control Registers

MSCSET1CON0

MSC Set 1 Control 0 Register

 (9FF00_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL3				0		SEL2							
r		rw				r		rw							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL1				0		SEL0							
r		rw				r		rw							

Field	Bits	Type	Description
SEL0	[4:0]	rw	Set 1[0] Input Selection This bit field defines the GTM timer source configured as Set1 signal 0 out. For decoding see set 1 in Table 26-76 .
SEL1	[12:8]	rw	Set 1[1] Input Selection This bit field defines the GTM timer source configured as Set1 signal 1 out. For decoding see set 1 in Table 26-76 .
SEL2	[20:16]	rw	Set 1[2] Input Selection This bit field defines the GTM timer source configured as Set1 signal 2 out. For decoding see set 1 in Table 26-76 .
SEL3	[28:24]	rw	Set 1[3] Input Selection This bit field defines the GTM timer source configured as Set1 signal 3 out. For decoding see set 1 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET1CON1
MSC Set 1 Control 1 Register
(9FF04_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL7					0			SEL6					
r		rw					r			rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL5					0			SEL4					
r		rw					r			rw					

Field	Bits	Type	Description
SEL4	[4:0]	rw	Set 1[4] Input Selection This bit field defines the GTM timer source configured as Set1 signal 4 out. For decoding see set 1 in Table 26-76 .
SEL5	[12:8]	rw	Set 1[5] Input Selection This bit field defines the GTM timer source configured as Set1 signal 5 out. For decoding see set 1 in Table 26-76 .
SEL6	[20:16]	rw	Set 1[6] Input Selection This bit field defines the GTM timer source configured as Set1 signal 6 out. For decoding see set 1 in Table 26-76 .
SEL7	[28:24]	rw	Set 1[7] Input Selection This bit field defines the GTM timer source configured as Set1 signal 7 out. For decoding see set 1 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET1CON2
MSC Set 1 Control 2 Register
(9FF08_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL11						0		SEL10					
r		rw						r		rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL9						0		SEL8					
r		rw						r		rw					

Field	Bits	Type	Description
SEL8	[4:0]	rw	Set 1[8] Input Selection This bit field defines the GTM timer source configured as Set1 signal 8 out. For decoding see set 1 in Table 26-76 .
SEL9	[12:8]	rw	Set 1[9] Input Selection This bit field defines the GTM timer source configured as Set1 signal 9 out. For decoding see set 1 in Table 26-76 .
SEL10	[20:16]	rw	Set 1[10] Input Selection This bit field defines the GTM timer source configured as Set1 signal 10 out. For decoding see set 1 in Table 26-76 .
SEL11	[28:24]	rw	Set 1[11] Input Selection This bit field defines the GTM timer source configured as Set1 signal 11 out. For decoding see set 1 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET1CON3
MSC Set 1 Control 3 Register
(9FF0C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL15						0		SEL14					
r		rw						r		rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL13						0		SEL12					
r		rw						r		rw					

Field	Bits	Type	Description
SEL12	[4:0]	rw	Set 1[12] Input Selection This bit field defines the GTM timer source configured as Set1 signal 12 out. For decoding see set 1 in Table 26-76 .
SEL13	[12:8]	rw	Set 1[13] Input Selection This bit field defines the GTM timer source configured as Set1 signal 13 out. For decoding see set 1 in Table 26-76 .
SEL14	[20:16]	rw	Set 1[14] Input Selection This bit field defines the GTM timer source configured as Set1 signal 14 out. For decoding see set 1 in Table 26-76 .
SEL15	[28:24]	rw	Set 1[15] Input Selection This bit field defines the GTM timer source configured as Set1 signal 15 out. For decoding see set 1 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET2CON0
MSC Set 2 Control 0 Register
(9FF10_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL3						0		SEL2					
r		rw						r		rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL1						0		SEL0					
r		rw						r		rw					

Field	Bits	Type	Description
SEL0	[4:0]	rw	Set 2[0] Input Selection This bit field defines the GTM timer source configured as Set2 signal 0 out. For decoding see set 2 in Table 26-76 .
SEL1	[12:8]	rw	Set 2[1] Input Selection This bit field defines the GTM timer source configured as Set2 signal 1 out. For decoding see set 2 in Table 26-76 .
SEL2	[20:16]	rw	Set 2[2] Input Selection This bit field defines the GTM timer source configured as Set2 signal 2 out. For decoding see set 2 in Table 26-76 .
SEL3	[28:24]	rw	Set 2[3] Input Selection This bit field defines the GTM timer source configured as Set2 signal 3 out. For decoding see set 2 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET2CON1
MSC Set 2 Control 1 Register
(9FF14_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL7						0		SEL6					
r		rw						r		rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL5						0		SEL4					
r		rw						r		rw					

Field	Bits	Type	Description
SEL4	[4:0]	rw	Set 2[4] Input Selection This bit field defines the GTM timer source configured as Set2 signal 4 out. For decoding see set 2 in Table 26-76 .
SEL5	[12:8]	rw	Set 2[5] Input Selection This bit field defines the GTM timer source configured as Set2 signal 5 out. For decoding see set 2 in Table 26-76 .
SEL6	[20:16]	rw	Set 2[6] Input Selection This bit field defines the GTM timer source configured as Set2 signal 6 out. For decoding see set 2 in Table 26-76 .
SEL7	[28:24]	rw	Set 2[7] Input Selection This bit field defines the GTM timer source configured as Set2 signal 7 out. For decoding see set 2 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET2CON2
MSC Set 2 Control 2 Register
(9FF18_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL11						0		SEL10					
r		rw						r		rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL9						0		SEL8					
r		rw						r		rw					

Field	Bits	Type	Description
SEL8	[4:0]	rw	Set 2[8] Input Selection This bit field defines the GTM timer source configured as Set2 signal 8 out. For decoding see set 2 in Table 26-76 .
SEL9	[12:8]	rw	Set 2[9] Input Selection This bit field defines the GTM timer source configured as Set2 signal 9 out. For decoding see set 2 in Table 26-76 .
SEL10	[20:16]	rw	Set 2[10] Input Selection This bit field defines the GTM timer source configured as Set2 signal 10 out. For decoding see set 2 in Table 26-76 .
SEL11	[28:24]	rw	Set 2[11] Input Selection This bit field defines the GTM timer source configured as Set2 signal 11 out. For decoding see set 2 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET2CON3
MSC Set 2 Control 3 Register
(9FF1C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL15						0		SEL14					
r		rw						r		rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL13						0		SEL12					
r		rw						r		rw					

Field	Bits	Type	Description
SEL12	[4:0]	rw	Set 2[12] Input Selection This bit field defines the GTM timer source configured as Set2 signal 12 out. For decoding see set 2 in Table 26-76 .
SEL13	[12:8]	rw	Set 2[13] Input Selection This bit field defines the GTM timer source configured as Set2 signal 13 out. For decoding see set 2 in Table 26-76 .
SEL14	[20:16]	rw	Set 2[14] Input Selection This bit field defines the GTM timer source configured as Set2 signal 14 out. For decoding see set 2 in Table 26-76 .
SEL15	[28:24]	rw	Set 2[15] Input Selection This bit field defines the GTM timer source configured as Set2 signal 15 out. For decoding see set 2 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET3CON0
MSC Set 3 Control 0 Register
(9FF20_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL3						0		SEL2					
r		rw						r		rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL1						0		SEL0					
r		rw						r		rw					

Field	Bits	Type	Description
SEL0	[4:0]	rw	Set 3[0] Input Selection This bit field defines the GTM timer source configured as Set3 signal 0 out. For decoding see set 3 in Table 26-76 .
SEL1	[12:8]	rw	Set 3[1] Input Selection This bit field defines the GTM timer source configured as Set3 signal 1 out. For decoding see set 3 in Table 26-76 .
SEL2	[20:16]	rw	Set 3[2] Input Selection This bit field defines the GTM timer source configured as Set3 signal 2 out. For decoding see set 3 in Table 26-76 .
SEL3	[28:24]	rw	Set 3[3] Input Selection This bit field defines the GTM timer source configured as Set3 signal 3 out. For decoding see set 3 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET3CON1
MSC Set 3 Control 1 Register
(9FF24_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL7					0			SEL6					
r		rw					r			rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL5					0			SEL4					
r		rw					r			rw					

Field	Bits	Type	Description
SEL4	[4:0]	rw	Set 3[4] Input Selection This bit field defines the GTM timer source configured as Set3 signal 4 out. For decoding see set 3 in Table 26-76 .
SEL5	[12:8]	rw	Set 3[5] Input Selection This bit field defines the GTM timer source configured as Set3 signal 5 out. For decoding see set 3 in Table 26-76 .
SEL6	[20:16]	rw	Set 3[6] Input Selection This bit field defines the GTM timer source configured as Set3 signal 6 out. For decoding see set 3 in Table 26-76 .
SEL7	[28:24]	rw	Set 3[7] Input Selection This bit field defines the GTM timer source configured as Set3 signal 7 out. For decoding see set 3 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET3CON2
MSC Set 3 Control 2 Register
(9FF28_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL11						0		SEL10					
r		rw						r		rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL9						0		SEL8					
r		rw						r		rw					

Field	Bits	Type	Description
SEL8	[4:0]	rw	Set 3[8] Input Selection This bit field defines the GTM timer source configured as Set3 signal 8 out. For decoding see set 3 in Table 26-76 .
SEL9	[12:8]	rw	Set 3[9] Input Selection This bit field defines the GTM timer source configured as Set3 signal 9 out. For decoding see set 3 in Table 26-76 .
SEL10	[20:16]	rw	Set 3[10] Input Selection This bit field defines the GTM timer source configured as Set3 signal 10 out. For decoding see set 3 in Table 26-76 .
SEL11	[28:24]	rw	Set 3[11] Input Selection This bit field defines the GTM timer source configured as Set3 signal 11 out. For decoding see set 3 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)

MSCSET3CON3
MSC Set 3 Control 3 Register
(9FF2C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0		SEL15						0			SEL14					
r		rw						r			rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0		SEL13						0			SEL12					
r		rw						r			rw					

Field	Bits	Type	Description
SEL12	[4:0]	rw	Set 3[12] Input Selection This bit field defines the GTM timer source configured as Set3 signal 12 out. For decoding see set 3 in Table 26-76 .
SEL13	[12:8]	rw	Set 3[13] Input Selection This bit field defines the GTM timer source configured as Set3 signal 13 out. For decoding see set 3 in Table 26-76 .
SEL14	[20:16]	rw	Set 3[14] Input Selection This bit field defines the GTM timer source configured as Set3 signal 14 out. For decoding see set 3 in Table 26-76 .
SEL15	[28:24]	rw	Set 3[15] Input Selection This bit field defines the GTM timer source configured as Set3 signal 15 out. For decoding see set 3 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET4CON0
MSC Set 4 Control 0 Register
(9FF30_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL3						0		SEL2					
r		rw						r		rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL1						0		SEL0					
r		rw						r		rw					

Field	Bits	Type	Description
SEL0	[4:0]	rw	Set 4[0] Input Selection This bit field defines the GTM timer source configured as Set4 signal 0 out. For decoding see set 4 in Table 26-76 .
SEL1	[12:8]	rw	Set 4[1] Input Selection This bit field defines the GTM timer source configured as Set4 signal 1 out. For decoding see set 4 in Table 26-76 .
SEL2	[20:16]	rw	Set 4[2] Input Selection This bit field defines the GTM timer source configured as Set4 signal 2 out. For decoding see set 4 in Table 26-76 .
SEL3	[28:24]	rw	Set 4[3] Input Selection This bit field defines the GTM timer source configured as Set4 signal 3 out. For decoding see set 4 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET4CON1
MSC Set 4 Control 1 Register
(9FF34_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL7					0		SEL6						
r		rw					r		rw						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL5					0		SEL4						
r		rw					r		rw						

Field	Bits	Type	Description
SEL4	[4:0]	rw	Set 4[4] Input Selection This bit field defines the GTM timer source configured as Set4 signal 4 out. For decoding see set 4 in Table 26-76 .
SEL5	[12:8]	rw	Set 4[5] Input Selection This bit field defines the GTM timer source configured as Set4 signal 5 out. For decoding see set 4 in Table 26-76 .
SEL6	[20:16]	rw	Set 4[6] Input Selection This bit field defines the GTM timer source configured as Set4 signal 6 out. For decoding see set 4 in Table 26-76 .
SEL7	[28:24]	rw	Set 4[7] Input Selection This bit field defines the GTM timer source configured as Set4 signal 7 out. For decoding see set 4 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET4CON2
MSC Set 4 Control 2 Register
(9FF38_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL11						0		SEL10					
r		rw						r		rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL9						0		SEL8					
r		rw						r		rw					

Field	Bits	Type	Description
SEL8	[4:0]	rw	Set 4[8] Input Selection This bit field defines the GTM timer source configured as Set4 signal 8 out. For decoding see set 4 in Table 26-76 .
SEL9	[12:8]	rw	Set 4[9] Input Selection This bit field defines the GTM timer source configured as Set4 signal 9 out. For decoding see set 4 in Table 26-76 .
SEL10	[20:16]	rw	Set 4[10] Input Selection This bit field defines the GTM timer source configured as Set4 signal 10 out. For decoding see set 4 in Table 26-76 .
SEL11	[28:24]	rw	Set 4[11] Input Selection This bit field defines the GTM timer source configured as Set4 signal 11 out. For decoding see set 4 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSCSET4CON3
MSC Set 4 Control 3 Register
(9FF3C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SEL15						0			SEL14				
r		rw						r			rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		SEL13						0			SEL12				
r		rw						r			rw				

Field	Bits	Type	Description
SEL12	[4:0]	rw	Set 4[12] Input Selection This bit field defines the GTM timer source configured as Set4 signal 12 out. For decoding see set 4 in Table 26-76 .
SEL13	[12:8]	rw	Set 4[13] Input Selection This bit field defines the GTM timer source configured as Set4 signal 13 out. For decoding see set 4 in Table 26-76 .
SEL14	[20:16]	rw	Set 4[14] Input Selection This bit field defines the GTM timer source configured as Set4 signal 14 out. For decoding see set 4 in Table 26-76 .
SEL15	[28:24]	rw	Set 4[15] Input Selection This bit field defines the GTM timer source configured as Set4 signal 15 out. For decoding see set 4 in Table 26-76 .
0	[7:5], [15:13], [23:21], [31:29]	r	Reserved Read as 0; should be written with 0.

Generic Timer Module (GTM)
MSC0INLCON
MSC0 Input Low Control Register (9FF60_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SEL_x (x = 0-15)	[x*2+1: x*2]	rw	GTM MSC0 Low x Output Selection GTM output gtm_msc0altinl[x] is controller by the timer output defined by: 00 _B Set1 signal x 01 _B Set2 signal x 10 _B Set3 signal x 11 _B Set4 signal x This output is connect to MSC0 ALTINL[x] and MSC0 ALTINHE[x]

MSC0INHCON
MSC0 Input High Control Register (9FF64_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Generic Timer Module (GTM)

Field	Bits	Type	Description
SELx (x = 0-15)	[x*2+1: x*2]	rw	GTM MSC0 High x Output Selection GTM output gtm_msc0altinh[x] is controller by the timer output defined by: 00 _B Set1 signal x 01 _B Set2 signal x 10 _B Set3 signal x 11 _B Set4 signal x This output is connect to MSC0 ALTINH[x]

MSC0INLEXTCON
MSC0 Input Low Extended Control Register
(9FF70_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x = 0-15)	[x*2+1: x*2]	rw	GTM MSC0 Low Extended x Output Selection GTM output gtm_msc0altinextl[x] is controller by the timer output defined by: 00 _B Set1 signal x 01 _B Set2 signal x 10 _B Set3 signal x 11 _B Set4 signal x This output is connect to MSC0 ALTINLE[x]

Generic Timer Module (GTM)
MSC1INLCON
MSC1 Input Low Control Register (9FF68_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SEL_x (x = 0-15)	[x*2+1: x*2]	rw	GTM MSC1 Low x Output Selection GTM output gtm_msc1altin[x] is controller by the timer output defined by: 00 _B Set1 signal x 01 _B Set2 signal x 10 _B Set3 signal x 11 _B Set4 signal x This output is connect to MSC1 ALTINL[x] and MSC1 ALTINHE[x]

MSC1INHCON
MSC1 Input High Control Register (9FF6C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Generic Timer Module (GTM)

Field	Bits	Type	Description
SELx (x = 0-15)	[x*2+1: x*2]	rw	GTM MSC1 High x Output Selection GTM output gtm_msc1altinh[x] is controller by the timer output defined by: 00 _B Set1 signal x 01 _B Set2 signal x 10 _B Set3 signal x 11 _B Set4 signal x This output is connect to MSC1 ALTINH[x] and MSC1 ALTINLE[x]

26.22.4 DSADC Connections

This register defines the DSADC to GTM connections of TC26x B-Step.

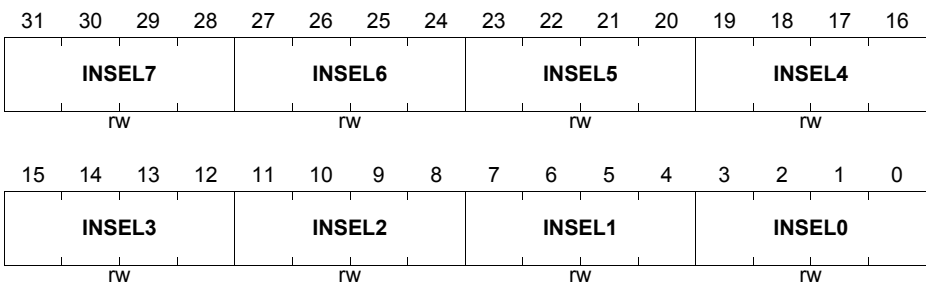
The 16 to 1 multiplexer is implemented once per TIM0 / 1 / 2 module / channel. The input signal from the DSADC is called *dsadc_x_trig_in_y* x = number of DSADC, y = SRM / SAUL / SBLL. The output of the multiplexer is called *timy_muxout_x*, (y = 0,1,2)(x = 0...7).

DSADCINSEL0

DSADC Input Select 0 Register

(9FD7C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
INSEL0	[3:0]	rw	<p>In Selection for DSADCn GTM connection</p> <p>This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM0 channel 0</p> <p>0000_Bconnected DSADC0_SRM0</p> <p>0001_Breserved, do not use this combination</p> <p>0010_Bconnected DSADC2_SRM0</p> <p>0011_Bconnected DSADC3_SRM0</p> <p>0100_Breserved, do not use this combination</p> <p>0101_Breserved, do not use this combination</p> <p>0110_Breserved, do not use this combination</p> <p>0111_Breserved, do not use this combination</p> <p>1000_Breserved, do not use this combination</p> <p>1001_Breserved, do not use this combination</p> <p>1010_Bconnected DSADC0_SAUL</p> <p>1011_Bconnected DSADC0_SBLL</p> <p>1100_Breserved, do not use this combination</p> <p>1101_Breserved, do not use this combination</p> <p>1110_Breserved, do not use this combination</p> <p>1111_Bdrive '0'</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
INSEL1	[7:4]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM0 channel 1</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Breserved, do not use this combination 1011_Breserved, do not use this combination 1100_Bconnected DSADC0_SAUL 1101_Bconnected DSADC0_SBLL 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>
INSEL2	[11:8]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM0 channel 2</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Bconnected DSADC2_SAUL 1011_Bconnected DSADC2_SBLL 1100_Bconnected DSADC3_SAUL 1101_Bconnected DSADC3_SBLL 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
INSEL3	[15:12]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM0 channel 3</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Bconnected DSADC3_SAUL 1011_Bconnected DSADC3_SBLI 1100_Bconnected DSADC2_SAUL 1101_Bconnected DSADC2_SBLI 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>
INSEL4	[19:16]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM0 channel 4</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Breserved, do not use this combination 1011_Breserved, do not use this combination 1100_Breserved, do not use this combination 1101_Breserved, do not use this combination 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
INSEL5	[23:20]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM0 channel 5</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Breserved, do not use this combination 1011_Breserved, do not use this combination 1100_Breserved, do not use this combination 1101_Breserved, do not use this combination 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>
INSEL6	[27:24]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM0 channel 6</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Breserved, do not use this combination 1011_Breserved, do not use this combination 1100_Breserved, do not use this combination 1101_Breserved, do not use this combination 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
INSEL7	[31:28]	rw	In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM0 channel 7 0000 _B connected DSADC0_SRM0 0001 _B reserved, do not use this combination 0010 _B connected DSADC2_SRM0 0011 _B connected DSADC3_SRM0 0100 _B reserved, do not use this combination 0101 _B reserved, do not use this combination 0110 _B reserved, do not use this combination 0111 _B reserved, do not use this combination 1000 _B reserved, do not use this combination 1001 _B reserved, do not use this combination 1010 _B reserved, do not use this combination 1011 _B reserved, do not use this combination 1100 _B reserved, do not use this combination 1101 _B reserved, do not use this combination 1110 _B reserved, do not use this combination 1111 _B drive '0'

DSADCINSEL1
DSADC Input Select 1 Register
(9FD80_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INSEL7				INSEL6				INSEL5				INSEL4			
rw				rw				rw				rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INSEL3				INSEL2				INSEL1				INSEL0			
rw				rw				rw				rw			

Generic Timer Module (GTM)

Field	Bits	Type	Description
INSEL0	[3:0]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM1 channel 0</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Bconnected DSADC0_SAUL 1011_Bconnected DSADC0_SBILL 1100_Breserved, do not use this combination 1101_Breserved, do not use this combination 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
INSEL1	[7:4]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM1 channel 1</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Breserved, do not use this combination 1011_Breserved, do not use this combination 1100_Bconnected DSADC0_SAUL 1101_Bconnected DSADC0_SBLL 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>
INSEL2	[11:8]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM1 channel 2</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Bconnected DSADC2_SAUL 1011_Bconnected DSADC2_SBLL 1100_Bconnected DSADC3_SAUL 1101_Bconnected DSADC3_SBLL 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
INSEL3	[15:12]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM1 channel 3</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Bconnected DSADC3_SAUL 1011_Bconnected DSADC3_SBLI 1100_Bconnected DSADC2_SAUL 1101_Bconnected DSADC2_SBLI 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>
INSEL4	[19:16]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM1 channel 4</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Breserved, do not use this combination 1011_Breserved, do not use this combination 1100_Breserved, do not use this combination 1101_Breserved, do not use this combination 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>

Generic Timer Module (GTM)

Field	Bits	Type	Description
INSEL5	[23:20]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM1 channel 5</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Breserved, do not use this combination 1011_Breserved, do not use this combination 1100_Breserved, do not use this combination 1101_Breserved, do not use this combination 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>
INSEL6	[27:24]	rw	<p>In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM1 channel 6</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Breserved, do not use this combination 1011_Breserved, do not use this combination 1100_Breserved, do not use this combination 1101_Breserved, do not use this combination 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>

Generic Timer Module (GTM)

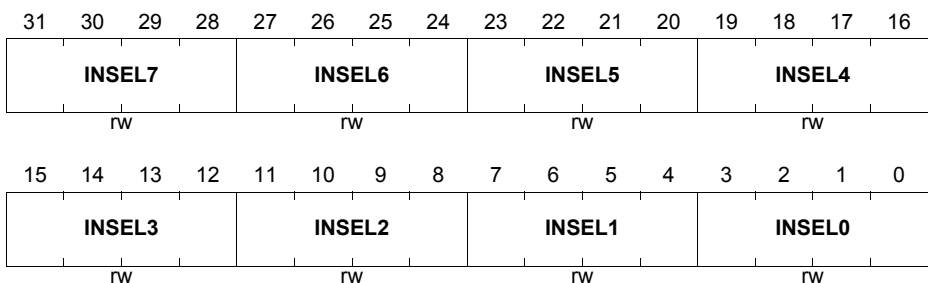
Field	Bits	Type	Description
INSEL7	[31:28]	rw	<p>In Selection for DSADCN GTM connection This bit field defines which DSADCN output is connected if the channel input mux is configured as DSADC input for TIM1 channel 7</p> <p>0000_Bconnected DSADC0_SRM0 0001_Breserved, do not use this combination 0010_Bconnected DSADC2_SRM0 0011_Bconnected DSADC3_SRM0 0100_Breserved, do not use this combination 0101_Breserved, do not use this combination 0110_Breserved, do not use this combination 0111_Breserved, do not use this combination 1000_Breserved, do not use this combination 1001_Breserved, do not use this combination 1010_Breserved, do not use this combination 1011_Breserved, do not use this combination 1100_Breserved, do not use this combination 1101_Breserved, do not use this combination 1110_Breserved, do not use this combination 1111_Bdrive '0'</p>

DSADCINSEL2

DSADC Input Select 2 Register

(9FD84_H)

Reset Value: 0000 0000_H



Generic Timer Module (GTM)

Field	Bits	Type	Description
INSELx (x = 0-7)	[x*4+3: x*4]	rw	In Selection for DSADCn GTM connection This bit field defines which DSADCn output is connected if the channel input mux is configured as DSADC input for TIM2 channel x 0000 _B connected DSADC0_SRM0 0001 _B reserved, do not use this combination 0010 _B connected DSADC2_SRM0 0011 _B connected DSADC3_SRM0 0100 _B reserved, do not use this combination 0101 _B reserved, do not use this combination 0110 _B reserved, do not use this combination 0111 _B reserved, do not use this combination 1000 _B reserved, do not use this combination 1001 _B reserved, do not use this combination 1010 _B reserved, do not use this combination 1011 _B reserved, do not use this combination 1100 _B reserved, do not use this combination 1101 _B reserved, do not use this combination 1110 _B reserved, do not use this combination 1111 _B drive '0'

Register DSADCOUTSELxy defines the GTM to DSADC connections of TC26x B-Step. The DSADC trigger inputs (one per DSADC) 8 to 1 multiplexer is implemented twice per DSADC module / channel. The output signal to the DSADC is called *dsadcx_trig0* and *dsadcx_trig1*, The inputs of the multiplexer are called *dsadc_x_muxin[7:0]*, x = number of DSADCs.

Which of the possible trigger inputs is used by the DSADC can be selected in the VADC module.

Generic Timer Module (GTM)

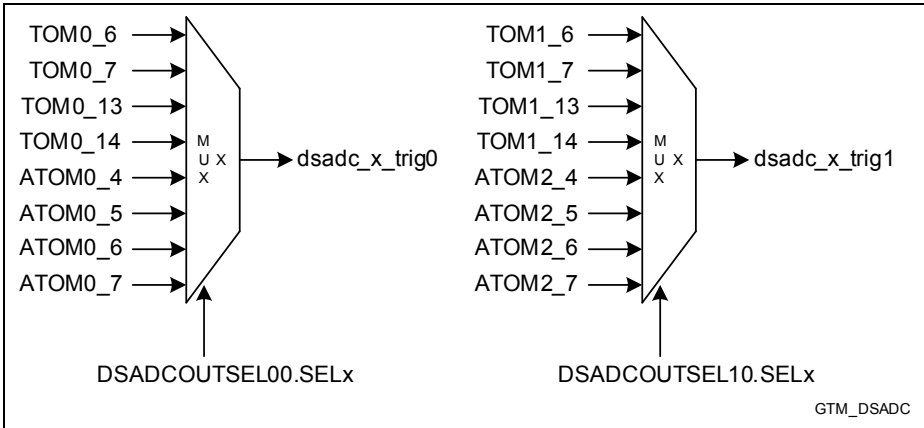


Figure 26-86 DSADC Output Multiplexers

DSADCOUTSEL00

DSADC Output Select 00 Register (9FD88_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			0							0	0	0			
			r							rw	r	rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SEL3	0	SEL2	0	0	0	0	SEL0							
r	rw	r	rw	r	rw	r	rw	r	rw	r	rw				

Generic Timer Module (GTM)

Field	Bits	Type	Description
SEL0	[2:0]	rw	Output Selection for DSADC0 GTM connection This bit field defines which TOM / ATOM channel output is used as DSADC0 trigger 0. 000 _B connected to TOM0_6 001 _B connected to TOM0_7 010 _B connected to TOM0_13 011 _B connected to TOM0_14 100 _B connected to ATOM0_4 101 _B connected to ATOM0_5 110 _B connected to ATOM0_6 111 _B connected to ATOM0_7
SEL2	[10:8]	rw	Output Selection for DSADC2 GTM connection This bit field defines which TOM / ATOM channel output is used as DSADC2 trigger 0. 000 _B connected to TOM0_6 001 _B connected to TOM0_7 010 _B connected to TOM0_13 011 _B connected to TOM0_14 100 _B connected to ATOM0_4 101 _B connected to ATOM0_5 110 _B connected to ATOM0_6 111 _B connected to ATOM0_7
SEL3	[14:12]	rw	Output Selection for DSADC3 GTM connection This bit field defines which TOM / ATOM channel output is used as DSADC3 trigger 0. 000 _B connected to TOM0_6 001 _B connected to TOM0_7 010 _B connected to TOM0_13 011 _B connected to TOM0_14 100 _B connected to ATOM0_4 101 _B connected to ATOM0_5 110 _B connected to ATOM0_6 111 _B connected to ATOM0_7
0	[6:4], [18:16], [22:20]	rw	Reserved Should be written with 0.

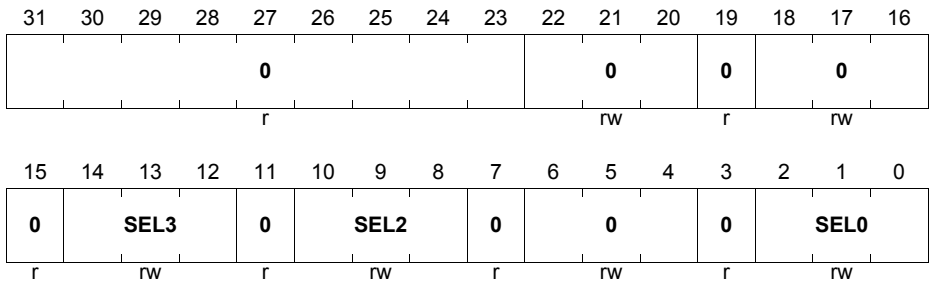
Generic Timer Module (GTM)

Field	Bits	Type	Description
0	3, 7, 11, 15, 19, [31:23]	r	Reserved Read as 0; should be written with 0.

DSADCOUTSEL10

DSADC Output Select 10 Register (9FD90_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SEL0	[2:0]	rw	Output Selection for DSADC0 GTM connection This bit field defines which TOM / ATOM channel output is used as DSADC0 trigger 1. 000 _B connected to TOM1_6 001 _B connected to TOM1_7 010 _B connected to TOM1_13 011 _B connected to TOM1_14 100 _B connected to ATOM2_4 101 _B connected to ATOM2_5 110 _B connected to ATOM2_6 111 _B connected to ATOM2_7

Generic Timer Module (GTM)

Field	Bits	Type	Description
SEL2	[10:8]	rw	Output Selection for DSADC2 GTM connection This bit field defines which TOM / ATOM channel output is used as DSADC2 trigger 1. 000 _B connected to TOM1_6 001 _B connected to TOM1_7 010 _B connected to TOM1_13 011 _B connected to TOM1_14 100 _B connected to ATOM2_4 101 _B connected to ATOM2_5 110 _B connected to ATOM2_6 111 _B connected to ATOM2_7
SEL3	[14:12]	rw	Output Selection for DSADC3 GTM connection This bit field defines which TOM / ATOM channel output is used as DSADC3 trigger 1. 000 _B connected to TOM1_6 001 _B connected to TOM1_7 010 _B connected to TOM1_13 011 _B connected to TOM1_14 100 _B connected to ATOM2_4 101 _B connected to ATOM2_5 110 _B connected to ATOM2_6 111 _B connected to ATOM2_7
0	[6:4], [18:16], [22:20]	rw	Reserved Should be written with 0.
0	3, 7, 11, 15, 19, [31:23]	r	Reserved Read as 0; should be written with 0.

26.22.5 ADC Connections

This register defines the GTM to ADC connections of TC26x B-Step.

The ADC trigger inputs 16 to 1 multiplexer is implemented twice per ADC module / channel. The output signal to the ADC is called *adcx_trig0* and *adcx_trig1*, The inputs of the multiplexer are called *adc_x_muxin[7:0]*, x = number of ADCs.

Mapping of ADC Triggers

Table 26-79 ADC0 / 1 / 2 Timer Triggers

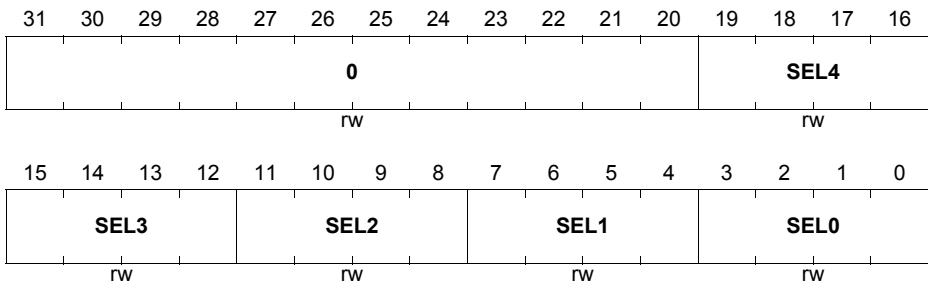
SEL0 / 1 / 2	Trigger 0	Trigger 1
0000 _B	No trigger is generated	No trigger is generated
0001 _B	TOM0 Channel 6 output	TOM1 Channel 6 output
0010 _B	TOM0 Channel 7 output	TOM1 Channel 7 output
0011 _B	TOM0 Channel 13 output	TOM1 Channel 13 output
0100 _B	TOM0 Channel 14 output	TOM1 Channel 14 output
0101 _B	ATOM0 Channel 4 output	ATOM1 Channel 4 output
0110 _B	ATOM0 Channel 5 output	ATOM1 Channel 5 output
0111 _B	ATOM0 Channel 6 output	ATOM1 Channel 6 output
1000 _B	ATOM0 Channel 7 output	ATOM1 Channel 7 output
1001 _B	ATOM1 Channel 4 output	TOM0 Channel 14 output
1010 _B	ATOM1 Channel 5 output	TOM0 Channel 15 output
1011 _B	ATOM1 Channel 6 output	ATOM0 Channel 4 output
1100 _B	ATOM1 Channel 7 output	ATOM0 Channel 5 output
1101 _B	Reserved	ATOM0 Channel 6 output
1110 _B	Reserved	ATOM0 Channel 7 output
1111 _B	Reserved	Reserved

Table 26-80 ADC3 / 4 Timer Triggers

SEL3 / 4	Trigger 0	Trigger 1
0000 _B	No trigger is generated	No trigger is generated
0001 _B	TOM0 Channel 6 output	TOM1 Channel 6 output
0010 _B	TOM0 Channel 7 output	TOM1 Channel 7 output
0011 _B	TOM0 Channel 13 output	TOM1 Channel 13 output
0100 _B	TOM0 Channel 14 output	TOM1 Channel 14 output

Generic Timer Module (GTM)
Table 26-80 ADC3 / 4 Timer Triggers (cont'd)

SEL3 / 4	Trigger 0	Trigger 1
0101 _B	ATOM0 Channel 4 output	ATOM1 Channel 4 output
0110 _B	ATOM0 Channel 5 output	ATOM1 Channel 5 output
0111 _B	ATOM0 Channel 6 output	ATOM1 Channel 6 output
1000 _B	ATOM0 Channel 7 output	ATOM1 Channel 7 output
1001 _B	TOM1 Channel 6 output	TOM1 Channel 14 output
1010 _B	TOM1 Channel 7 output	TOM1 Channel 15 output
1011 _B	ATOM2 Channel 4 output	ATOM2 Channel 6 output
1100 _B	ATOM2 Channel 5 output	ATOM2 Channel 7 output
1101 _B	ATOM3 Channel 4 output	ATOM3 Channel 6 output
1110 _B	ATOM3 Channel 5 output	ATOM3 Channel 7 output
1111 _B	Reserved	Reserved

ADCTRIG0OUT0
ADC Trigger 0 Output Select 0 Register(9FDB0_H)
Reset Value: 0000 0000_H


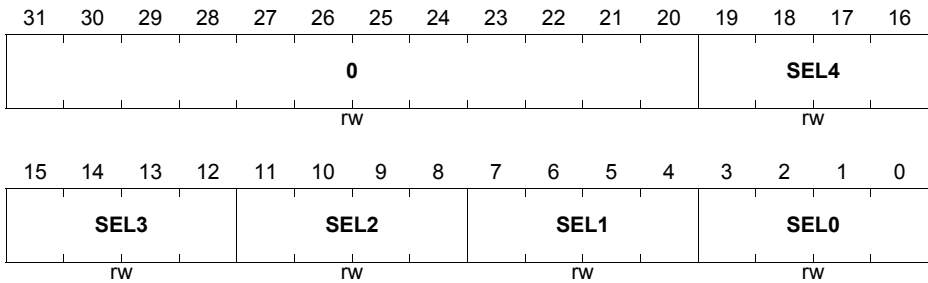
Field	Bits	Type	Description
SELx (x = 0-4)	[x*4+3: x*4]	rw	Output Selection for ADCx GTM connection This bit field defines which TOM / ATOM channel output is used as ADCx trigger 0. The decoding is defined by Table 26-79 or Table 26-80 depending on the ADC.
0	[31:20]	rw	Reserved Should be written with 0.

Generic Timer Module (GTM)

ADCTRIG1OUT0

ADC Trigger 1 Output Select 0 Register(9FDB8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SELx (x = 0-4)	[x*4+3: x*4]	rw	Output Selection for ADCx GTM connection This bit field defines which TOM / ATOM channel output is used as ADCx trigger 1. The decoding is defined by Table 26-79 or Table 26-80 depending on the ADC.
0	[31:20]	rw	Reserved Should be written with 0.

26.22.6 SENT Connections

This register defines the GTM to SENT connections of TC26x B-Step.

The trigger generation 16 to 1 multiplexer is implemented twice per ADC module / channel. The output signal to the ADC is called *adcx_trig0* and *adcx_trig1*. As the SENT channels overlay and replace ADC channels the ADC triggers will be also reused for the SENT channels. Therefore no additional outputs for separate operation are defined here.

adcx_trig0 is connected to *sent_trig_x*.

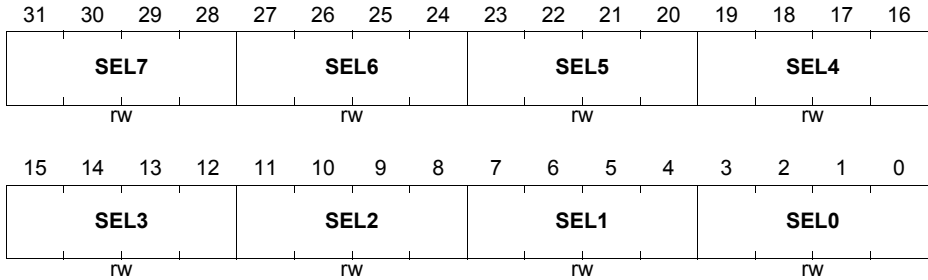
26.22.7 CAN Connection

CANOUTSEL

CAN Output Select Register

(9FDA0_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SEL0	[3:0]	rw	Output Selection for CAN GTM connection This bit field defines which TOM / ATOM channel output is used as CAN node trigger 0. The decoding is defined by Table 26-81
SEL1	[7:4]	rw	Output Selection for CAN GTM connection This bit field defines which TOM / ATOM channel output is used as CAN node trigger 1. The decoding is defined by Table 26-81
SEL2	[11:8]	rw	Output Selection for CAN GTM connection This bit field defines which TOM / ATOM channel output is used as CAN node trigger 2. The decoding is defined by Table 26-82
SEL3	[15:12]	rw	Output Selection for CAN GTM connection This bit field defines which TOM / ATOM channel output is used as CAN node trigger 3. The decoding is defined by Table 26-82
SEL4	[19:16]	rw	Output Selection for CAN GTM connection This bit field defines which TOM / ATOM channel output is used as CAN node trigger 4. The decoding is defined by Table 26-83

Generic Timer Module (GTM)

Field	Bits	Type	Description
SEL5	[23:20]	rw	Output Selection for CAN GTM connection This bit field defines which TOM / ATOM channel output is used as CAN node trigger 5. The decoding is defined by Table 26-83
SEL6	[27:24]	rw	Output Selection for CAN GTM connection This bit field defines which TOM / ATOM channel output is used as CAN node trigger 6. The decoding is defined by Table 26-84
SEL7	[31:28]	rw	Output Selection for CAN GTM connection This bit field defines which TOM / ATOM channel output is used as CAN node trigger 7. The decoding is defined by Table 26-84

Mapping of CAN Triggers
Table 26-81 CAN Timer Triggers

SEL0 / 1	Trigger 0	Trigger 1
0000 _B	TOM1 Channel 4 output	TOM1 Channel 4 output
0001 _B	TOM1 Channel 5 output	TOM1 Channel 5 output
0010 _B	TOM1 Channel 11 output	TOM1 Channel 11 output
0011 _B	TOM1 Channel 12 output	TOM1 Channel 12 output
0100 _B	ATOM2 Channel 0 output	ATOM2 Channel 0 output
0101 _B	ATOM2 Channel 1 output	ATOM2 Channel 1 output
0110 _B	ATOM2 Channel 2 output	ATOM2 Channel 2 output
0111 _B	ATOM2 Channel 3 output	ATOM2 Channel 3 output
1000 _B	TOM0 Channel 6 output	TOM1 Channel 6 output
1001 _B	TOM0 Channel 7 output	TOM1 Channel 7 output
1010 _B	TOM0 Channel 13 output	TOM1 Channel 13 output
1011 _B	TOM0 Channel 14 output	TOM1 Channel 14 output
1100 _B	ATOM0 Channel 4 output	ATOM1 Channel 4 output
1101 _B	ATOM0 Channel 5 output	ATOM1 Channel 5 output
1110 _B	ATOM0 Channel 6 output	ATOM1 Channel 6 output
1111 _B	ATOM0 Channel 7 output	ATOM1 Channel 7 output

Table 26-82 CAN Timer Triggers

SEL2 / 3	Trigger 2	Trigger 3
0000 _B	TOM1 Channel 4 output	TOM1 Channel 4 output
0001 _B	TOM1 Channel 5 output	TOM1 Channel 5 output
0010 _B	TOM1 Channel 11 output	TOM1 Channel 11 output
0011 _B	TOM1 Channel 12 output	TOM1 Channel 12 output
0100 _B	ATOM2 Channel 0 output	ATOM2 Channel 0 output
0101 _B	ATOM2 Channel 1 output	ATOM2 Channel 1 output
0110 _B	ATOM2 Channel 2 output	ATOM2 Channel 2 output
0111 _B	ATOM2 Channel 3 output	ATOM2 Channel 3 output
1000 _B	Reserved	Reserved
1001 _B	Reserved	Reserved
1010 _B	Reserved	Reserved
1011 _B	Reserved	Reserved
1100 _B	Reserved	Reserved
1101 _B	Reserved	Reserved
1110 _B	Reserved	Reserved
1111 _B	Reserved	Reserved

Table 26-83 CAN Timer Triggers

SEL4 / 5	Trigger 4	Trigger 5
0000 _B	TOM1 Channel 4 output	TOM1 Channel 4 output
0001 _B	TOM1 Channel 5 output	TOM1 Channel 5 output
0010 _B	TOM1 Channel 11 output	TOM1 Channel 11 output
0011 _B	TOM1 Channel 12 output	TOM1 Channel 12 output
0100 _B	ATOM2 Channel 0 output	ATOM2 Channel 0 output
0101 _B	ATOM2 Channel 1 output	ATOM2 Channel 1 output
0110 _B	ATOM2 Channel 2 output	ATOM2 Channel 2 output
0111 _B	ATOM2 Channel 3 output	ATOM2 Channel 3 output
1000 _B	Reserved	Reserved
1001 _B	Reserved	Reserved
1010 _B	Reserved	Reserved

Generic Timer Module (GTM)
Table 26-83 CAN Timer Triggers (cont'd)

SEL4 / 5	Trigger 4	Trigger 5
1011 _B	Reserved	Reserved
1100 _B	Reserved	Reserved
1101 _B	Reserved	Reserved
1110 _B	Reserved	Reserved
1111 _B	Reserved	Reserved

Table 26-84 CAN Timer Triggers

SEL6 / 7	Trigger 6	Trigger 7
0000 _B	TOM1 Channel 4 output	TOM1 Channel 4 output
0001 _B	TOM1 Channel 5 output	TOM1 Channel 5 output
0010 _B	TOM1 Channel 11 output	TOM1 Channel 11 output
0011 _B	TOM1 Channel 12 output	TOM1 Channel 12 output
0100 _B	ATOM2 Channel 0 output	ATOM2 Channel 0 output
0101 _B	ATOM2 Channel 1 output	ATOM2 Channel 1 output
0110 _B	ATOM2 Channel 2 output	ATOM2 Channel 2 output
0111 _B	ATOM2 Channel 3 output	ATOM2 Channel 3 output
1000 _B	Reserved	Reserved
1001 _B	Reserved	Reserved
1010 _B	Reserved	Reserved
1011 _B	Reserved	Reserved
1100 _B	Reserved	Reserved
1101 _B	Reserved	Reserved
1110 _B	Reserved	Reserved
1111 _B	Reserved	Reserved

26.22.8 CCU6x Connections

This section summarize the connections to the CCU6x.

Generic Timer Module (GTM)**Table 26-85 GTM to CCU60 Connections**

GTM Output	CCU60 Input
TOM0_8	T12HRD
TOM1_8	T13HRD

Table 26-86 GTM to CCU61 Connections

GTM Output	CCU61 Input
ATOM0_1	T12HRD
TOM0_9	T13HRD

26.22.9 PSI5 Connections

This register defines the GTM to PSI5 connections of TC26x B-Step.

The PSI5 trigger inputs are generated by 16 to 1 multiplexer that are implemented twice per PSI5 module / channel plus 3 combined for all PSI5 channels together. The output signal to the PSI5 is called *psi5x_trig0*, The inputs of the multiplexer are called *psi5_x_muxin[7:0]*, x = number of PSI5 triggers, TC26x B-Step has 6 triggers.

Mapping of PSI5 Triggers

Table 26-87 PSI5 Timer Triggers

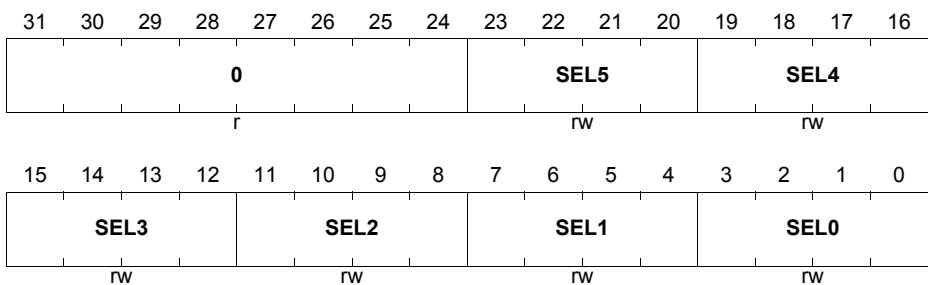
SELx	Trigger 0 to 5
0000 _B	No trigger is generated
0001 _B	TOM1 Channel 6 output
0010 _B	TOM1 Channel 7 output
0011 _B	TOM1 Channel 13 output
0100 _B	TOM1 Channel 14 output
0101 _B	ATOM2 Channel 4 output
0110 _B	ATOM2 Channel 5 output
0111 _B	ATOM2 Channel 6 output
1000 _B	ATOM2 Channel 7 output
1001 _B	TOM0 Channel 6 output
1010 _B	TOM0 Channel 7 output
1011 _B	TOM0 Channel 13 output
1100 _B	TOM0 Channel 14 output
1101 _B	Reserved
1110 _B	Reserved
1111 _B	Reserved

Table 26-88 PSI5-S Timer Triggers

SELx	Trigger 0 / 4	Trigger 1 / 5	Trigger 2 / 6
0000 _B	No trigger is generated	No trigger is generated	No trigger is generated
0001 _B	TOM0_6	TOM1_6	TOM1_6
0010 _B	TOM0_7	TOM1_7	TOM1_7

Generic Timer Module (GTM)
Table 26-88 PSI5-S Timer Triggers (cont'd)

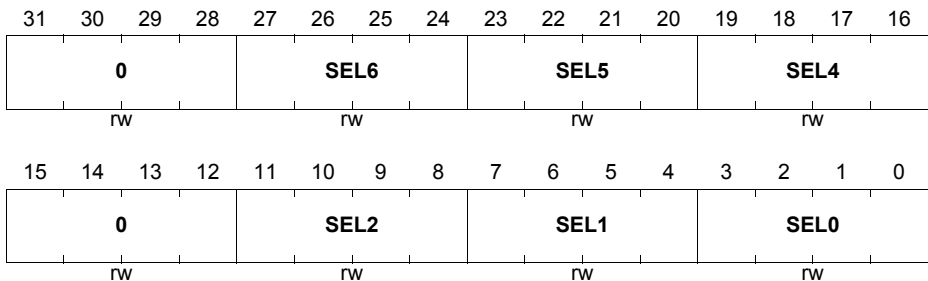
SELx	Trigger 0 / 4	Trigger 1 / 5	Trigger 2 / 6
0011 _B	TOM0_13	TOM1_13	TOM1_13
0100 _B	TOM0_14	TOM1_14	TOM1_14
0101 _B	ATOM0_4	ATOM1_4	ATOM3_4
0110 _B	ATOM0_5	ATOM1_5	ATOM3_5
0111 _B	ATOM0_6	ATOM1_6	ATOM3_6
1000 _B	ATOM0_7	ATOM1_7	ATOM3_7
1001 _B	Reserved	Reserved	Reserved
1010 _B	Reserved	Reserved	Reserved
1011 _B	Reserved	Reserved	Reserved
1100 _B	Reserved	Reserved	Reserved
1101 _B	Reserved	Reserved	Reserved
1110 _B	Reserved	Reserved	Reserved
1111 _B	Reserved	Reserved	Reserved

PSI5OUTSEL0
PSI5 Output Select 0 Register
(9FDA4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SELx (x = 0-5)	[x*4+3: x*4]	rw	Output Selection for PSI5x GTM connection This bit field defines which TOM / ATOM channel output is used as PSI5x trigger 0. Decoding is defined by Table 26-87 .

Generic Timer Module (GTM)

Field	Bits	Type	Description
0	[31:24]	r	Reserved Read as 0; should be written with 0.

PSI5SOUTSEL
PSI5-S Output Select Register (9FDA8_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
SELx (x = 0-2)	[x*4+3: x*4]	rw	Output Selection for PSI5x GTM connection This bit field defines which TOM / ATOM channel output is used as PSI5x trigger 0. Decoding is defined by Table 26-88 .
SELx (x = 4-6)	[x*4+3: x*4]	rw	Output Selection for PSI5x GTM connection This bit field defines which TOM / ATOM channel output is used as PSI5x trigger 0. Decoding is defined by Table 26-88 .
0	[15:12], [31:28]	rw	Reserved Should be written with 0.

26.22.10 GTM Data Exchange Registers

The registers defines additional data exchange and control options for GTM towards the on-chip system of TC26x B-Step.

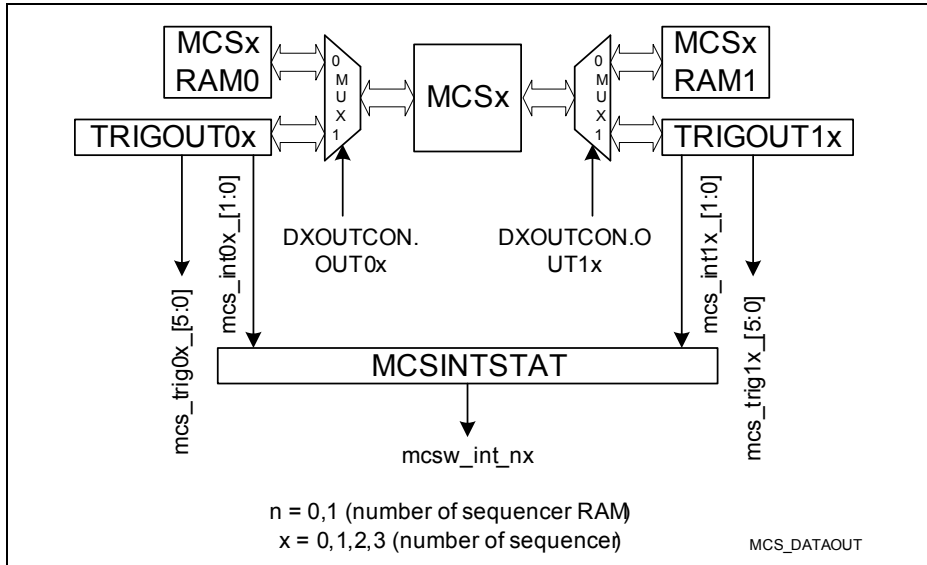
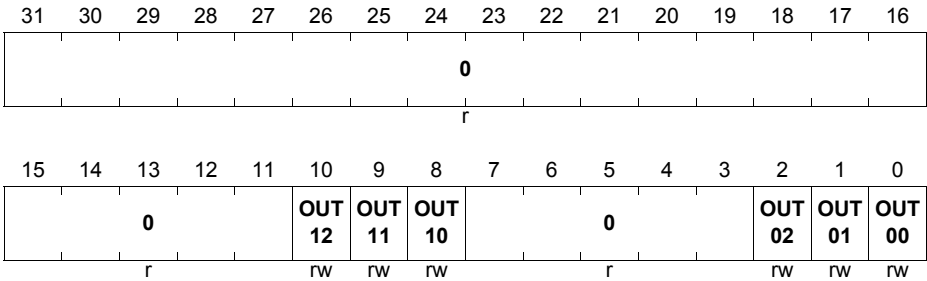


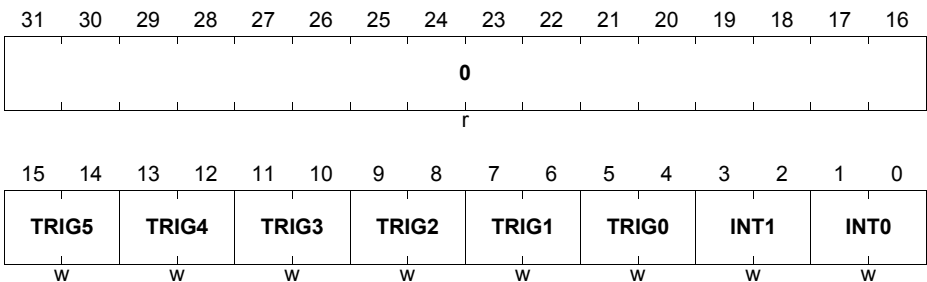
Figure 26-87 MCS Trigger Output Path

The TRIGOUT registers enable the GTM internal sequencers to generate dedicated interrupts or triggers from the sequence flow by writing to this register. The TRIGOUT registers are mirrored to address 0x0000 for each of the two sequencer RAMs. If a write for address 0x0000 updates the RAM or the TRIGOUT register is controlled by the bus accessible register DXOUTCON in central for all sequencers and there RAMs. This need to be configured only once when setting up the GTM with the reminder of the system.

Register MCSINTSTAT contains interrupt status flags and register MCSINTCLR provides the option to clear these status flags from the system bus. Both registers are optional register.

Generic Timer Module (GTM)
DXOUTCON
Data Exchange Output Control Register (9FE00_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
OUT0x (x = 0-2)	x	rw	Output 0x Control This bit defines if register TRIGOUT0x is accessible from the MCS instead the RAM0 or not. 0 _B RAM0 memory is accessed 1 _B Register TRIGOUT0x is accessed
OUT1x (x = 0-2)	x+8	rw	Output 1x Control This bit defines if register TRIGOUT1x is accessible from the MCS instead the RAM1 or not. 0 _B RAM1 memory is accessed 1 _B Register TRIGOUT1x is accessed
0	[7:3],[3 1:11]	r	Reserved Read as 0; should be written with 0.

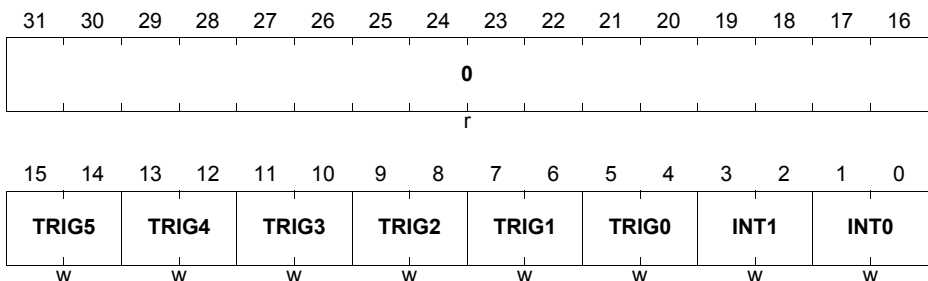
TRIGOUT0n (n = 0-2)
Trigger Output 0 Register n (9FE04_H+n*4_H)
Reset Value: 0000 0000_H


Generic Timer Module (GTM)

Field	Bits	Type	Description
INT0	[1:0]	w	Interrupt Trigger Request 0 This bit field if an interrupt request 0 is generated or not. 00 _B No modification of current state 01 _B Pending interrupt request 0 is cleared 10 _B Interrupt request 0 is set 11 _B No modification of current state
INT1	[3:2]	w	Interrupt Trigger Request 1 This bit field if an interrupt request 1 is generated or not. 00 _B No modification of current state 01 _B Pending interrupt request 1 is cleared 10 _B Interrupt request 1 is set 11 _B No modification of current state
TRIG0	[5:4]	w	Trigger 0 This bit field if an trigger 0 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 0 is cleared 10 _B Trigger 0 is set 11 _B No modification of current state
TRIG1	[7:6]	w	Trigger 1 This bit field if an trigger 1 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 1 is cleared 10 _B Trigger 1 is set 11 _B No modification of current state
TRIG2	[9:8]	w	Trigger 2 This bit field if an trigger 2 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 2 is cleared 10 _B Trigger 2 is set 11 _B Current trigger 2 level is toggled

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRIG3	[11:10]	w	Trigger 3 This bit field if an trigger 3 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 3 is cleared 10 _B Trigger 3 is set 11 _B Current trigger 3 level is toggled
TRIG4	[13:12]	w	Trigger 4 This bit field if an trigger 4 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 4 is cleared 10 _B Trigger 4 is set 11 _B Current trigger 4 level is toggled
TRIG5	[15:14]	w	Trigger 5 This bit field if an trigger 5 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 5 is cleared 10 _B Trigger 5 is set 11 _B Current trigger 5 level is toggled
0	[31:16]	r	Reserved Read as 0; should be written with 0.

TRIGOUT1n (n = 0-2)
Trigger Output 0 Register n (9FE44_H+n*4_H) **Reset Value: 0000 0000_H**


Generic Timer Module (GTM)

Field	Bits	Type	Description
INT0	[1:0]	w	Interrupt Trigger Request 0 This bit field if an interrupt request 0 is generated or not. 00 _B No modification of current state 01 _B Pending interrupt request 0 is cleared 10 _B Interrupt request 0 is set 11 _B No modification of current state
INT1	[3:2]	w	Interrupt Trigger Request 1 This bit field if an interrupt request 1 is generated or not. 00 _B No modification of current state 01 _B Pending interrupt request 1 is cleared 10 _B Interrupt request 1 is set 11 _B No modification of current state
TRIG0	[5:4]	w	Trigger 0 This bit field if an trigger 0 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 0 is cleared 10 _B Trigger 0 is set 11 _B No modification of current state
TRIG1	[7:6]	w	Trigger 1 This bit field if an trigger 1 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 1 is cleared 10 _B Trigger 1 is set 11 _B No modification of current state
TRIG2	[9:8]	w	Trigger 2 This bit field if an trigger 2 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 2 is cleared 10 _B Trigger 2 is set 11 _B Current trigger 2 level is toggled

Generic Timer Module (GTM)

Field	Bits	Type	Description
TRIG3	[11:10]	w	Trigger 3 This bit field if an trigger 3 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 3 is cleared 10 _B Trigger 3 is set 11 _B Current trigger 3 level is toggled
TRIG4	[13:12]	w	Trigger 4 This bit field if an trigger 4 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 4 is cleared 10 _B Trigger 4 is set 11 _B Current trigger 4 level is toggled
TRIG5	[15:14]	w	Trigger 5 This bit field if an trigger 5 request is generated or not. 00 _B No modification of current state 01 _B Pending trigger 5 is cleared 10 _B Trigger 5 is set 11 _B Current trigger 5 level is toggled
0	[31:16]	r	Reserved Read as 0; should be written with 0.

MCS to ADC Triggers

The triggers x0 can be used either as GPIO outputs or a gated trigger inputs for some ADCs.

Table 26-89 MCS0 Trigger Assignments

MCS Trigger Output	Port	ADC Input	Control
RAM0			
mcs_trig00_[0]	n.A.	ADC0 BFDAT0	TRIGOUT00.TRIG0
mcs_trig00_[1]	n.A.	ADC0 BFSEL0	TRIGOUT00.TRIG1
mcs_trig00_[2]	n.A.	ADC0 BFDAT1	TRIGOUT00.TRIG2
mcs_trig00_[3]	n.A.	ADC0 BFSEL1	TRIGOUT00.TRIG3
mcs_trig00_[4]	n.A.	Reserved	TRIGOUT00.TRIG4
mcs_trig00_[5]	n.A.	Reserved	TRIGOUT00.TRIG5

Generic Timer Module (GTM)
Table 26-89 MCS0 Trigger Assignments (cont'd)

MCS Trigger Output	Port	ADC Input	Control
RAM1			
mcs_trig10_[0]	n.A.	ADC1 BFDAT0	TRIGOUT10.TRIG0
mcs_trig10_[1]	n.A.	ADC1 BFSEL0	TRIGOUT10.TRIG1
mcs_trig10_[2]	n.A.	ADC1 BFDAT1	TRIGOUT10.TRIG2
mcs_trig10_[3]	n.A.	ADC1 BFSEL1	TRIGOUT10.TRIG3
mcs_trig10_[4]	n.A.	Reserved	TRIGOUT10.TRIG4
mcs_trig10_[5]	n.A.	Reserved	TRIGOUT10.TRIG5

Table 26-90 MCS1 Trigger Assignments

MCS Trigger Output	Port	ADC Input	Control
RAM0			
mcs_trig01_[0]	n.A.	ADC2 BFDAT0	TRIGOUT01.TRIG0
mcs_trig01_[1]	n.A.	ADC2 BFSEL0	TRIGOUT01.TRIG1
mcs_trig01_[2]	n.A.	ADC2 BFDAT1	TRIGOUT01.TRIG2
mcs_trig01_[3]	n.A.	ADC2 BFSEL1	TRIGOUT01.TRIG3
mcs_trig01_[4]	n.A.	Reserved	TRIGOUT01.TRIG4
mcs_trig01_[5]	n.A.	Reserved	TRIGOUT01.TRIG5
RAM1			
mcs_trig11_[0]	n.A.	ADC3 BFDAT0	TRIGOUT11.TRIG0
mcs_trig11_[1]	n.A.	ADC3 BFSEL0	TRIGOUT11.TRIG1
mcs_trig11_[2]	n.A.	ADC3 BFDAT1	TRIGOUT11.TRIG2
mcs_trig11_[3]	n.A.	ADC3 BFSEL1	TRIGOUT11.TRIG3
mcs_trig11_[4]	n.A.	Reserved	TRIGOUT11.TRIG4
mcs_trig11_[5]	n.A.	Reserved	TRIGOUT11.TRIG5

Table 26-91 MCS2 Trigger Assignments

MCS Trigger Output	Port	ADC Input	Control
RAM0			
mcs_trig02_[0]	n.A.	Reserved	TRIGOUT02.TRIG0
mcs_trig02_[1]	n.A.	Reserved	TRIGOUT02.TRIG1

Generic Timer Module (GTM)
Table 26-91 MCS2 Trigger Assignments (cont'd)

MCS Trigger Output	Port	ADC Input	Control
mcs_trig02_[2]	n.A.	Reserved	TRIGOUT02.TRIG2
mcs_trig02_[3]	n.A.	Reserved	TRIGOUT02.TRIG3
mcs_trig02_[4]	n.A.	Reserved	TRIGOUT02.TRIG4
mcs_trig02_[5]	n.A.	Reserved	TRIGOUT02.TRIG5
RAM1			
mcs_trig12_[0]	n.A.	Reserved	TRIGOUT12.TRIG0
mcs_trig12_[1]	n.A.	Reserved	TRIGOUT12.TRIG1
mcs_trig12_[2]	n.A.	Reserved	TRIGOUT12.TRIG2
mcs_trig12_[3]	n.A.	Reserved	TRIGOUT12.TRIG3
mcs_trig12_[4]	n.A.	Reserved	TRIGOUT12.TRIG4
mcs_trig12_[5]	n.A.	Reserved	TRIGOUT12.TRIG5

Table 26-92 MCS0 Interrupt Trigger Assignments

MCS Trigger Output	Interrupt Router	Control
RAM0		
mcs_int00_[0]	SRC_GTMMCSW00	TRIGOUT00.INT0
mcs_int00_[1]	Reserved	TRIGOUT00.INT1
RAM1		
mcs_int10_[0]	SRC_GTMMCSW01	TRIGOUT10.INT0
mcs_int10_[1]	Reserved	TRIGOUT10.INT1

Table 26-93 MCS1 Interrupt Trigger Assignments

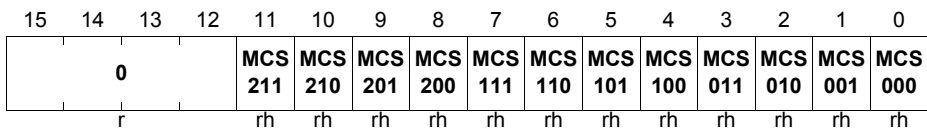
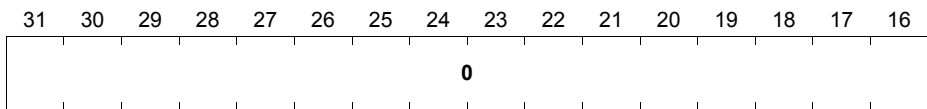
MCS Trigger Output	Interrupt Router	Control
RAM0		
mcs_int01_[0]	SRC_GTMMCSW10	TRIGOUT01.INT0
mcs_int01_[1]	Reserved	TRIGOUT01.INT1
RAM1		
mcs_int11_[0]	SRC_GTMMCSW11	TRIGOUT11.INT0
mcs_int11_[1]	Reserved	TRIGOUT11.INT1

Table 26-94 MCS2 Interrupt Trigger Assignments

MCS Trigger Output	Interrupt Router	Control
RAM0		
mcs_int02_[0]	SRC_GTMMCSW20	TRIGOUT02.INT0
mcs_int02_[1]	Reserved	TRIGOUT02.INT1
RAM1		
mcs_int12_[0]	SRC_GTMMCSW21	TRIGOUT12.INT0
mcs_int12_[1]	Reserved	TRIGOUT12.INT1

MCSINTSTAT

MCS Interrupt Status Register (9FE70_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
MCSn00 (n = 0-2)	n*4	rh	MCSn RAM0 Interrupt 0 Status 0 _B No interrupt was requested 1 _B A interrupt was requested The requested interrupt is MCSW0n This bit is cleared when bit MCSINTCLR.MCSn00 is set.
MCSn01 (n = 0-2)	n*4+1	rh	MCSn RAM0 Interrupt 1 Status 0 _B No interrupt was requested 1 _B A interrupt was requested The requested interrupt is MCSW0n This bit is cleared when bit MCSINTCLR.MCSn01 is set.

Generic Timer Module (GTM)

Field	Bits	Type	Description
MCSn10 (n = 0-2)	n*4+2	rh	MCSn RAM1 Interrupt 0 Status 0 _B No interrupt was requested 1 _B A interrupt was requested The requested interrupt is MCSW1n This bit is cleared when bit MCSINTCLR.MCSn10 is set.
MCSn11 (n = 0-2)	n*4+3	rh	MCSn RAM1 Interrupt 1 Status 0 _B No interrupt was requested 1 _B A interrupt was requested The requested interrupt is MCSW1n This bit is cleared when bit MCSINTCLR.MCSn11 is set.
0	[31:12]	r	Reserved Read as 0; should be written with 0.

MCSINTCLR
MCS Interrupt Clear Register
(9FE74_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			MCS 211	MCS 210	MCS 201	MCS 200	MCS 111	MCS 110	MCS 101	MCS 100	MCS 011	MCS 010	MCS 001	MCS 000	
r			w	w	w	w	w	w	w	w	w	w	w	w	

Field	Bits	Type	Description
MCSn00 (n = 0-2)	n*4	w	MCSn RAM0 Interrupt 0 Status Clear 0 _B No action 1 _B Bit MCSINTSTAT.MCSn00 is cleared This bit read always as zero.
MCSn01 (n = 0-2)	n*4+1	w	MCSn RAM0 Interrupt 1 Status Clear 0 _B No action 1 _B Bit MCSINTSTAT.MCSn01 is cleared This bit read always as zero.

Generic Timer Module (GTM)

Field	Bits	Type	Description
MCSn10 (n = 0-2)	n*4+2	w	MCSn RAM1 Interrupt 0 Status Clear 0 _B No action 1 _B Bit MCSINTSTAT.MCSn10 is cleared This bit read always as zero.
MCSn11 (n = 0-2)	n*4+3	w	MCSn RAM1 Interrupt 1 Status Clear 0 _B No action 1 _B Bit MCSINTSTAT.MCSn11 is cleared This bit read always as zero.
0	[31:12]	r	Reserved Read as 0; should be written with 0.

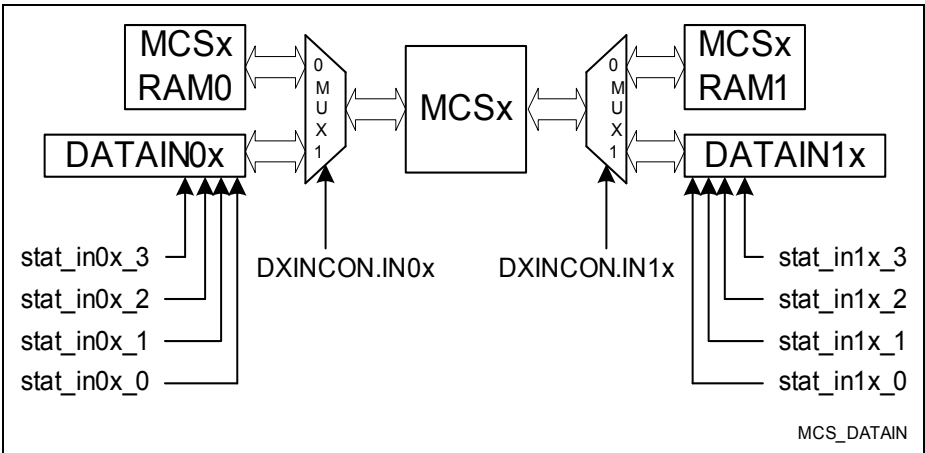


Figure 26-88 MCS Data Input path

The DATAIN registers enable the GTM internal sequencers to access data provided by other masters in the system directly in the sequencer data flow without ARU interaction. In addition the status of dedicated system triggers could also be observed here. The DATAIN registers are mirrored to address 0x0001 for each of the two sequencer RAMs. If a read for address 0x0001 fetch data from the RAM or the DATAIN register is controlled by the bus accessible register DXINCON in central for all sequencers and there RAMs. In addition register DATAIN contains the control if the 4 LSB of the DATAIN register contains data or the status of the four system signals. This need to be configured only once when setting up the GTM with the reminder of the system.

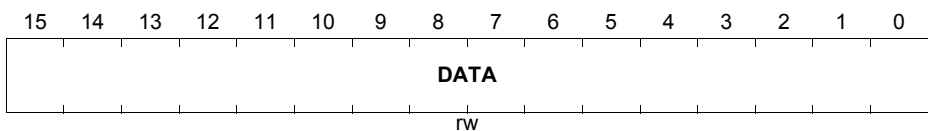
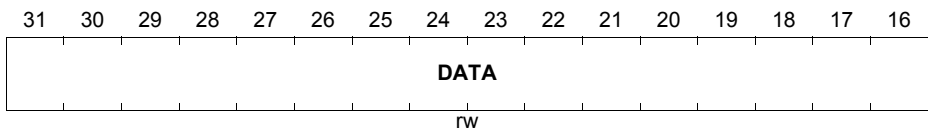
Generic Timer Module (GTM)
DXINCON
Data Exchange Input Control Register (9FE90_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		0			DSS 12	DSS 11	DSS 00			0			DSS 02	DSS 01	DSS 00
		r			rw	rw	rw			r			rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		0			IN12	IN11	IN10			0			IN02	IN01	IN00
		r			rw	rw	rw			r			rw	rw	rw

Field	Bits	Type	Description
IN0x (x = 0-2)	x	rw	Input 0x Control This bit defines if register DATAIN0x is read from the MCS instead the RAM0 or not. 0 _B RAM0 memory is read 1 _B Register DATAIN0x is read
IN1x (x = 0-2)	x+8	rw	Input 1x Control This bit defines if register DATAIN1x is read from the MCS instead the RAM1 or not. 0 _B RAM1 memory is read 1 _B Register DATAIN1x is read
DSS0x (x = 0-2)	x+16	rw	Data Source Select 0x Control This bit defines if the 4 LSB of the read operation directed to register DATAIN0x deliver the register content or the state of four inputs. 0 _B Register DATAIN0x[3:0] is read 1 _B Status of inputs [3:0] is read The four Inputs are labeled datain0x0, datain0x1, datain0x2, and datain0x3.

Generic Timer Module (GTM)

Field	Bits	Type	Description
DSS1x (x = 0-2)	x+24	rw	Data Source Select 1x Control This bit defines if the 4 LSB of the read operation directed to register DATAIN1x deliver the register content or the state of four inputs. 0 _B Register DATAIN1x[3:0] is read 1 _B Status of inputs [3:0] is read The four Inputs are labeled datain1x0, datain1x1, datain1x2, and datain1x3.
0	[7:3], [15:11], [23:19], [31:27]	r	Reserved Read as 0; should be written with 0.

DATAIN0y (y = 0-2)
Data Input 0 y Register
(9FE94_H+y*4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
DATA	[31:0]	rw	Data This bit field hold the data for the RAM0 replacement.

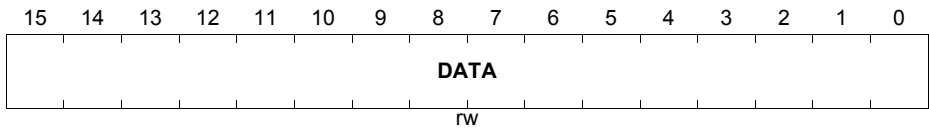
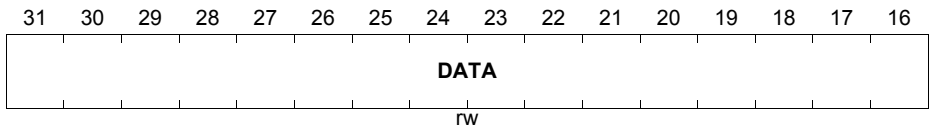
Generic Timer Module (GTM)

DATAIN1y (y = 0-2)

Data Input y Register

($9FED4_H + y * 4_H$)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
DATA	[31:0]	rw	Data This bit field hold the data for the RAM1 replacement.

Input connections

Table 26-95 MCS Data Input Signal connections

MCS Status Input	Input
stat_in00_[0]	ADC0 BFL0
stat_in01_[0]	ADC1 BFL0
stat_in02_[0]	ADC2 BFL0
stat_in10_[0]	ADC3 BFL0
stat_in00_[1]	ADC0 BFL1
stat_in01_[1]	ADC1 BFL1
stat_in02_[1]	ADC2 BFL1
stat_in10_[1]	ADC3 BFL1
all other	Reserved

26.22.11 SCU Connections

This section summarize the connections to the SCU.

Table 26-96 GTM to SCU Connections

GTM Output	SCU Input
TOM0_12	Input41
TOM1_12	Input51

26.22.12 GTM Debug Interface

OTGB0/1 are 16-bit trigger busses, OTGB2 is 32 bit wide. Further information is available in the OTGM (OCDS Trigger Mux) section of the device specification.

Note: If triggering or tracing is to be used the GTM clock must not be faster than the SPB clock.

26.22.12.1OCDS Trigger Bus (OTGB) Interface

GTM OTGB Features

- IO signals
 - TIM input signals (after filter)
 - TOM output signals
 - ATOM output signals
 - SPE NIPD and DIR signals
 - Groups of 8 signals (e.g. all inputs of a specific TIM, etc.)
 - Two arbitrary groups on one 16 bit OTGB0/1
 - OTGB0 and/or OTGB1 can be used for 2-4 groups in parallel (e.g. both with one TIM 8, one ATOM 8 and one TOM 16)
- MCS
 - Channel number, PC and data access type (read or write)
 - Data value on OTGB2
 - Only one 16 bit OTGB0/1 needed if no data trace. Other free for IO signals.
- ARU
 - ARU_DBG_DATA0/1_H/L on 32 bit OTGB2 (time multiplexed)
- DPLL
 - TASI and SASI signals
 - Accesses to RAM1A/1BC/2 (only one at a time)
- TBUs
 - Trace one selected TBU
 - Trigger signals for individual TBU comparators

The GTM module has five 16 bit and eight 32 bit Trigger Sets ([Table 26-97](#)) which are selected with the **OTSS** register.

Table 26-97 GTM Trigger Sets

Trigger Set	Details
TS16_IOS Trigger Set IO and Other Signals	Table 26-99
TS16_MCA MCS Channel, Address and PC	Table 26-100
TS32_MCD MCS Data	Table 26-101
TS32_ARU Trigger Set ARU	Table 26-102
TS16_DRA1A/1BC/2 DPLL RAM Access Address (3 Sets)	Table 26-103
TS32_DRD1A/1BC/2 DPLL RAM Access Data (3 Sets)	Table 26-104
TS32_TTB0/1/2 TBU Time Stamps (3 Sets)	Table 26-105

Table 26-98 shows all sensible Trigger Set mapping options. Simple permutations of OTGB0/1 are omitted. If OTGB0 and/or OTGB1 is not used for GTM, Trigger Sets of other sources can be added in the OTGM module. The Trigger Sets which will initiate the writing of a 32 or 64 bit data word to the MCDS trace are marked with bold letters. This is by design always the case for OTGB2 Trigger Sets. Independent OTGB0/1 Trigger Sets need an independent own sensitivity to their signal changes, configured in the OTGM module.

Table 26-98 Trigger Set Mapping Options

Width	OTGB0	OTGB1	OTGB2
32 Bit	TS16_IOS or MCA		
	TS16_IOS or MCA	TS16_IOS	
64 Bit			TS32_ARU or TTB0/1/2
	TS16_IOS or MCA		TS32_MCD
	TS16_IOS or MCA	TS16_IOS	
	TS16_MCA		
	TS16_MCA	TS16_IOS	TS32_DRD1A/1BC/2
	TS16_DRA1A/1BC/2		
	TS16_DRA1A/1BC/2	TS16_IOS	

Generic Timer Module (GTM)

The GTM trigger signals relate to the GTM internal clock, which can be slower or equal to the OTGB/OTGM clock.

IO and Other Signals Trigger Sets

IO Trigger Sets consist of the most important TIM, TOM and ATOM signals in groups of 8. In addition there is a group of SPE signals and one group for miscellaneous signals. The multiplexer allows to map arbitrary and different signal groups to the high and the low byte of a 16 bit Trigger Set. In addition it is possible to use one or two 16 bit Trigger Sets with this flexibility. All this is controlled with **OTSC0**.

Table 26-99 TS16_IOS Trigger Set IO and Other Signals

Bits	Name	Description	
[7:0]	SG0	A group of 8 signals from a selected TIM, TOM, ATOM or SPE	
		TIM	All 8 input signals after filter F_OUT
		TOML	Lower 8 output signals TOM_OUT
		TOMH	Higher 8 output signals TOM_OUT
		ATOM	All 8 output signals ATOM_CHx_OUT
		SPE	Bits 0,1: SPE0_NIPD, SPE_DIR0 Bits 2,3: SPE1_NIPD, SPE_DIR1 Bits 4,5: Reserved Bits 6,7: Reserved
[15:8]	SG1	MISC	Bit 0: DPLL TASI Bit 1: DPLL SASI Bit 4: TBU0 trigger (OTBU0T) Bit 5: TBU1 trigger (OTBU1T) Bit 6: TBU2 trigger (OTBU2T) Others: reserved
		Independent selection with same options as for Bits [7:0]	

MCS Trigger Sets

There are two MCS Trigger Sets: TS16_MCA (**Table 26-100**) and TS32_MCD (**Table 26-101**). TS16_MCA allows to observe the PC of all or a selected MCS Channel. If only one Channel is selected, the TS16_MCA value will stay stable until the MCS executes this Channel again. In case of data access by CPU or by a Channel, it represents the data address. With TS32_MCD the associated data value and even the fetched instruction opcode can be traced. Usually the latter is avoided to reduce the trace bandwidth. All this is controlled with **OTSC1**.

Table 26-100 TS16_MCA MCS Channel, Address and PC

Bits	Name	Description
[11:0]	ADDR	PC of MCS Channel or address of data access (32 bit word address)
[15:12]	AQ	Address Qualifier with Channel information 0_H ADDR is PC of MCS Channel 0 H ADDR is PC of MCS Channel 7_H ADDR is PC of MCS Channel 7 C_H reserved D_H reserved E_H ADDR is address of Channel data read access F_H ADDR is address of Channel data write access Others: reserved

Table 26-101 TS32_MCD MCS Data

Bits	Description
[31:0]	Read or written data value of current channel instruction

ARU Trigger Sets

There are four 29 bit ARU data words with debug information (ARU_DBG_DATA0_L/H and ARU_DBG_DATA1_L/H). These words may get valid in the same GTM kernel clock cycle but they will never change in consecutive cycles. This property is used to transmit them with time multiplexing over the OTGB2 bus. ARU_NEW_DATA0/1_IRQ controls the OTGB2 valid signal.

The implementation of TS32_ARU has to capture the ARU_NEW_DATA0/1_IRQ signals and clear these capture bits, when the associated ARU data has been output on OTGB2. The ARU data itself doesn't need to be captured because it will be valid long enough. If both ARU_NEW_DATA0/1_IRQ signals become active in the same cycle, the ARU data is output in four consecutive cycles with the sequence ARU_DBG_DATA0_L, DATA0_H, DATA1_L, DATA1_H.

Table 26-102 TS32_ARU Trigger Set ARU

Bits	Name	Description
[28:0]	DATA	ARU_DBG_DATA

Generic Timer Module (GTM)
Table 26-102 TS32_ARU Trigger Set ARU (cont'd)

Bits	Name	Description
29		Reserved
[31:30]	DQ	Qualifier for DATA 0_D Is ARU_DBG_DATA0_L 1_D Is ARU_DBG_DATA0_H 2_D Is ARU_DBG_DATA1_L 3_D Is ARU_DBG_DATA1_H

DPLL Trigger Sets

The TASI and SASI signals are part of the MISC signal group of TS16_IOS (Table 26-99).

The following tables Table 26-103 and Table 26-104 define six additional DPLL Trigger Sets for observing the DPLL RA

M accesses. Only one of the three DPLL (1A, 1B+C, 2) RAMs can be observed at a time. The Trigger Set selection is done with OTSS.

Table 26-103 TS16_DRA1A/1BC/2 DPLL RAM Access Address

Bits	Name	Description
[13:0]	ADDR	Address of data access. Range starts with address 0 for the selected RAM (1A, 1B+C or 2). Effective width depends on selected RAM
[15:14]	AQ	Address Qualifier 0_H DPLL data read access 1_H DPLL data write access 2_H CPU read access 3_H CPU write access

Table 26-104 TS32_DRD1A/1BC/2 DPLL RAM Access Data

Bits	Name	Description
[23:0]	DATA	Read or written data
[31:24]		Reserved

TBU Trigger Sets

Table 26-105 defines the Trigger Sets for the observation of the different time bases. The Trigger Set selection (one at a time) is done with OTSS.

Table 26-105 TS32_TTB0/1/2 TBU Time Stamps

Bits	Name	Description
[26:0]	TS	Current TBU_TS0/1/2 time stamp. Effective width depends on selected time base.
[31:27]		Reserved

26.22.12.2GTM Debug Registers

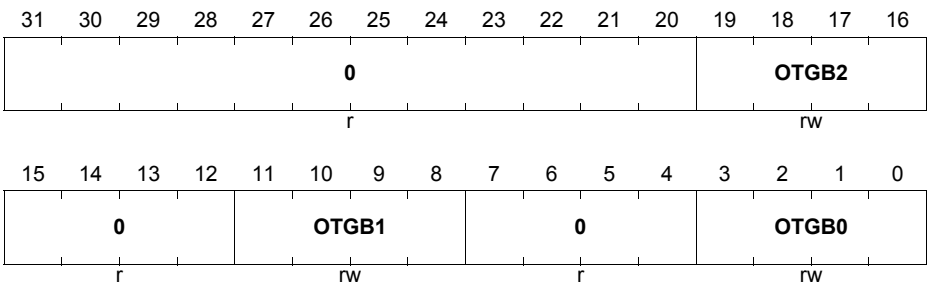
All debug control registers are cleared by Debug Reset.

OCDS Trigger Bus (OTGB)

Access are only supported for byte, halfword and word data and requires Supervisor Mode.

OTSS

OCDS Trigger Set Select Register (9FDD0_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
OTGB0	[3:0]	rw	Trigger Set for OTGB0 0 _D No Trigger Set selected 1 _D Trigger Set TS16_IOS (Table 26-99) 2 _D Trigger Set TS16_MCA (Table 26-100) 3 _D Trigger Set TS16_DRA1A (Table 26-103) 4 _D Trigger Set TS16_DRA1BC (Table 26-103) 5 _D Trigger Set TS16_DRA2 (Table 26-103) others , reserved
OTGB1	[11:8]	rw	Trigger Set for OTGB1 0 _D No Trigger Set selected 1 _D Trigger Set TS16_IOS (Table 26-99) 2 _D Trigger Set TS16_MCA (Table 26-100) 3 _D Trigger Set TS16_DRA1A (Table 26-103) 4 _D Trigger Set TS16_DRA1BC (Table 26-103) 5 _D Trigger Set TS16_DRA2 (Table 26-103) others , reserved

Generic Timer Module (GTM)

Field	Bits	Type	Description
OTGB2	[19:16]	rw	Trigger Set for OTGB2 0 _H No Trigger Set selected 1 _H Trigger Set TS32_ARU (Table 26-102) 2 _H Trigger Set TS32_MCD (Table 26-101) 3 _H Trigger Set TS32_DRD1A (Table 26-104) 4 _H Trigger Set TS32_DRD1BC (Table 26-104) 5 _H Trigger Set TS32_DRD2 (Table 26-104) 8 _H Trigger Set TS32_TTB0 (Table 26-105) 9 _H Trigger Set TS32_TTB1 (Table 26-105) A _H Trigger Set TS32_TTB2 (Table 26-105) others, reserved
0	[7:4], [15:12], [31:20]	r	Reserved Read as 0; must be written with 0.

OTSC0
OCDS Trigger Set Control 0 Register(9FDD4_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
B1HMI				0	B1HMT				B1LMI				0	B1LMT			
rw				r	rw				rw				r	rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
B0HMI				0	B0HMT				B0LMI				0	B0LMT			
rw				r	rw				rw				r	rw			

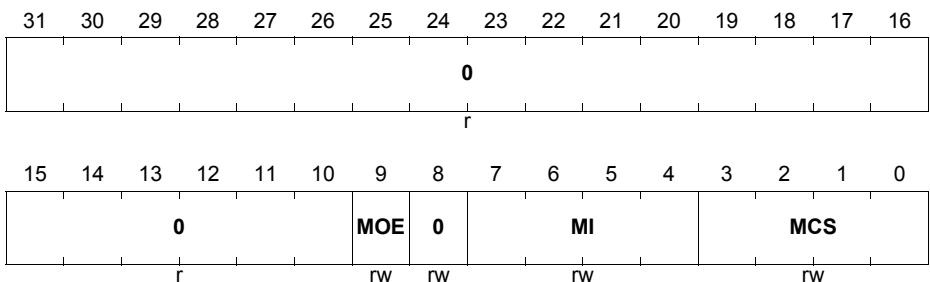
Field	Bits	Type	Description
B0LMT	[2:0]	rw	OTGB0 TS16_IOS Low Byte Module Type 0 _D No Module selected 1 _D TIM (F_OUT[7:0]) 2 _D TOML (TOM_OUT[7:0]) 3 _D TOMH (TOM_OUT[15:8]) 4 _D ATOM (ATOM_OUT[7:0]) 5 _D SPE signals (Table 26-99) 6 _D MISC signals (Table 26-99) others, reserved

Generic Timer Module (GTM)

Field	Bits	Type	Description
B0LMI	[7:4]	rw	OTGB0 TS16_IOS Low Byte Module Instance Index of the module instance. Index starts with 0, the max. value depends on GTM configuration and module type. For SPE and MISC signals this index is ignored.
B0HMT	[10:8]	rw	OTGB0 TS16_IOS High Byte Module Type 0 _D No Module selected 1 _D TIM (F_OUT[7:0]) 2 _D TOML (TOM_OUT[7:0]) 3 _D TOMH (TOM_OUT[15:8]) 4 _D ATOM (ATOM_OUT[7:0]) 5 _D SPE signals (Table 26-99) 6 _D MISC signals (Table 26-99) others , reserved
B0HMI	[15:12]	rw	OTGB0 TS16_IOS High Byte Module Instance Index of the module instance. Index starts with 0, the max. value depends on GTM configuration and module type. For SPE and MISC signals this index is ignored.
B1LMT	[18:16]	rw	OTGB1 TS16_IOS Low Byte Module Type 0 _D No Module selected 1 _D TIM (F_OUT[7:0]) 2 _D TOML (TOM_OUT[7:0]) 3 _D TOMH (TOM_OUT[15:8]) 4 _D ATOM (ATOM_OUT[7:0]) 5 _D SPE signals (Table 26-99) 6 _D MISC signals (Table 26-99) others , reserved
B1LMI	[23:20]	rw	OTGB1 TS16_IOS Low Byte Module Instance Index of the module instance. Index starts with 0, the max. value depends on GTM configuration and module type. For SPE and MISC signals this index is ignored.
B1HMT	[26:24]	rw	OTGB1 TS16_IOS High Byte Module Type 0 _D No Module selected 1 _D TIM (F_OUT[7:0]) 2 _D TOML (TOM_OUT[7:0]) 3 _D TOMH (TOM_OUT[15:8]) 4 _D ATOM (ATOM_OUT[7:0]) 5 _D SPE signals (Table 26-99) 6 _D MISC signals (Table 26-99) others , reserved

Generic Timer Module (GTM)

Field	Bits	Type	Description
B1HMI	[31:28]	rw	OTGB1 TS16_IOS High Byte Module Instance Index of the module instance. Index starts with 0, the max. value depends on GTM configuration and module type. For SPE and MISC signals this index is ignored.
0	3, 11, 19, 27	r	Reserved Read as 0; must be written with 0.

OTSC1
OCDS Trigger Set Control 1 Register(9FDD8_H)
Reset Value: 0000 0000_H


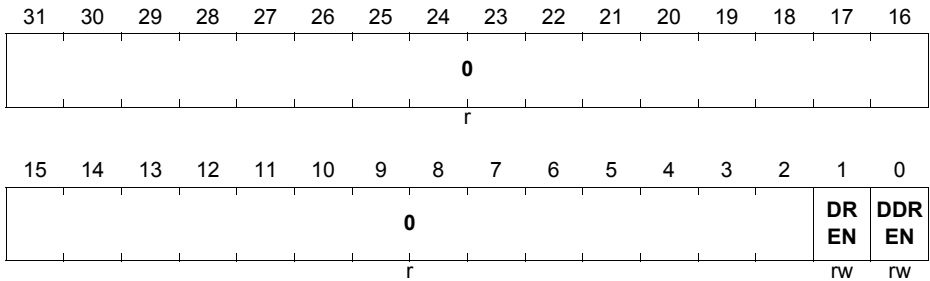
Field	Bits	Type	Description
MCS	[3:0]	rw	MCS Channel Select Required by TS16_MCA and TS32_MCD 0000 _B MCS Channel 0 0111 _B MCS Channel 7 1110 _B No MCS Channel (in conjunction with bit MCE set) 1111 _B All MCS Channels others , reserved
MI	[7:4]	rw	MCS Instance Required by TS16_MCA and TS32_MCD Index of the MCS instance. Index starts with 0, the max. value depends on the GTM configuration.
0	8	rw	Reserved Should be written with 0.
MOE	9	rw	MCS Opcode Trace Enable Required by TS32_MCD 0 _B Trigger Set TS32_MCD excludes opcode fetches 1 _B Trigger Set TS32_MCD includes opcode fetches

Generic Timer Module (GTM)

Field	Bits	Type	Description
0	[31:10]	r	Reserved Read as 0; must be written with 0.

ODA

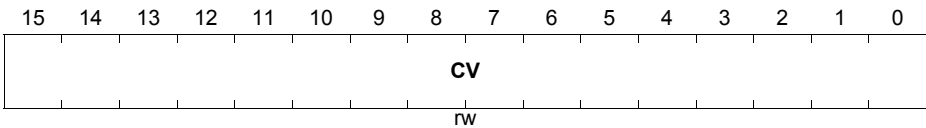
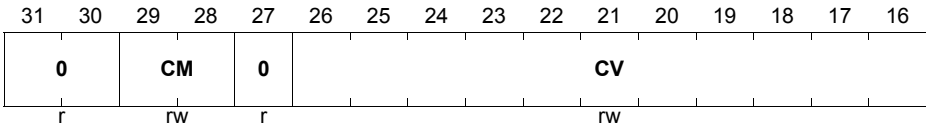
OCDS Debug Access Register (9FDDC_H) Reset Value: 0000 0000_H



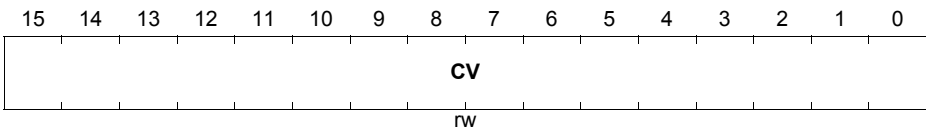
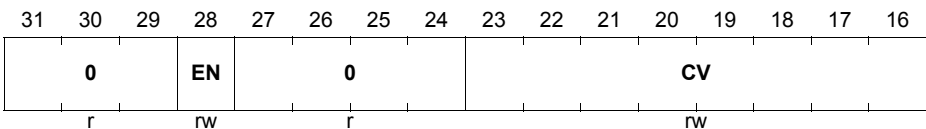
Field	Bits	Type	Description
DDREN	0	rw	Destructive Debug Read Enable For details see Table 26-106 .
DREN	1	rw	Destructive Read Enable For details see Table 26-106 .
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Table 26-106 Destructive Read Control encoding

ODA.DREN	ODA.DDREN	Description
0	0	Destructive read access are enabled for all masters beside the OCDS master
0	1	Destructive read access are enabled for all masters
1	0	Destructive read access are disabled for all masters
1	1	Destructive read access are disabled for all masters

Generic Timer Module (GTM)
OTBU0T
OCDS TBU0 Trigger Register
(9FDC4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CV	[26:0]	rw	Compare Value This value is compared to TBU_CH0_BASE register. As long as both match the associated TS16_IOS.MISC bit (Table 26-99) is active.
CM	[29:28]	rw	Compare Mode 0 _H Disabled 1 _H Compare lower 24 bits 2 _H Compare upper 24 bits 3 _H Compare all 27 bits
0	27, [31:30]	r	Reserved Read as 0; must be written with 0.

OTBU1T
OCDS TBU1 Trigger Register
(9FDC8_H)
Reset Value: 0000 0000_H


Generic Timer Module (GTM)

Field	Bits	Type	Description
CV	[23:0]	rw	Compare Value This value is compared to TBU_CH0_BASE register. As long as both match the associated TS16_IOS.MISC bit (Table 26-99) is active.
EN	28	rw	Enable 0 _B Disabled 1 _B Enabled
0	[27:24], [31:29]	r	Reserved Read as 0; must be written with 0.

OTBU2T
OCDS TBU2 Trigger Register (9FDCC_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		EN	0			CV									
r		rw	r			rw									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CV															
rw															

Field	Bits	Type	Description
CV	[23:0]	rw	Compare Value This value is compared to TBU_CH0_BASE register. As long as both match the associated TS16_IOS.MISC bit (Table 26-99) is active.
EN	28	rw	Enable 0 _B Disabled 1 _B Enabled
0	[27:24], [31:29]	r	Reserved Read as 0; must be written with 0.

26.23 **ision History**

Changes from version 0.3 to 1.0

- 2.2.1 Updated list of registers that drive value “illegal module access”
- 2.5 Fixed bugs in IRQ Specification
- 4.2 Clarified register bit settings in case of Trashbin functionality
- 4.5.2 Clarified register bit settings in case of Trashbin functionality
- 5.5.12,13 Fixed bad naming of register labels
- 10.4.2.5 Clarified modified behaviour of ECNT in mode TBCM
- 10.8.1,2 Clarified behaviour if bad TIM modes are selected
- 11.3.6 Clarified one shot mode in case of writing to CN0 while CN0 is counting
- 12.2.2 Clarified ATOM channel startup behaviour when ARU_EN is set and channel is enabled
- 12.3.1.1, 12.3.2.3, 12.3.3.5, 12.3.4.1 Not used bits can be written in any mode, but should be written as zero when not used
- 12.3.4 Clarified interrupt behaviour in SOMS mode: in SOMS mode only CCU1TC IRQ possible
- 12.6.2 Corrected some bit manipulation properties for bits OSM, TRIGOUT, CLK_SRC/CLK_SRC_SR
- 13.1.1ff Fixed declaration of maximum address range and clarified priorities of register accesses
- 14.1.1ff Documented additional memory layout parameters for the case that two large RAMs are connected to MCS
-
- 16.4.2 LOW_RES options
- 16.7.1 VTN
- 16.8.3ff NMB_T/S: MPVAL1/2 added
- 16.8.5ff PSTC/PSSC = 0 for FTD/FSD=0
- 16.8.6.7ff NMB_T/S calculation and use
- 16.11.1 Note 3): restriction for resolution choices
- 16.11.2 SWR and DEN
- 16.11.6 CRO, PSE, TOR, SOR, LOCK1/2
- 16.11.8/9 OSW and AOSV_2
- 16.11.28/29 old ADD_IN value used
- 16.13 no write protection of available RAM bits
- 16.13.9 THVAL for LOW_RES and !TSO_HRT
- 16.13.16ff all reciprocal values: SHL 32
- 16.14ff resolution of reciprocal values
- 20.1.1ff Activity checker removed
-
- 2.2.1 Updated list of registers that drive value “illegal module access”
- 2.5.4 Changed IRQ behaviour in single pulse mode on simultaneous clear and interrupt event
- 2.9.2 Protected write information

Generic Timer Module (GTM)

- 3.6.1 Clarified usage of ARU_ACCESS register
- 4.5.1 ARU read address only writeable if channel disabled
- 7.4.2 ARU read address only writeable if channel disabled
- 8.3, 8.5, 8.6: Removed confusing reset description
- 12.3 Clarified CN0 behaviour at channel enable
- 12.3.2ff SOMC mode description restructured
- 12.3.4ff SOMS mode description restructured
- 12.6.4 ARU read address only writeable if channel disabled
- 13.2.1 Specified behaviour in the case of bad opcode decoding
- 13.2.3 Added two non blocking ARU read instructions (NARD/NARDI)
- 13.2.4 Clarified behaviour on flags CY and V in arithmetic instructions
- 13.4.5 Added write protection to register MCS[i]_CH[x]_R6
- 16.6.2.1 higher reciprocal resolution TRIGGER
- 16.6.3.1 higher reciprocal resolution STATE
- 16.8.3.6 QDT is zero at the beginning
- 16.8.6.7 Store INC_CNTx when a valid event occurs and use it as MPx in step 5 or 25
- 16.11.1 condition for pulse number per increment
- 16.11.6 refined definitions of CRO, RCS, RCT, MT, MS, FTD, FSD and LOCKx
- 16.11.21 DPLL_ID_PMTR_x is now RPW
- 16.12.3 Bits 24:31 are read only
- 16.13.7ff Bits 24:31 are read only
- 16.13.12ff suppress update for unexpected missing TRIGGER or STATE respectively
- 16.13.16ff calculation of reciprocal values from predicted durations do not influence RCO
- 16.13.32ff Bits 24:31 are read only
- 16.13.43ff minor changes in formulations
- 16.13.47ff Bits 24:31 are read only
- 16.14.3 Bits 24:31 are read only
- 16.15.3 Bits 24:31 are read only
-
- 2.2.1 Updated list of registers that drive value “illegal module access”
- 9.4.3ff Address offset corrected
- 12.3.2ff SOMC mode description restructured
- 16.11.17 refined definitions of TINI and TAXI
-
- 2.2.1 Updated list of registers that may drive an aei_status signal with value “10”
- 2.5.4 Added note in single pulse mode
- 2.6ff Extended description of software debugger support (e.g. added lists of registers with special behaviour in the case of debugger access)
- 2.9.1 ision ID updated
- 2.9.3 Reset force protection includes SW RAM reset

Generic Timer Module (GTM)

- 2.9.9-11 Moved register description of AEI_bridge from module integration guide to this document. Added control and status information for diagnosis of the bridge status.
- 3.3.2 Clarified usage of debug mechanism
- 3.6.1 Extended functionality of ARU_ACCESS register
- 4.5.1 Note added
- 9.4.2 Second note added
- 11.2.2 Note to trigger signal added
- 11.8.2 Meaning of bits in ENDIS_CTRL changed
- 11.8.5 Meaning of bits in OUTEN_CTRL changed
- 12.2.2 Clarified ATOM channel startup behaviour in case of ARU_EN=1
- 12.3.2.1 Added note that less/equal compare only applies for comparisons against TBU_TS1/2
- 12.3.2.2.1 Added ACBI=00111 behaviour in table
- 12.3.2.3.3 Clarified WR_REQ and WRF bit behaviour for ATOM SOMC late update mechanism
- 12.3.2.4 Fixed ACB10 bit description
- 12.3.3.5 Clarified CLK_SRC_SR register description
- 12.3.4 Global ATOM SOMS mode behaviour update
- 12.3.4.6 Update SL and CLK_SRC_SR register description
- 12.6.2 Update SL and CLK_SRC_SR register description
- 13.4.11 Added feature to reset connected RAMs by software
- 15.1.1 Fixed minor specification issues
- 15.4.1 Completed SSL bit field description
- 16.5.10.2 Influence of SYSF to entry number
- 16.6.2.5 New Calculation for FULL_SCALE
- 16.6.2.7 New Meaning of CTO(N)
- 16.6.3.5 New Calculation for FULL_SCALE
- 16.6.3.7 New Meaning of CSO(N)
- 16.6.4.2 New Calculation for FULL_SCALE
- 16.6.5.2 New Calculation for FULL_SCALE
- 16.7ff Handling of actions in the past, considering of VTN, VSN and SYSF for calculations, use of RDT_T_FS1 and RDT_S_FS1
- 16.8.6.7 Use of NMB_T_TAR and NMB_S_TAR
- 16.10 New registers for RAM initialization, NMB_T_TAR and NMB_S_TAR
- 16.11.ff Notes to specify write abilities added. Influence of SYSF, new FST and FSS bits in NUTC/NUSC registers, new DCGI bit in DPLL_IRQ_NOTIFY register; new interrupt line connections; DPPLL_RAM_INI register
- 16.12ff Notes to specify write abilities added
- 16.13.4 NMB_T_TAR explained
- 16.13.8 NMB_S_TAR explained
- 16.13.16 Sign extension
- 16.13.17 Sign extension

Generic Timer Module (GTM)

- 16.14.1ff SYSF influence explained
- 18.5.2 Changed DPLL Interrupt ordering
-
- reworked MSC connection multiplexer structure together with controlling registers
- added forth set for MSC connections
- reworded old sets for MSC connections
- replace MSCxINCONy registers by MSCSETmCONn registers
- add registers MSCxINL/HCON registers
- removed second PSI5 connection register
- add GTM to Port Mapping for BGA-292
- add TIM0 inputs coming from ERU in SCU
- add TIM input connections for BGS-292
- add section for GTM to SENT connections
- add six additional triggers for CAN / TTCAN
- remove GTM to PSI5 trigger 1
- correct the address at the register image for register ADCTRIG0OUT1
- removed function for P32.1 (TOUT37 and TIN37)
- changes access limits for debug register from word to byte limitation

Changes from version 1.0 to 1.1

- add ATOM1 for ADC0 / 1 / 2 Trigger 0
- add ATOM0 for ADC0 / 1 / 2 Trigger 1
- replace ATOM5 with ATOM 3 for ADC5 / 6 / 7 Trigger 0/1
- skip ADC8 triggers
- shift ADC7 triggers to one group for ADC5 / 6 / 7 Triggers
- replace some TOM2 triggers with TOM0 triggers for ADC5 / 6 / 7 Trigger 0 / 1
- removed registers ADCTRIG0OUT1 and ADCTRIG1OUT1
- removed ATOM4 from MSC Set 4
- add register PSI5SOUTSEL
- removed DSADC inputs for TIM1 and TIM2 channels 6 and 7
- add connections from the ADC modules to GTM (end of conversion information)
- add debug improvements for TBU debugging

Changes from version 1.1 to 1.2

- change offset address of register DPLL_RAM_INI in the register image from 283BC_H to 281FC_H
- add debug improvements for TBU debugging
- bring back triggers for ADC 8
-
- 2.1.2 Restricted TOM/ATOM trigger lines to a certain length
- 2.3.1 Changed number of ARU data destinations
- 2.3.2 Introduced formula to determine the device dependent roundtrip time

Generic Timer Module (GTM)

- 2.6.1 Specified register behaviour of GTM_halt more precisely
- 2.9.1 ision ID updated
- 2.9.7 Note about protection included
- 3.6.1 Changed register bit field mode
- 3.6.11 Register bit 0 name corrected
- 3.6.12 Note about protection included
- 4.5.1 Reset value corrected
- 4.5.5 Note about protection included
- 5.1 Fixed specification error, FIFO RAM not initialized with zeros after reset
- 5.5.12 Note cleared automatically included
- 9.1 Specified behaviour of TBU_UPx signals more precisely
- 10ff Clarified indices descriptions of registers and bit fields
- 11.2.2 Added note for signed compare of ACT_TB
- 11.8.4 Changed naming ans description of bit field ACT_TB
- 11.8.27 note about write protection included
- 12.3.2.3.3 Specified behaviour of WR_REQ bit more precisely
- 12.3.4.1ff Specified ATOM SOMS mode more precisely
- 12.6.12 Note about write prtction included
- 13.1.1 Added hint for MCS RAM addresses and changed MCS block diagram
- 13.3.3 Note cleared by CPU included
- 13.4.6 Reset value corrected
- 13.4.9 Note protection included
- 13.4.11 Reset value corrected
- 13.4.11 Note reset values dependencies included
- 14.4.14 Note cleared by MCS channel included
- 16.5.10.1 typo corrected: APS pointer
- 16.7 number of actions calculated per increment in worst case and typical
- 16.5.10.1 typo corrected: APS pointer
- 16.7.1ff calculation of fractional part changed;decision of calculation case is dependent on the relation between q and m
- 16.7.7ff chapter ordering swapped with 16.7.8
- 16.8.4.2ff more detailed description
- 16.8.6.3ff direction change description changed
- 16.8.6.7 changed schedule of RAM update, PSTC and PSSC first value, SGE effect
- 16.10 introduced addresses for shadow registers, changed RAM addresses for calculated target pulse numbers
- 16.11.1 description of change between normal and emergency mode
- 16.11.2 write protection explained, more detailed description of SYN_NT/SYN_NS
- 16.11.12 explained differences for direction change in normal mode (SMC=0) and for SMC=1
- 16.11.13 explained direction change process
- 16.11.19 Note protection and cleared automatically included
- 16.11.24 more detailed explanation of address pointer value extension

Generic Timer Module (GTM)

- 16.11.33 New Register
- 16.11.34 New Register
- 16.11.35 New Register
- 16.11.36 New Register
- 16.12.2 explanation DLA value for low resolution
- 16.13.4 Removed NMB_T_TAR register
- 16.13.8 Removed NMB_S_TAR register
- 16.13.10 explanation TOV value for low resolution
- 16.13.11 explanation TOV_S value for low resolution
- 16.13.16 New RAM storage address
- 16.13.17 New RAM storage address
- 16.13.18 New RAM storage address
- 16.13.19 New RAM storage address
- 16.13.45 mode limited
- 16.13.46 mode limited
- 16.15.1ff start address of RAM2 changed
- SPE: Note about protection included
- 18.5.1ff Register bit mode changed to R
- 18.5.1ff Register bit mode changed to R
- 19.7.4 Note about protection included
- 21.3 Changed reference to address map
-
- 2.2.1 updated list of register which return AEI status of illegal write access
- 2.3.3 (wrong) ARU round trip time calculation removed
- 12.6.6 Note about protection included
- 12.6.8 Note about protection included
- 8.1.1 Added clock source MUX to FXU
- 8.8.ff Added clock selection bit to register CMU_CLK_5_CTRL
- 13.1.1 Updated figure in Architecture section
- 13.3.3 RAM ECC Error disables MCS channel and raises ERR bit
- 13.3.5ff Bit9 naming corrected
- 13.4.12ff Bit9 naming corrected
- 14.1 Reformulated MCFG Spec to be more generic and device independent
- 14.2.1 Reformulated MCFG_CTRL register to be more generic and device independent
- 16.5.10.2 explanation of RAM init procedure
- 16.6.2 explanation of the use of filter values
- 16.7.1ff precise case conditions
- 16.7.5.6ff precise case conditions
- 16.7.8.2ff precise case conditions
- 16.8.5 correct PSTC/PSSC calculation
- 16.11.2 precise SMC bit meaning
- 16.11.23 correct copy/paste error for INC_CNT2

Generic Timer Module (GTM)

- 18.2.3ff number of submodules (TIM, MCS, TOM and ATOM) described generic with “[i]”
- 18.3 Interrupt signals for submodules TIM, MCS, TOM and ATOM described generic with “[i]”
-
- 2.2.1 updated list of register which return AEI status of illegal write access
- 2.3.1ff Table of ARU sources corrected
- 8.5 Added clock selection bit to description
- 8.8.4ff Added clock selection bit to register CMU_CLK_5_CTRL
- 10.7 Corrected link to sub section in overview table
- 11.8.7 Description of bit field FUPD_CTRL
- 12.3.2.2.1 Condition of bit SLA added
- 12.3.2.3 Condition of bit SLA added
- 12.3.2.4 Description of new mode bit SLA
- 12.3.3.5 Bit 16 is not used instead of reserved
- 12.3.4.9 Bit 16 is not used instead of reserved
- 12.6.2 Description of new mode bit SLA added
- 13.1.1 Added ECC Error handling to specification
- 13.3ff Clarified some minor parts of the description
- 13.3.3 Clarified the behaviour of the ERR bit in STA register
- 16.6.2.7 Changed CTO/CTON behaviour in case of negative CDT_TX(_nom) values
- 16.6.3.7 Changed CSO/CSON behaviour in case of negative CDT_SX(_nom) values
- 16.6.4.4 Specified setting of CDT_TX(_nom) in case of negative values
- 16.6.5.4 Specified setting of CDT_SX(_nom) in case of negative values
- 16.7.6.1 Changed calculation of TSAC when action in far future (normal mode)
- 16.7.8.1 Changed calculation of TSAC when action in far future (emergency mode)
- 16.8.3.6 Removed deletion of FF’s in case of new trigger/state event
- 16.8.6.3ff Specify that nmb_t/s_tar_new values replaces nmb_t/s_tar values in case of direction change
- 16.10 reconfiguration of some table elements for better reading
- 16.11.6 Introduced RAM2_ERR status flag, active when access to not configured memory space happened
- 20.4 MON_ACTIVITY_1 register introduced
- 20.7 MON_ACTIVITY_1 register introduced
- 20.8.3 MON_ACTIVITY_1 register introduced
-
- update connection to QSPI
- update bit description for OTSC1.MOE
- remove CPU access to RAM from MCS trigger set TS16_MCA
- correct number of words for DPLL1B RAM in table GTM memory addresses
- add note to warn fro unexpected triggers
- remove bit discriptions in register ODA as explained in seperate table

Changes from version 1.2 to 1.3

- shift ERAY input from TIM channels 0 to channels 7
- update TIM input mapping for TC26x and QFP176
- add note for clock speeds in debug modes
- add information for BGA 416 and BGA 516
- update DSADC to GTM connections
 - add TIM0 inputs
 - move SCU inputs for TIM0 from 1110_B to 1100_B
 - add register DSADCINSEL0
- increase number of bit fields in register DSADCINSELx to 8 for all versions
- add DSADC inputs to all channels of TIM 1 and 2
- shift some DSADC TIM inputs in TIM mapping
-
- 2.1.2 trigger chain length description modified
- 2.3.1 Explain ARU round trip time more precise
- 2.5 ff Typos
- 2.6.1 Typo
- 2.9.9 Register structure updated; introduced Bit field SYNC_INPUT_REG
- 9.1 Removed TBU_TS0 indexing (x)
- 10.1 Removed specific channel number of eight
- 10.1 Removed TBU_TS0 indexing (z)
- 10.2.2 Corrected sentence
- 10.2.2.2.1 Corrected naming of filter mode
- 10.2.2.3.1 Corrected naming of filter mode
- 10.3.1 Corrected timeout calculation formula
- 10.4.2.5 Removed specific channel number of eight
- 10.6 Removed specific channel number of eight
- 10.7 Removed specific channel number of eight
- 10.8.1-13 Removed specific channel number of eight
- 10.8.14 Inserted note on register bit width dependency
- 12.3.2.4 bit ACB10 description
- 12.6.12 bit 1 TRG CCU1TC description
- 16.5.3 declaration of limits for SUB_INC1,2 clocks
- 16.8.3 after eq. 16.25b, 16.26b: explanation of SUB_INC1,2 frequency doubling possibilities
- 16.8.6.3 explanation of “make calculation” steps 3, after 5 : nmb_t replaced by nmb_t_tarin emergency step 2: explanation added at the end of chapter: some corrections
- 16.8.6.4 some corrections like 16.8.3.8 at end
- 16.10 explanation of TBU_TS_x and INC_CNT1,2 RAM Region 2 end depends on device chosen
- 16.11.1 Bits 24:11 explained more detailed

Generic Timer Module (GTM)

- 16.11.3 Bits 3,4: explained more detailed, correction
- 16.11.24,25 heading correction, Bits 5:0 explanation
- 18.3 in table GTM_DPLL_XXX description corrected
- 18.5.1 bit 24 and 25 description corrected
- 18.5.12 description corrected
- 20.8.3 MON_ACTIVITY_1 register at address 0x08
- 2.1.3 new functionality GTM signal multiplexing added
- 2.2.1 CMU_CLK_CTRL, CMU_FXCLK_CTRL, TIM[i]_CH[x]_GPR1 included
- DPLL_PSA, DPLL_DLA, DPLL_NA, DPLL_DTA removed
- 2.5 ff new error interrupt
- 2.8 new address map due to new register
- 2.9.1 bit field YEAR replaced by STEP
- 2.9.9 NOTE text improved
- 2.9.12 new error interrupt
- 2.9.13 new register GTM_TIM[i]_AUX_IN_SRC
- BRC new error interrupt
- 5.4 new error interrupt
- 5.5.14 new error interrupt
- 8.5 selection of clocks for fxclk
- 8.7 new CMU_FXCLK_CTRL
- 8.8.1 clarifying disable/enable functionality
- 8.8.5 no exception for CMU_CLK5_CTRL
- 8.8.9 selection of clocks for fxclk
- 10.1.1 Block diagram updated due to new functionality INPSRC, EXTCAPSRC
- 10.1.2 new functionality INPSRC selection added
- 10.1.3 new functionality EXTCAPSRC selection added
- 10.3 TDU Block diagram and description updated due to new functionality of edge selection
- 10.4.1 TIM channel block diagram and description updated due to new functionality (ECNT increased to 16 bit, ECNT can be captured to GPRx_REG, External_capture)
- 10.4.2ff New functionality External_capture, EGPRx_SEL added to existing TIM modes
- 10.4.2.6 New TIM mode TGPS added
- 10.7 Configuration registers updated due to new added functions
- 10.8.1-2 New bitfields added to support new functionality
- 10.8.5 Register removed, rearranged to TDUV, TDUC register
- 10.8.15-20 Register description for new functions added
- 11.4 TOM_CH2 special mode hint added
- 11.8.17 CLK_SRC_SR bits: description improved
- RST_CCU0 bit: note added
- SPEM bit: note added
- CLK_SRC_SR bits: description improved
- 12.3.2.2.1ff table design corrected

Generic Timer Module (GTM)

- 12.3.2.2.1ff TRIGOUT bit: added
- CLK_SRC_SR bits: NOTE added
- ACBO bits: NOTE added
- 13.1.1 Clarified usage of write accesses to MCS registers in the case of multiple source. Usage of accessing common trigger register is clarified.
- 13.4.11 Added Error interrupt register MCS[i]_CH[x]_EIRQ_EN
- 13.4.13-14 Added Note for writing common trigger bits via AEI
- 15.1.1 figure updated, description of new DIR selection added
- 15.4.1 TSEL bit: TIM0_IN6 feature added
- LSEL bit: new configuration bit
- 16.6.2ff TRIGGER time stamp resolution for filter values considered
- 16.6.3ff STATE time stamp resolution for filter values considered
- 16.7ff up to 32 actions served for device 4
- 16.10 additional registers and address map changed
- 16.11.2 some control bits added
- 16.11.6 control register for 8 additional actions
- 16.11.78 control bits for 8 additional actions
- 16.11.20-21 additional EIRQ enable register
- 16.11.30-35 status register with changed address and omitting CTON/CSON
- up to 32 ID_PMTM register changed SHADOW register bits
- 16.12.1-4 extension to serve 32 actions
- 16.13.38-39 centralization for TRIGGER and STATE calculation
- 16.15ff additional register address range for changed and additional registers to serve up to 32 actions
- 17.2.2 figure updated
- 17.2.4 new chapter for SPE revolution detection
- 17.3 new interrupt SPE_RCMP
- 17.4 new register added
- 17.5.5-6 new register
- 17.5.7 additional IRQ
- 17.5.8 additional IRQ
- 17.5.9 additional IRQ
- 17.5.11 new register
- 18.3 new GTM Error Interrupt
- 18.4 new BRC, FIFO[i]_CH[x], TIM[i]_CH[x], MCS[i]_CH[x], SPE[i], CMP, DPLL, GTM Error Interrupt register
- 18.5.13-18 description of new Error Interrupt Register
- 19.5 new error interrupt
- 19.6 new error interrupt
- 19.7.6 description of new Error Interrupt Register
- 20.5 Note concerning ARU roundtrip time measurement by MCS added
- 10.1.3 EXTCAP_SRC: enhanced functionality description

Generic Timer Module (GTM)

- 10.4.1 new functionality: TIM input signal value available via ECNT[0] if channel is disabled
- 10.4.2 ff EXT_CAP_EN: enhanced functionality description. Added to each TIM channel description of external capture behaviour.
TGPS mode: enhanced functionality description. new functionality: in TGPS mode GPR1 operates as a shadow register for CNTS
- 10.8.7 Register description updated due to added TGPS functionality
- 2.2.1 List of registers and address ranges which return AEI status “10” corrected
- 3.3.1 Note for bit 12 (RREQ) and bit 13 (WREQ) of register ARU_ACCESS added
- 10.8.19 Note added to ECNT register (behaviour of ECNT if channel gets disabled)
- 14ff layout option changed to layout configuration
- 16.4ff clear index 9
- 16.4.2 no hex for width
- 16.5.5 IP like formulation of clock frequency requirements
- 16.5.6 Remove uncleared abbreviation “CDG”
- 16.6.2 Reduction of CTON, CTO function to just CTO flag
- 16.6.3 Reduction of CSON, CSO function to just CSO flag
- 16.11.30 Clarification on SYS/SYT flags, because not influenced for TOR/SOR states
- 18.5.14ff correct naming IRQ to EIRQ
- 19.1.1 include CMP_EIRQ
- 19.3 include CMP_EIRQ_EN
- 19.5 include CMP_EIRQ
- changed reset value for TOM and ATOM CTRL registers SL bits from 0 to 1
-
- changed reset value for TOM and ATOM CTRL registers OL bits from 1 to 0
- update information about access protection

Changes from version 1.3 to 1.4

- correct typo in ADCx Timer Trigger tables so that it matches with the register description
- update GTM connections to ports for TC26x
 - add TOM1_12 and ATOM0_4 for P10.0
 - add TOM1_6 and ATOM1_6 for P10.4
 - add TOM1_8 and ATOM0_0 for P10.7
 - add TOM1_13 and ATOM0_5 for P10.8
 - add TIM2_2, TOM1_4 and ATOM3_4 for P14.10
 - add TOM1_0 and ATOM3_0 for P14.7
 - add TIM2_3 and TOM1_2 for P14.8
 - add TIM2_2 and TOM1_3 for P14.9
 - add TIM0_3 and TOM0_3 for P20.1
 - add TIM0_3 and TOM1_8 for P21.0
 - add TIM0_4 and TOM1_9 for P21.1

Generic Timer Module (GTM)

- add TOM1_2 and ATOM1_2 for P23.5
- update GTM connections to ports for TC29x
 - move timer mapping for P00.13 to P00.15
 - move timer mapping for P00.15 to P00.13
 - update timer mapping for P01.0
 - move timer mapping for P01.09 to P01.1
 - move timer mapping for P01.13 to P01.10
 - move timer mapping for P01.15 to P01.11
 - move timer mapping for P01.08 to P01.12
 - move timer mapping for P01.11 to P01.13
 - move timer mapping for P01.02 to P01.15
 - move timer mapping for P01.01 to P01.2
 - add P01.3 to timer mapping
 - move timer mapping for P01.12 to P01.8
 - move timer mapping for P01.10 to P01.9
 - replace for P02.0 TOM0_12 by TOM0_8 and TOM1_12 by TOM1_8
 - move timer mapping for P02.14 to P02.13
 - move timer mapping for P02.15 to P02.14
 - move timer mapping for P02.13 to P02.15
 - move timer mapping for P10.13 to P10.11
 - move timer mapping for P10.15 to P10.12
 - move timer mapping for P10.12 to P10.13
 - move timer mapping for P10.11 to P10.14
 - move timer mapping for P10.14 to P10.15
 - move timer mapping for P14.15 to P13.14
 - move timer mapping for P14.11 to P13.09
 - move timer mapping for P13.09 to P14.11
 - move timer mapping for P13.12 to P14.12
 - move timer mapping for P13.11 to P14.13
 - move timer mapping for P13.10 to P14.14
 - move timer mapping for P13.14 to P14.15
 - add P26.0 to timer mapping
 - removed P34.0
 - 11.2.1 bit field name CLK_SRC_STAT replaced
 - 11.8.1 bit field name CLK_SRC_STAT replaced
 - 12.6.1 bit field name CLK_SRC_STAT replaced
 - 13.2.2.10 reference to ACB replaced by reference to MHB
 - 16.13.7 more precisely explanation of THMI
 - 16.13.8 more precisely explanation of THMA
 - 16.13.9 more precisely explanation of THVAL
 - 17.5.6 note related to SPEi_RCMP interrupt added
 - remove GTM to QSPI connections
 - change encoding for TS16_DRA1A/1BC/2 DPLL RAM Access Address table

Generic Timer Module (GTM)

- updated register names for ATOMi_CHx_CTRL, TS_T, TS_T_OLD, PSTM, PSTM_OLD, PSSM, and PSSM_OLD for header generation

Changes from version 1.4 to 1.5

- removed second vadc_G6ARBCNT inputs for TIM1 at position 1100
- update end addresses for the DPLL RAMs in the RAM address table
- removed TIM2_5 connection to P34.5
- aligned for TC26x and TC24x the pad to TIM input mapping
- 2.3.3 round trip time advice
- 2.5 correct register name from (...CEI1 to ...CEI5) to (...CEI0 to ...CEI4)
- 2.5.4 exception advice
- 5.1 Explanation of flushing the FIFO after write access to START_ADDR or END_ADDR register
- 5.2.2 Further explanation of ring buffer mode
- 5.5.2ff Note added for further explanation
- 5.5.4ff Note added for further explanation
- 8.3 more precisely explanation of using clocks
- 10.1.2 more precisely explanation of F_{in}(x)
- 10.4.2.5 removed wrong explanation that channels 1 to m-1 have to be disabled
- 10.4.2.6 corrected TGPS mode explanation (counting is done via selected CMU clock)
- 11.1 new indices definition
- 11.3.6 additional one shot mode handling hint
- 12.3.3.4 additional one shot mode handling hint
- 12.6.1 UPEN_CTRL0 Note deleted
- 16.10.5.1ff address relation and names in EXT register region and RAM2 precised
- 16.6.2ff signal names changed
- 16.7 operation times precised for new version
- 16.8.3.4 description of sub_inc generation conditions
- 16.11.2 relation of CMU_CLKo to system clock
- 16.11.3 - 6,8 activity of the enable bits only for DEN=1, reset of SWON_x for DEN=0
- 17.5.4 additional note to SPE_OUT_CTRL
- 182.5ff more precisely explanation of interrupt bundling
- correct typos

Changes from version 1.5 to 1.6

- removed bit OTSC1.MCE as this bit is not required for a correct trace
- correct typos
- replace connections for DSADC1 with DSADC3 for TC26x and TC24x
- removed GxARBCNT for x>3 of TC26x and TC24x
- changed P01.8 TIN158 / TOUT158 to TIN162 / TOUT162
- add ADC output trigger 4 for TC24x

Generic Timer Module (GTM)

- remove connections to and from P10.12
- remove connections to and from P13.8
- remove connections to and from P15.9
- added QFP-144 pinning for TC24x
- remove P00.5 for QFP-80
- remove P00.9 for QFP-100
- and hint for DPLL RAM 1a access errors of TC26x
- added DSADC and ADC connections for TC29x
- add missing bit field for ADC 3 to registers ADCTRIG0OUT0 and ADCTRIG1OUT0 for TC24x
- 2.5 more detailed description of IRQ_NOTIFY
- 2.9.13 bit 0 formula corrected
- 5.5.3 end replaced by start
- 5.5.4 upper replaced by lower
- 5.5.6 EMPTY and FULL description corrected
- 10.2.3 more detailed information about filter reconfiguration
- 10.5 more detailed information about MAP interface
- 10.8.1ff more detailed information about TIM MODE reconfiguration and GELx_SEL
- 10.8.6 additional information about GPR0
- 10.8.7 additional information about GPR1
- 10.8.9 additional information about CNTS
- 10.8.20 additional information about input selection by EXT_CAP_SRC
- 11.3 more detailed specification behavior in case of reset of CN0 by CCU0 compare and trigger
- 12.3.2.3 additional information about compare strategy; figure and description above added
- 12.3.3ff more detailed specification of behavior in case of reset of CN0 by CCU0 compare and by trigger
- 12.3.3.4 figure description corrected
- 12.3.3.5 more detailed information about CLK_SRC_SR and RST_CCU0 in SOMP mode
- 12.6.2 additional / corrected notes for bit SL
- 13.4.1 added missing Note for MEM_ERR_IRQ bit
- 14.1ff new chapter for memory layout configurations and for memory layout parameters
- 16ff all equations numbering transformed from 16.xxx to DPLL-xxx

Changes from version 1.6 to 1.7

- add register MSC0INEXTLCON for TC24x
- removed PSI5 and PSI5-S connections for TC24x
- make some minor correct in the pinning list for TC24x QFP100

Changes from version 1.7 to 1.8

- add register MSC0INEXTLCON for TC26x
- update to Bosch specification version 1.5.5.1:
- 2.2.1 Additional information about status '10'
- 2.4.1 precising the number of generated clocks
- 2.5.1ff corrected signal nmae as same in figure
- 2.9.1 Additional information about bits and values
- 4.5.3 precising information about DID
- 5.5.1 precising information about WULOCK
- 8.3.6 precising the number of generated clocks
- 8.8.1 removed note: Any disabling to EN_ECLKz will be reset internal counters for external clocks
- 10.1.1 replaced GPRz by GPR0 and GPR1
- 10.1.2 state bit VAL_x(1) and MODE_x(1) introduced in diagram and textual description
- 10.1.3.1 precising enable external capture functionality
- 10.3.1 replaced GPRz by GPR0 and GPR1; Extended description of ACB(2:0) bit behavior in case of a timeout event
- 10.4.1 replaced GPRz by GPR0 and GPR1
- 10.4.2 replaced GPRzOFL by GPROFL
- 10.4.2.1 Extended description of CNTS register update
- 10.4.2.3 replaced GPRzOFL by GPROFL
- 10.6 replaced GPRz by GPR0 and GPR1
- 10.6 replaced GPRzOF by GPROFL
- 10.8.2. replaced GPRz by GPR0 and GPR1
- 10.8.10-12 replaced GPRzOFL by GPROFL
- 10.8.15 Description of VAL_0 and MODE_0 enhanced
- 10.8.16 replaced GPRzOFL by GPROFL
- 10.8.20 bit width of EXT_CAP_SRC corrected; new values added
- 11.3.6.1 precising of one shot mode handling
- 11.3.7.1 precising of pulse count modulation mode
- 12.3.2.2 precising description of serve last compare strategy
- 12.3.3.4. precising of one shot mode handling
- 12.6.9 additioanl note for RST_CCU0
- 13.3.3 precising usage of instruction in conjunction with STA
- 16.8.6.7 replaced invalid by inactive
- 16.11 precising section headline and content
- 16.11ff replaced lower case letters in bit fields by upper case letters; added missing prefix DPLL_ to all registers or momory labels
- 16.11.1 precising bit fields 15:11 and 24:16
- 16.11.7 replaced bit field 23:0 by 31:0; additional note for limitations
- 16.11.14 precising NUTE and WNUT definitions

Generic Timer Module (GTM)

- 16.11.17ff replaced lost by loss
- 16.11.30 precisising SOR, MS, TOR, LOCK2, LOCK1 definition
- 16.11.37ff replaced section 16.12ff by 16.11.37-40
- 16.11.41ff replaced section 16.13ff by 16.11.41-91
- 16.11.50 precisising TOV definition
- 16.11.51 precisising TOV_S definition
- 16.11.54 precisising MPVAL1 definition
- 16.11.55 precisising MPVAL2 definition
- 16.11.76 precisising TLR definition
- 16.11.77 precisising SLR definition
- 16.11.92ff replaced section 16.14ff by 16.11.92-95
- 16.11.94 precisising PD_S definition
- 16.11.98ff replaced section 16.15ff by 16.11.96-98
- 16.11.98 replaced ACBj by ACB
- 16.12ff added missing prefix DPLL_ to all register or memory labels
- 16.12.1 precisising RDT_T definition
- 16.12.2 precisising TSF_T definition
- 16.12.3 precisising PD NT definition
- 16.12.4 precisising DT_T definition
- 20.4 replaced MCS[z]_CH[x]_MCA by MCA_i_x
- 20.8.1 replaced MCS[x] by MCSx
- 20.8.2,3 replaced MCS[z]_CH[x]_MCA by MCA_i_x

Changes from version 1.8 to 1.9

- correct error in BGA-416 TOUT numbering for 10.13 and 10.11
- connection shown in figures and register for SET5 and SET6 to MSC2 was corrected
- update description of bit field OCS.SUS
- update description of (A)TOM SOMP Mode
- clarify reset behavior of debug registers
- correct error for 2.11; it is connected to TIM4_4 not to TIM4_7
-

27 Capture/Compare Unit 6 (CCU6)

The CCU6 is a high-resolution 16-bit capture and compare unit with application specific modes, mainly for AC drive control. Special operating modes support the control of Brushless DC-motors using Hall sensors or Back-EMF detection. Furthermore, block commutation and control mechanisms for multi-phase machines are supported.

It also supports inputs to start several timers synchronously, an important feature in devices with several CCU6 kernels.

This chapter is structured as follows:

- Introduction (see [Section 27.1](#))
including register overview (see [Section 27.1.3](#))
- Operating T12 (see [Section 27.2](#))
including T12-related registers (see [Section 27.2.8](#))
and capture/compare control registers (see [Section 27.2.9](#))
- Operating T13 (see [Section 27.3](#))
including T13-related registers (see [Section 27.3.6](#))
- Synchronous start feature (see [Section 27.4](#))
- Trap handling (see [Section 27.5](#))
- Multi-Channel mode (see [Section 27.6](#))
- Hall sensor mode (see [Section 27.7](#))
- Modulation control registers (see [Section 27.8](#))
- Interrupt handling (see [Section 27.9](#))
including interrupt registers (see [Section 27.9.2](#))
- General module operation (see [Section 27.10](#))
including general registers (see [Section 27.10.5](#))
and BPI registers (see [Section 27.10.6](#))
- Module implementation (see [Section 27.11](#))

27.1 Introduction

The CCU6 unit is made up of a Timer T12 Block with three capture/compare channels and a Timer T13 Block with one compare channel. The T12 channels can independently generate PWM signals or accept capture triggers, or they can jointly generate control signal patterns to drive AC-motors or inverters.

A rich set of status bits, synchronized updating of parameter values via shadow registers, and flexible generation of interrupt request signals provide means for efficient software-control.

*Note: The capture/compare module itself is named CCU6 (capture/compare unit 6).
A capture/compare channel inside this module is named CC6x.*

27.1.1 Feature Set Overview

This section gives an overview over the different building blocks and their main features.

Timer 12 Block Features

- Three capture/compare channels, each channel can be used either as capture or as compare channel
- Generation of a three-phase PWM supported (six outputs, individual signals for high-side and low-side switches)
- 16-bit resolution, maximum count frequency = peripheral clock
- Dead-time control for each channel to avoid short-circuits in the power stage
- Concurrent update of T12 registers
- Center-aligned and edge-aligned PWM can be generated
- Single-shot mode supported
- Start can be controlled by external events
- Capability of counting external events
- Many interrupt request sources
- Hysteresis-like control mode

Timer 13 Block Features

- One independent compare channel with one output
- 16-bit resolution, maximum count frequency = peripheral clock
- Concurrent update of T13 registers
- Can be synchronized to T12
- Interrupt generation at period-match and compare-match
- Single-shot mode supported
- Start can be controlled by external events
- Capability of counting external events

Additional Specific Functions

- Block commutation for Brushless DC-drives implemented
- Position detection via Hall-sensor pattern
- Noise filter supported for position input signals
- Automatic rotational speed measurement and commutation control for block commutation
- Integrated error handling
- Fast emergency stop without CPU load via external signal ($\overline{\text{CTRAP}}$)
- Control modes for multi-channel AC-drives
- Output levels can be selected and adapted to the power stage

Capture/Compare Unit 6 (CCU6)

27.1.2 Block Diagram

The Timer T12 can operate in capture and/or compare mode for its three channels. The modes can also be combined (e.g. a channel operates in compare mode, whereas another channel operates in capture mode). The Timer T13 can operate in compare mode only. The multi-channel control unit generates output patterns which can be modulated by T12 and/or T13. The modulation sources can be selected and combined for the signal modulation.

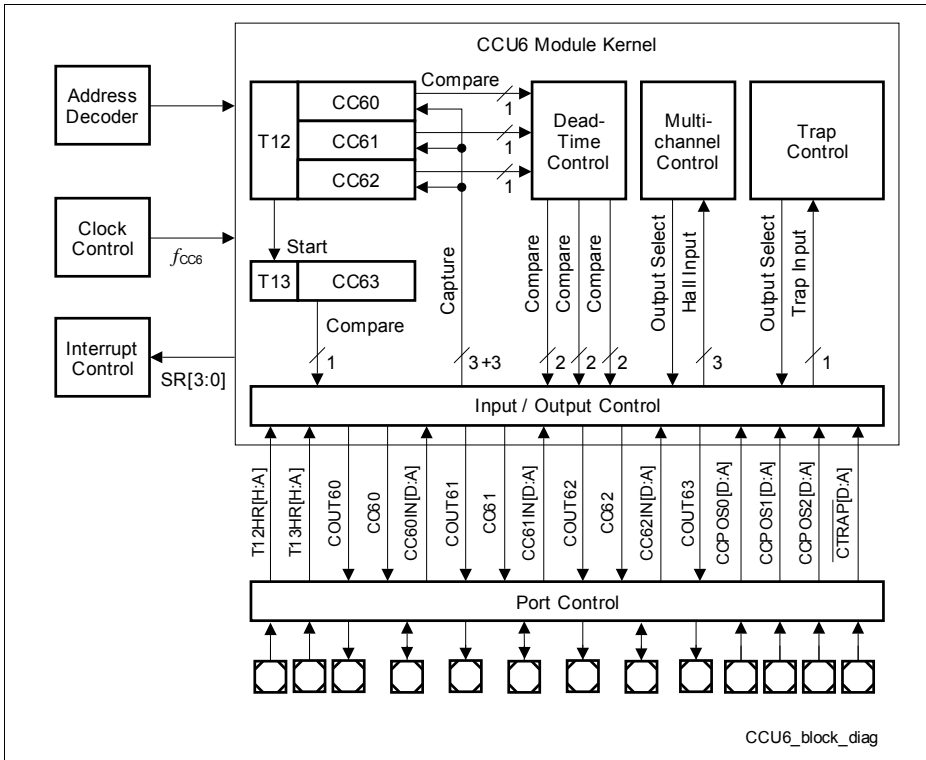


Figure 27-1 CCU6 Block Diagram

27.1.3 CCU6 Kernel Registers

For the generation of the overall register table, the prefix “CCU6x_” has to be added to the register names in this table to identify the registers of different CCU6 kernels that are implemented. In this naming convention, x indicates the kernel number.

Table 27-1 shows all registers required for programming of a CCU6. It summarizes the CCU6 kernel registers and defines their offset addresses. BPI registers are listed separately in **Table 27-14**, and Module Output Select registers are listed in **Table 27-16**.

The CCU6 module has no destructive register read mechanisms.

CCU6 Kernel Register Overview

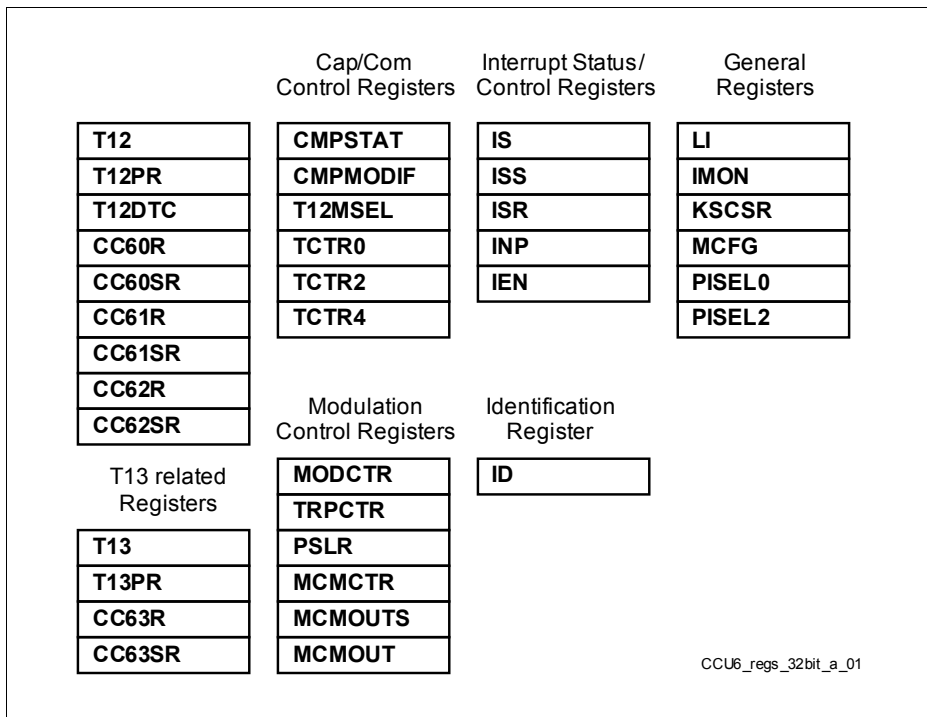


Figure 27-2 CCU6 Registers

Note: In the case of a write access to addresses inside the address range (that is covered by the same chip select signal), but that are not the addresses explicitly mentioned for the module, the write access is not taken into account for the

Capture/Compare Unit 6 (CCU6)

module. The same principle is valid for read accesses. In case of a read access to another address, the module does not react.

Note: The exact register address is given by the relative address of the register (given in [Table 27-1](#)) plus the kernel base address (given in [Table 27-15](#)) of the kernel.

Table 27-1 CCU6 Module Registers

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Reset	Page Num.
			Read	Write		
General Registers						
ID	Module Identification Register	08 _H	U, SV	BE	Application Reset	27-115
PISEL0	Port Input Select Register 0	10 _H	U, SV	U, SV, P	Application Reset	27-116
PISEL2	Port Input Select Register 2	14 _H	U, SV	U, SV, P	Application Reset	27-118
KSCSR	Kernel State Control Sensitivity Register	1C _H	U, SV	U, SV, P	Debug Reset	27-127
MCFG	Module Configuration Register	04 _H	U, SV	U,SV, P	Application Reset	27-121
IMON	Input Monitoring Register	98 _H	U, SV	U, SV, P	Application Reset	27-122
LI	Lost Indicator Register	9C _H	U, SV	U, SV, P	Application Reset	27-125
Timer T12 related Registers						
T12	Timer 12 Counter Register	20 _H	U, SV	U, SV, P	Application Reset	27-33
T12PR	Timer 12 Period Register	24 _H	U, SV	U, SV, P	Application Reset	27-34
T12DTC	Dead-Time Control Register for Timer T12	28 _H	U, SV	U, SV, P	Application Reset	27-37
CC60R	Capture/Compare Register Channel CC60	30 _H	U, SV	U,SV, P	Application Reset	27-35
CC61R	Capture/Compare Register Channel CC61	34 _H	U, SV	U,SV, P	Application Reset	27-35

Capture/Compare Unit 6 (CCU6)
Table 27-1 CCU6 Module Registers (cont'd)

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Reset	Page Num.
			Read	Write		
CC62R	Capture/Compare Register Channel CC62	38 _H	U, SV	U,SV, P	Application Reset	27-35
CC60SR	Capture/Compare Shadow Register Channel CC60	40 _H	U, SV	U, SV, P	Application Reset	27-36
CC61SR	Capture/Compare Shadow Register Channel CC61	44 _H	U, SV	U, SV, P	Application Reset	27-36
CC62SR	Capture/Compare Shadow Register Channel CC62	48 _H	U, SV	U, SV, P	Application Reset	27-36

Capture/Compare Control Registers

CMPSTAT	Compare State Register	60 _H	U, SV	U, SV, P	Application Reset	27-39
CMPMODIF	Compare State Modification Register	64 _H	U, SV	U, SV, P	Application Reset	27-42
T12MSEL	T12 Capture/Compare Mode Select Register	68 _H	U, SV	U, SV, P	Application Reset	27-43
TCTR0	Timer Control Register 0	70 _H	U, SV	U, SV, P	Application Reset	27-44
TCTR2	Timer Control Register 2	74 _H	U, SV	U, SV, P	Application Reset	27-48
TCTR4	Timer Control Register 4	78 _H	U, SV	U, SV, P	Application Reset	27-51

Timer T13 related Registers

T13	Timer 13 Counter Register	50 _H	U, SV	U, SV, P	Application Reset	27-66
T13PR	Timer 13 Period Register	54 _H	U, SV	U, SV, P	Application Reset	27-67
CC63R	Compare Register for Timer 13	58 _H	U, SV	U,SV, P	Application Reset	27-68

Capture/Compare Unit 6 (CCU6)
Table 27-1 CCU6 Module Registers (cont'd)

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Reset	Page Num.
			Read	Write		
CC63SR	Compare Shadow Register for Timer 13	5C _H	U, SV	U, SV, P	Application Reset	27-69

Modulation Control Registers

MODCTR	Modulation Control Register	80 _H	U, SV	U, SV, P	Application Reset	27-83
TRPCTR	Trap Control Register	84 _H	U, SV	U, SV, P	Application Reset	27-85
PSLR	Passive State Level Register	88 _H	U, SV	U, SV, P	Application Reset	27-88
MCMOUTS	Multi-Channel Mode Output Shadow Register	8C _H	U, SV	U, SV, P	Application Reset	27-92
MCMOUT	Multi-Channel Mode Output Register	90 _H	U, SV	U, SV, P	Application Reset	27-93
MCMCTR	Multi-Channel Mode Control Register	94 _H	U, SV	U, SV, P	Application Reset	27-89

Interrupt Status and Node Registers

IS	Interrupt Status Register	A0 _H	U, SV	U, SV, P	Application Reset	27-98
ISS	Interrupt Status Set Register	A4 _H	U, SV	U, SV, P	Application Reset	27-101
ISR	Interrupt Status Reset Register	A8 _H	U, SV	U, SV, P	Application Reset	27-103
INP	Interrupt Node Pointer Register	AC _H	U, SV	U, SV, P	Application Reset	27-108
IEN	Interrupt Node Pointer Register	B0 _H	U, SV	U, SV, P	Application Reset	27-105

27.2 Operating Timer T12

The timer T12 block is the main unit to generate the 3-phase PWM signals. A 16-bit counter is connected to 3 channel registers via comparators, that generate a signal when the counter contents match one of the channel register contents. A variety of control functions facilitate the adaptation of the T12 structure to different application needs. Besides the 3-phase PWM generation, the T12 block offers options for individual compare and capture functions, as well as dead-time control and hysteresis-like compare mode.

This section provides information about:

- T12 overview (see [Section 27.2.1](#))
- Counting scheme (see [Section 27.2.2](#))
- Compare modes (see [Section 27.2.3](#))
- Compare mode output path (see [Section 27.2.4](#))
- Capture modes (see [Section 27.2.5](#))
- Shadow transfer (see [Section 27.2.6](#))
- T12 operating mode selection (see [Section 27.2.7](#))
- T12 register description (see [Section 27.2.8](#))

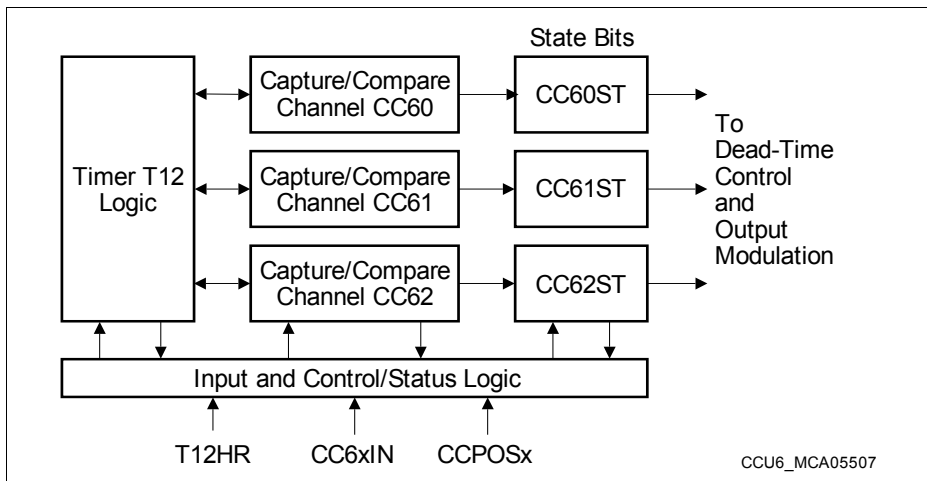


Figure 27-3 Overview Diagram of the Timer T12 Block

27.2.1 T12 Overview

Figure 27-4 shows a detailed block diagram of Timer T12. The functions of the timer T12 block are controlled by bits in registers **TCTR0**, **TCTR2**, and **PISELO**.

Timer T12 receives its input clock (f_{T12}) from the module clock f_{CC6} via a programmable prescaler and an optional 1/256 divider or from an input signal T12HR. These options are controlled via bit fields T12CLK and T12PRE (see Table 27-2). T12 can count up or down, depending on the selected operation mode. A direction flag, CDIR, indicates the current counting direction.

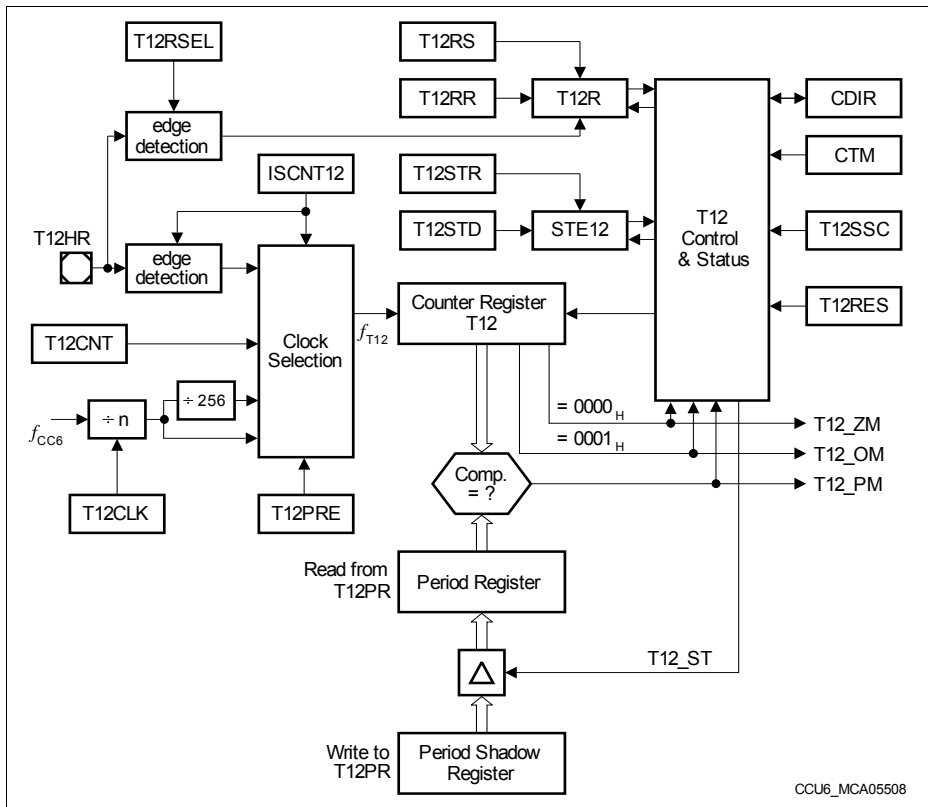


Figure 27-4 Timer T12 Logic and Period Comparators

Via a comparator, the T12 counter register **T12** is connected to a Period Register **T12PR**. This register determines the maximum count value for T12.

In Edge-Aligned mode, T12 is cleared to 0000_H after it has reached the period value defined by T12PR. In Center-Aligned mode, the count direction of T12 is set from 'up' to

Capture/Compare Unit 6 (CCU6)

'down' after it has reached the period value (please note that in this mode, T12 exceeds the period value by one before counting down). In both cases, signal T12_PM (T12 Period Match) is generated. The Period Register receives a new period value from its Shadow Period Register.

A read access to T12PR delivers the current period value at the comparator, whereas a write access targets the Shadow Period Register to prepare another period value. The transfer of a new period value from the Shadow Period Register into the Period Register (see [Section 27.2.6](#)) is controlled via the 'T12 Shadow Transfer' control signal, T12_ST. The generation of this signal depends on the operating mode and on the shadow transfer enable bit STE12. Providing a shadow register for the period value as well as for other values related to the generation of the PWM signal allows a concurrent update by software for all relevant parameters.

Two further signals indicate whether the counter contents are equal to 0000_H (T12_ZM = zero match) or 0001_H (T12_OM = one match). These signals control the counting and switching behavior of T12.

The basic operating mode of T12, either Edge-Aligned mode ([Figure 27-5](#)) or Center-Aligned mode ([Figure 27-6](#)), is selected via bit CTM. A Single-Shot control bit, T12SSC, enables an automatic stop of the timer when the current counting period is finished (see [Figure 27-7](#) and [Figure 27-8](#)).

The start or stop of T12 is controlled by the Run bit T12R that can be modified by bits in register [TCTR4](#). The run bit can be set/cleared by software via the associated set/clear bits T12RS or T12RR, it can be set by a selectable edge of the input signal T12HR ([TCTR2.T12RSEL](#)), or it is cleared by hardware according to preselected conditions.

The timer T12 run bit T12R must not be set while the applied T12 period value is zero. Timer T12 can be cleared via control bit T12RES. Setting this write-only bit does only clear the timer contents, but has no further effects, for example, it does not stop the timer.

The generation of the T12 shadow transfer control signal, T12_ST, is enabled via bit STE12. This bit can be set or reset by software indirectly through its associated set/clear control bits T12STR and T12STD.

While Timer T12 is running, write accesses to the count register T12 are not taken into account. If T12 is stopped and the Dead-Time counters are 0, write actions to register T12 are immediately taken into account.

27.2.2 T12 Counting Scheme

This section describes the clocking and counting capabilities of T12.

27.2.2.1 Clock Selection

In **Timer Mode** (**PISEL2.ISCNT12** = 00_B), the input clock f_{T12} of Timer T12 is derived from the internal module clock f_{CC6} through a programmable prescaler and an optional 1/256 divider. The resulting prescaler factors are listed in **Table 27-2**. The prescaler of T12 is cleared while T12 is not running (**TCTR0.T12R** = 0) to ensure reproducible timings and delays.

Table 27-2 Timer T12 Input Frequency Options

T12CLK	Resulting Input Clock f_{T12} Prescaler Off (T12PRE = 0)	Resulting Input Clock f_{T12} Prescaler On (T12PRE = 1)
000 _B	f_{CC6}	$f_{CC6} / 256$
001 _B	$f_{CC6} / 2$	$f_{CC6} / 512$
010 _B	$f_{CC6} / 4$	$f_{CC6} / 1024$
011 _B	$f_{CC6} / 8$	$f_{CC6} / 2048$
100 _B	$f_{CC6} / 16$	$f_{CC6} / 4096$
101 _B	$f_{CC6} / 32$	$f_{CC6} / 8192$
110 _B	$f_{CC6} / 64$	$f_{CC6} / 16384$
111 _B	$f_{CC6} / 128$	$f_{CC6} / 32768$

In **Counter Mode**, timer T12 counts one step:

- If a 1 is written to **TCTR4.T12CNT** and **PISEL2.ISCNT12** = 01_B
- If a rising edge of input signal T12HR is detected and **PISEL2.ISCNT12** = 10_B
- If a falling edge of input signal T12HR is detected and **PISEL2.ISCNT12** = 11_B

27.2.2.2 Edge-Aligned / Center-Aligned Mode

In **Edge-Aligned Mode** (CTM = 0), timer T12 is always counting upwards (CDIR = 0). When reaching the value given by the period register (period-match T12_PM), the value of T12 is cleared with the next counting step (saw tooth shape).

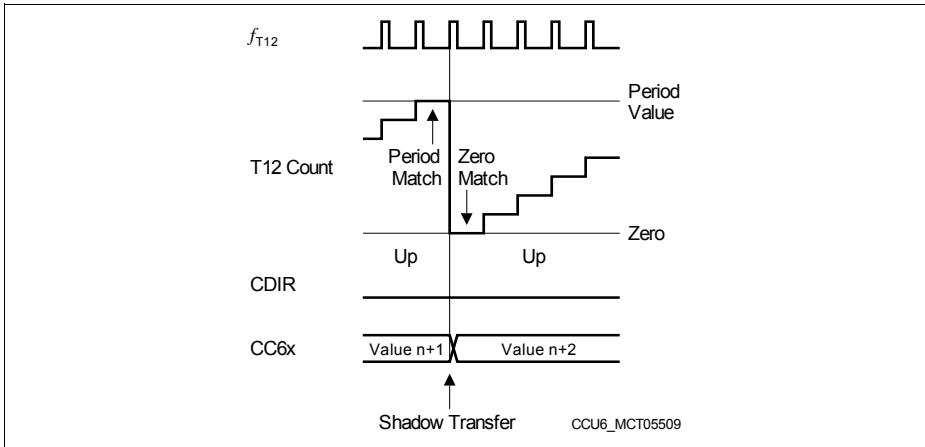


Figure 27-5 T12 Operation in Edge-Aligned Mode

As a result, in Edge-Aligned mode, the timer period is given by:

$$T12_{PER} = \langle \text{Period-Value} \rangle + 1; \text{ in } T12 \text{ clocks } (f_{T12}) \quad (27.1)$$

In **Center-Aligned Mode** (CTM = 1), timer T12 is counting upwards or downwards (triangular shape). When reaching the value given by the period register (period-match T12_PM) while counting upwards (CDIR = 0), the counting direction control bit CDIR is changed to downwards (CDIR = 1) with the next counting step.

When reaching the value 0001_H (one-match T12_OM) while counting downwards, the counting direction control bit CDIR is changed to upwards with the next counting step.

As a result, in Center-Aligned mode, the timer period is given by:

$$T12_{PER} = (\langle \text{Period-Value} \rangle + 1) \times 2; \text{ in } T12 \text{ clocks } (f_{T12}) \quad (27.2)$$

- With the next clock event of f_{T12} the count direction is set to counting up (CDIR = 0) when the counter reaches 0001_H while counting down.
- With the next clock event of f_{T12} the count direction is set to counting down (CDIR = 1) when the Period-Match is detected while counting up.
- With the next clock event of f_{T12} the counter counts up while CDIR = 0 and it counts down while CDIR = 1.

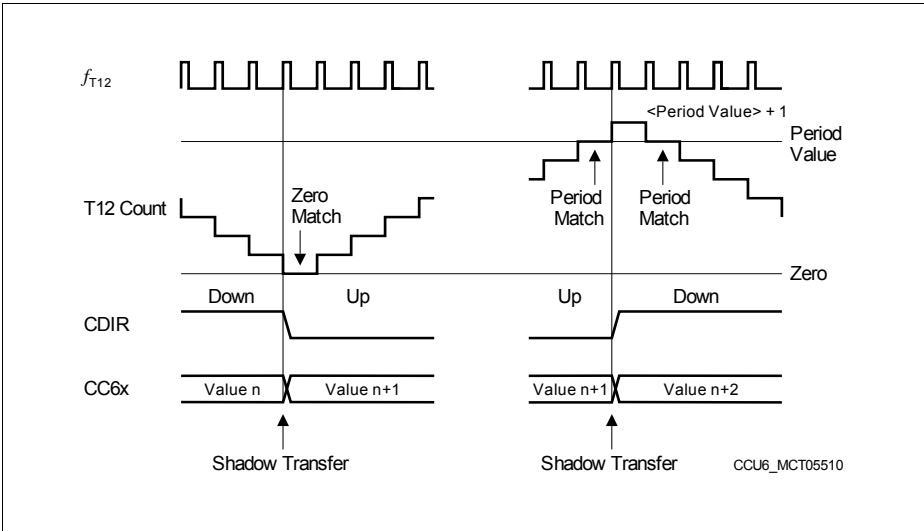


Figure 27-6 T12 Operation in Center-Aligned Mode

Note: Bit CDIR changes with the next timer clock event after the one-match or the period-match. Therefore, the timer continues counting in the previous direction for one cycle before actually changing its direction (see [Figure 27-6](#)).

27.2.2.3 Single-Shot Mode

In Single-Shot Mode, the timer run bit T12R is cleared by hardware. If bit T12SSC = 1, the timer T12 will stop when the current timer period is finished.

In Edge-Aligned mode, T12R is cleared when the timer becomes zero after having reached the period value (see [Figure 27-7](#)).

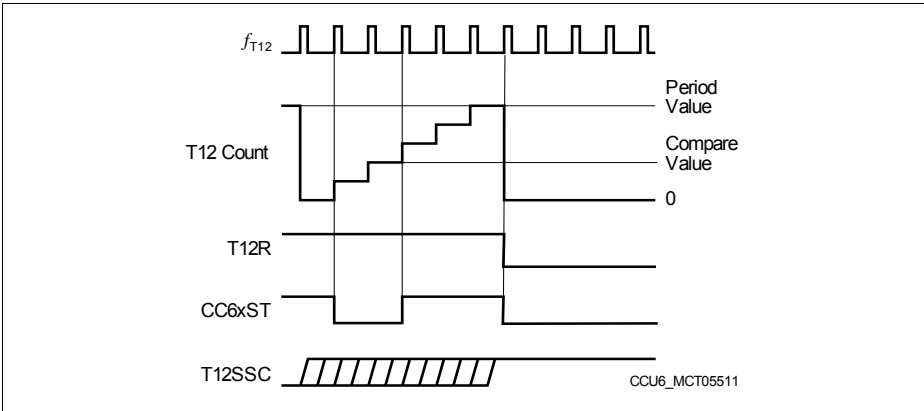


Figure 27-7 Single-Shot Operation in Edge-Aligned Mode

In Center-Aligned mode, the period is finished when the timer has counted down to zero (one clock cycle after the one-match while counting down, see [Figure 27-8](#)).

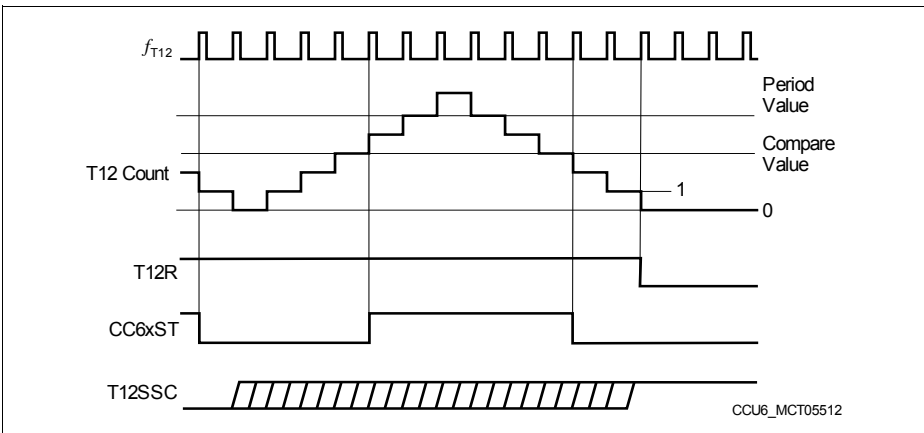


Figure 27-8 Single-Shot Operation in Center-Aligned Mode

27.2.3 T12 Compare Mode

Associated with Timer T12 are three individual capture/compare channels, that can perform compare or capture operations with regard to the contents of the T12 counter. The capture functions are explained in [Section 27.2.5](#).

27.2.3.1 Compare Channels

In Compare Mode (see [Figure 27-9](#)), the three individual compare channels CC60, CC61, and CC62 can generate a three-phase PWM pattern.

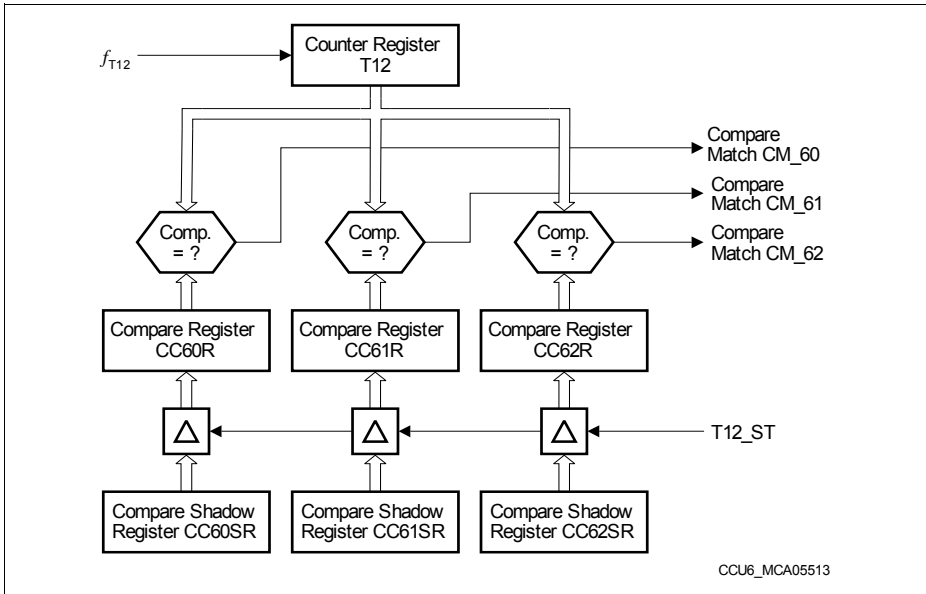


Figure 27-9 T12 Channel Comparators

Each compare channel is connected to the T12 counter register via its individual equal-to comparator, generating a match signal when the contents of the counter matches the contents of the associated compare register. Each channel consists of the comparator and a double register structure - the actual compare register CC6xR, feeding the comparator, and an associated shadow register CC6xSR, that is preloaded by software and transferred into the compare register when signal T12 shadow transfer, T12_ST, gets active. Providing a shadow register for the compare value as well as for other values related to the generation of the PWM signal facilitates a concurrent update by software for all relevant parameters of a three-phase PWM.

27.2.3.2 Channel State Bits

Associated with each (compare) channel is a State Bit, **CMPSTAT.CC6xST**, holding the status of the compare (or capture) operation (see **Figure 27-10**). In compare mode, the State Bits are modified according to a set of switching rules, depending on the current status of timer T12.

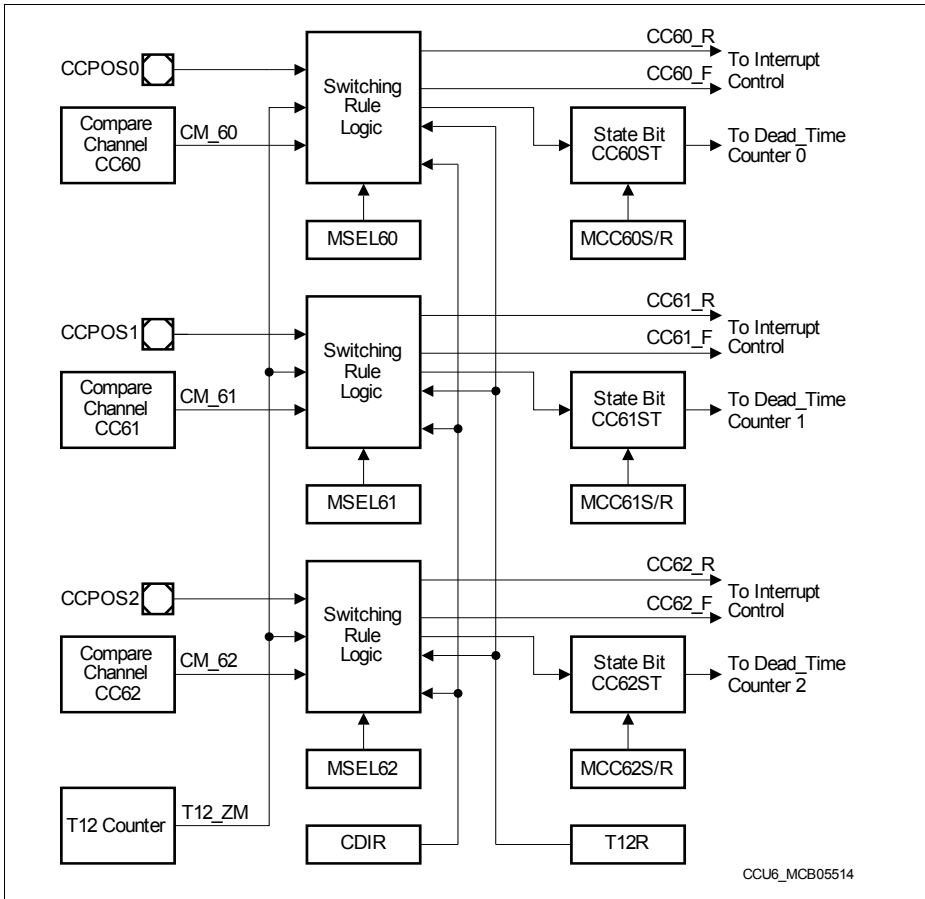


Figure 27-10 Compare State Bits for Compare Mode

The inputs to the switching rule logic for the CC6xST bits are the timer direction (CDIR), the timer run bit (T12R), the timer T12 zero-match signal (T12_ZM), and the actual individual compare-match signals CM_6x as well as the mode control bits, **T12MSEL.MSEL6x**.

Capture/Compare Unit 6 (CCU6)

In addition, each state bit can be set or cleared by software via the appropriate set and reset bits in register **CMPMODIF**, MCC6xS and MCC6xR. The input signals CCPOSx are used in hysteresis-like compare mode, whereas in normal compare mode, these inputs are ignored.

Note: In Hall Sensor, single shot or capture modes, additional/different rules are taken into account (see related sections).

A compare interrupt event CC6x_R is signaled when a compare match is detected while counting upwards, whereas the compare interrupt event CC6x_F is signaled when a compare match is detected while counting down. The actual setting of a State Bit has no influence on the interrupt generation in compare mode.

A modification of a State Bit CC6xST by the switching rule logic due to a compare action is only possible while Timer T12 is running (T12R = 1). If this is the case, the following switching rules apply for setting and clearing the State Bits in Compare Mode (illustrated in **Figure 27-11** and **Figure 27-12**):

A State Bit **CC6xST** is set to 1:

- with the next T12 clock (f_{T12}) after a compare-match when T12 is counting up (i.e., when the counter is incremented above the compare value);
- with the next T12 clock (f_{T12}) after a zero-match AND a parallel compare-match when T12 is counting up.

A State Bit **CC6xST** is cleared to 0:

- with the next T12 clock (f_{T12}) after a compare-match when T12 is counting down (i.e., when the counter is decremented below the compare value in center-aligned mode);
- with the next T12 clock (f_{T12}) after a zero-match AND NO parallel compare-match when T12 is counting up.

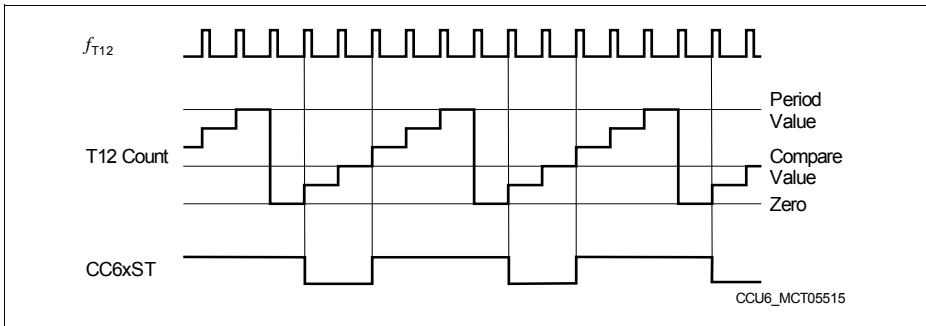


Figure 27-11 Compare Operation, Edge-Aligned Mode

Figure 27-13 illustrates some more examples for compare waveforms. It is important to note that in these examples, it is assumed that some of the compare values are changed

Capture/Compare Unit 6 (CCU6)

while the timer is running. This change is performed via a software preload of the Shadow Register, CC6xSR. The value is transferred to the actual Compare Register CC6xR with the T12 Shadow Transfer signal, T12_ST, that is assumed to be enabled.

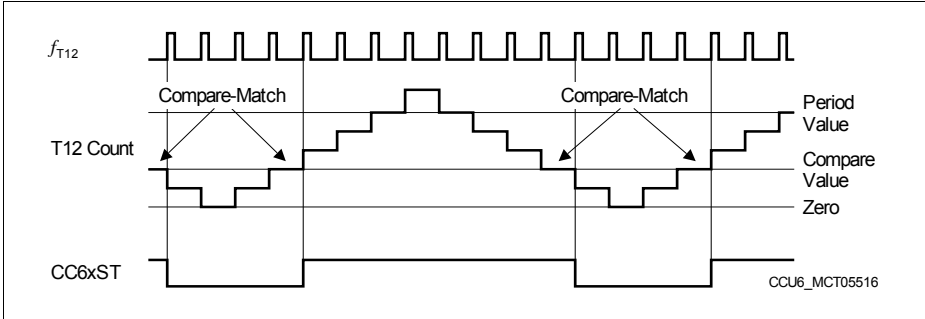


Figure 27-12 Compare Operation, Center-Aligned Mode

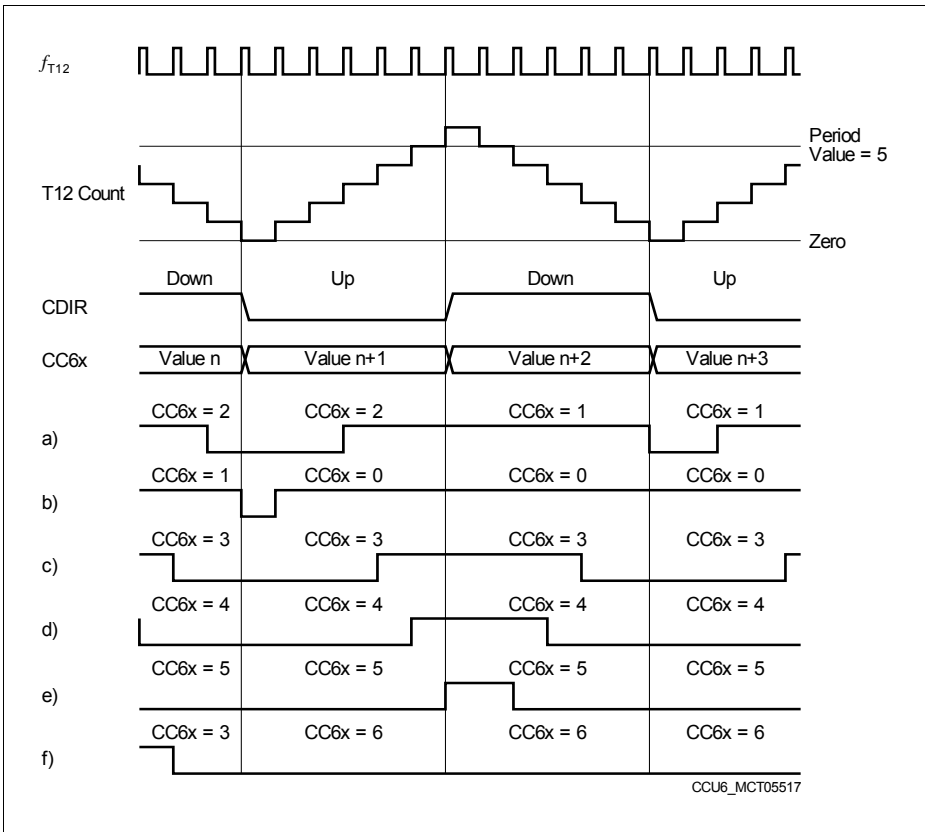


Figure 27-13 Compare Waveform Examples

Example b) illustrates the transition to a duty cycle of 100%. First, a compare value of 0001_H is used, then changed to 0000_H . Please note that a low pulse with the length of one T12 clock is still produced in the cycle where the new value 0000_H is in effect; this pulse originates from the previous value 0001_H . In the following timer cycles, the State Bit CC6xST remains at 1, producing a 100% duty cycle signal. In this case, the compare rule 'zero-match AND compare-match' is in effect.

Example f) shows the transition to a duty cycle of 0%. The new compare value is set to $\langle \text{Period-Value} \rangle + 1$, and the State Bit CC6ST remains cleared.

Figure 27-14 illustrates an example for the waveforms of all three channels. With the appropriate dead-time control and output modulation, a very efficient 3-phase PWM signal can be generated.

Capture/Compare Unit 6 (CCU6)

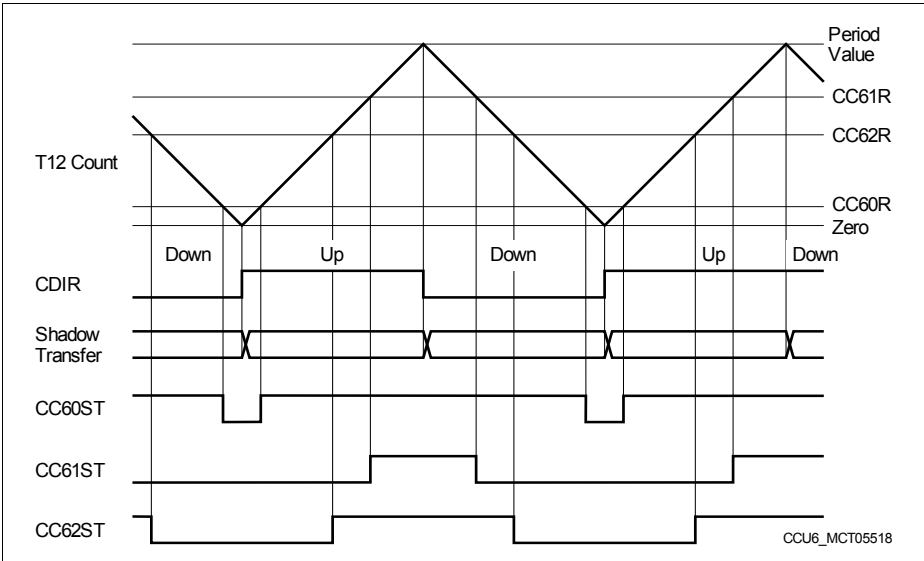


Figure 27-14 Three-Channel Compare Waveforms

27.2.3.3 Hysteresis-Like Control Mode

The hysteresis-like control mode (**T12MSEL.MSEL6x = 1001_B**) offers the possibility to switch off the PWM output if the input CCPOSx becomes 0 by clearing the State Bit CC6xST. This can be used as a simple motor control feature by using a comparator indicating, e.g., overcurrent. While CCPOSx = 0, the PWM outputs of the corresponding channel are driving their passive levels, because the setting of bit CC6xST is only possible while CCPOSx = 1.

As long as input CCPOSx is 0, the corresponding State Bit is held 0. When CCPOSx is at high level, the outputs can be in active state and are determined by bit CC6xST (see **Figure 27-10** for the state bit logic and **Figure 27-15** for the output paths).

The CCPOSx inputs are evaluated with f_{CC6} .

This mode can be used to introduce a timing-related behavior to a hysteresis controller. A standard hysteresis controller detects if a value exceeds a limit and switches its output according to the compare result. Depending on the operating conditions, the switching frequency and the duty cycle are not fixed, but change permanently.

If (outer) time-related control loops based on a hysteresis controller in an inner loop should be implemented, the outer loops show a better behavior if they are synchronized to the inner loops. Therefore, the hysteresis-like mode can be used, that combines timer-related switching with a hysteresis controller behavior. For example, in this mode, an output can be switched on according to a fixed time base, but it is switched off as soon as a falling edge is detected at input CCPOSx.

This mode can also be used for standard PWM with overcurrent protection. As long as there is no low level signal at pin CCPOSx, the output signals are generated in the normal manner as described in the previous sections. Only if input CCPOSx shows a low level, e.g. due to the detection of overcurrent, the outputs are shut off to avoid harmful stress to the system.

27.2.4 Compare Mode Output Path

Figure 27-15 gives an overview on the signal path from a channel State Bit to its output pin in its simplest form. As illustrated, a user has a variety of controls to determine the desired output signal switching behavior in relation to the current state of the State Bit, CC6xST. Please refer to **Section 27.2.4.3** for details on the output modulation.

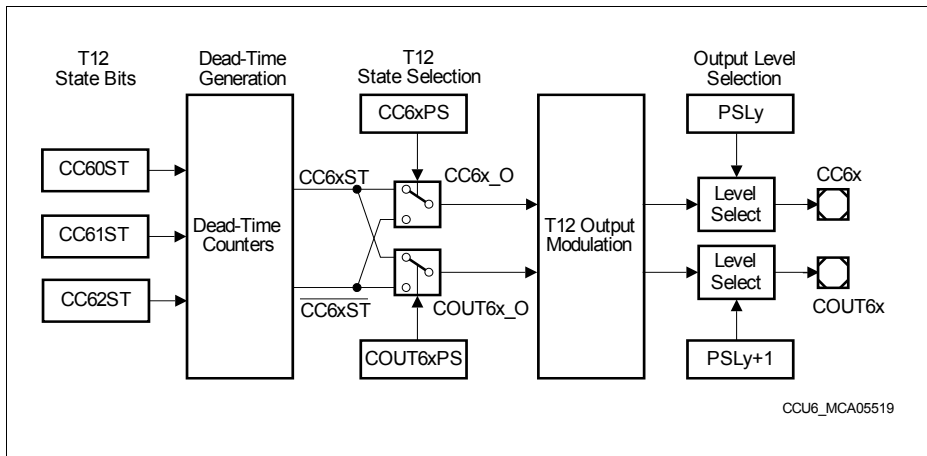


Figure 27-15 Compare Mode Simplified Output Path Diagram

The output path is based on signals that are defined as active or passive. The terms active and passive are not related to output levels, but to internal actions. This mainly applies for the modulation, where T12 and T13 signals are combined with the multi-channel signals and the trap function. The Output level Selection allows the user to define the output level at the output pin for the passive state (inverted level for the active state). It is recommended to configure this block in a way that an external power switch is switched off while the CCU6 delivers an output signal in the passive state.

27.2.4.1 Dead-Time Generation

The generation of (complementary) signals for the high-side and the low-side switches of one power inverter phase is based on the same compare channel. For example, if the high-side switch should be active while the T12 counter value is above the compare value (State Bit = 1), then the low-side switch should be active while the counter value is below the compare value (State Bit = 0).

In most cases, the switching behavior of the connected power switches is not symmetrical concerning the switch-on and switch-off times. A general problem arises if the time for switch-on is smaller than the time for switch-off of the power device. In this case, a short-circuit can occur in the inverter bridge leg, which may damage the complete system. In order to solve this problem by HW, this capture/compare unit

Capture/Compare Unit 6 (CCU6)

contains a programmable Dead-Time Generation Block, that delays the passive to active edge of the switching signals by a programmable time (the active to passive edge is not delayed).

The Dead-Time Generation Block, illustrated in **Figure 27-16**, is built in a similar way for all three channels of T12. It is controlled by bits in register **T12DTC**. Any change of a CC6xST State Bit activates the corresponding Dead-Time Counter, that is clocked with the same input clock as T12 (f_{T12}). The length of the dead-time can be programmed by bit field DTM. This value is identical for all three channels. Writing **TCTR4.DTRES = 1** sets all dead-times to passive.

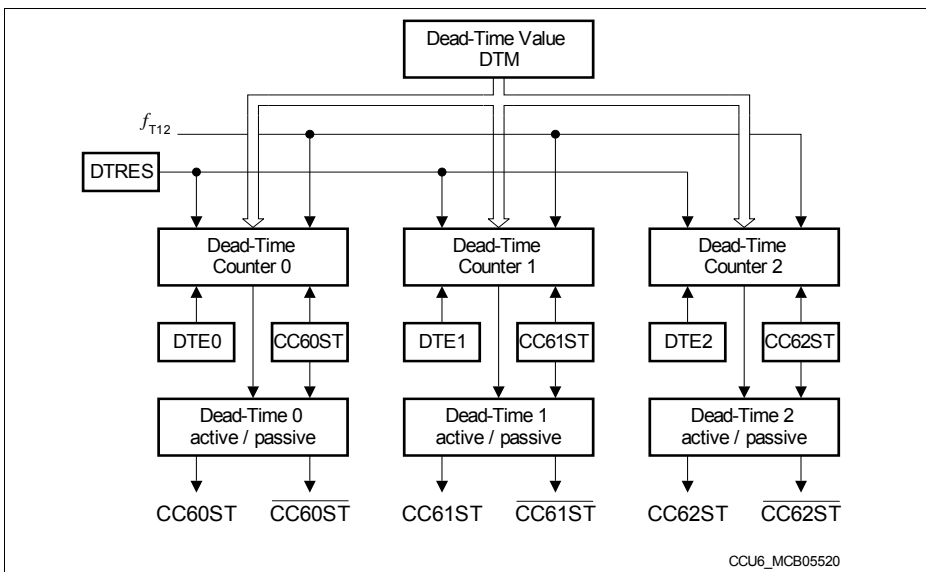


Figure 27-16 Dead-Time Generation Block Diagram

Each of the three dead-time counters has its individual dead-time enable bit, DTE_x. An enabled dead-time counter generates a dead-time delaying the passive-to-active edge of the channel output signal. The change in a State Bit CC6_xST is not taken into account while the dead-time generation of this channel is currently in progress (active). This avoids an unintentional additional dead-time if a State Bit CC6_xST changes too early. A disabled dead-time counter is always considered as passive and does not delay any edge of CC6_xST.

Based on the State Bits CC6_xST, the Dead-Time Generation Block outputs a direct signal CC6_xST and an inverted signal $\overline{\text{CC6xST}}$ for each compare channel, each masked with the effect of the related Dead-Time Counters (waveforms illustrated in **Figure 27-17**).

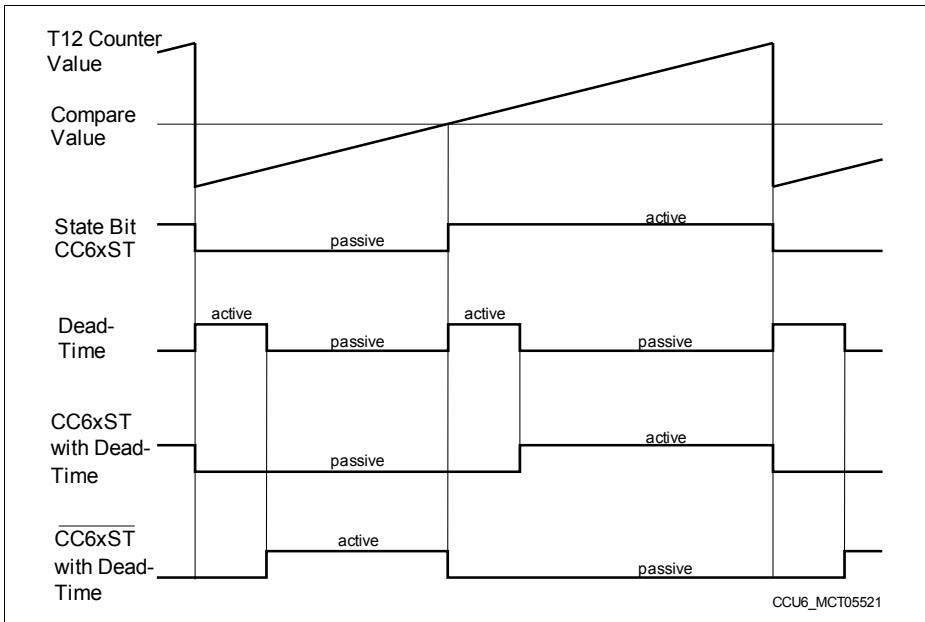


Figure 27-17 Dead-Time Generation Waveforms

27.2.4.2 State Selection

To support a wide range of power switches and drivers, the state selection offers the flexibility to define when an output can be active and can be modulated, especially useful for **complementary or multi-phase PWM** signals.

The state selection is based on the signals $\overline{\text{CC6xST}}$ and $\overline{\text{CC6xST}}$ delivered by the dead-time generator (see [Figure 27-15](#)). Both signals are never active at the same time, but can be passive at the same time. This happens during the dead-time of each compare channel after a change of the corresponding State Bit CC6xST .

The user can select independently for each output signal CC6xO and COUT6xO if it should be active before or after the compare value has been reached (see register [CMPSTAT](#)). With this selection, the active (conducting) phases of complementary power switches in a power inverter bridge leg can be positioned with respect to the compare value (e.g. signal CC6xO can be active before, whereas COUT6xO can be active after the compare value is reached). Like this, the output modulation, the trap logic and the output level selection can be programmed independently for each output signal, although two output signals are referring to the same compare channel.

27.2.4.3 Output Modulation and Level Selection

The last block of the data path is the Output Modulation block. Here, all the modulation sources and the trap functionality are combined and control the actual level of the output pins (controlled by the modulation enable bits T1xMODENy and MCMEN in register **MODCTR**). The following signal sources can be combined here **for each T12 output signal** (see **Figure 27-18** for compare channel CC60):

- A **T12 related compare signal** CC6x_O (for outputs CC6x) or COUT6x_O (for outputs COUT6x) delivered by the T12 block (state selection with dead-time) with an individual enable bit T12MODENy per output signal (y = 0, 2, 4 for outputs CC6x and y = 1, 3, 5 for outputs COUT6x)
- The **T13 related compare signal** CC63_O delivered by the T13 state selection with an individual enable bit T13MODENy per output signal (y = 0, 2, 4 for outputs CC6x and y = 1, 3, 5 for outputs COUT6x)
- A **multi-channel output signal** MCMPy (y = 0, 2, 4 for outputs CC6x and y = 1, 3, 5 for outputs COUT6x) with a common enable bit MCMEN
- The **trap state** TRPS with an individual enable bit TRPENy per output signal (y = 0, 2, 4 for outputs CC6x and y = 1, 3, 5 for outputs COUT6x)

If one of the modulation input signals CC6x_O/COUT6x_O, CC63_O, or MCMPy of an output modulation block is enabled and is at passive state, the modulated is also in passive state, regardless of the state of the other signals that are enabled. Only if all enabled signals are in active state the modulated output shows an active state. If no modulation input is enabled, the output is in passive state.

If the Trap State is active (TRPS = 1), then the outputs that are enabled for the trap signal (by TRPENy = 1) are set to the passive state.

The output of each of the modulation control blocks is connected to a level select block that is configured by register **PSLR**. It offers the option to determine the actual output level of a pin, depending on the state of the output line (decoupling of active/passive state and output polarity) as specified by the Passive State Select bit PSLy. If the modulated output signal is in the passive state, the level specified directly by PSLy is output. If it is in the active state, the inverted level of PSLy is output. This allows the user to adapt the polarity of an active output signal to the connected circuitry.

The PSLy bits have shadow registers to allow for updates without undesired pulses on the output lines. The bits related to CC6x and COUT6x (x = 0, 1, 2) are updated with the T12 shadow transfer signal (T12_ST). A read action returns the actually used values, whereas a write action targets the shadow bits. Providing a shadow register for the PSL value as well as for other values related to the generation of the PWM signal facilitates a concurrent update by software for all relevant parameters.

Figure 27-18 shows the output modulation structure for compare channel CC60 (output signals CC60 and COUT60). A similar structure is implemented for the other two compare channels CC61 and CC62.

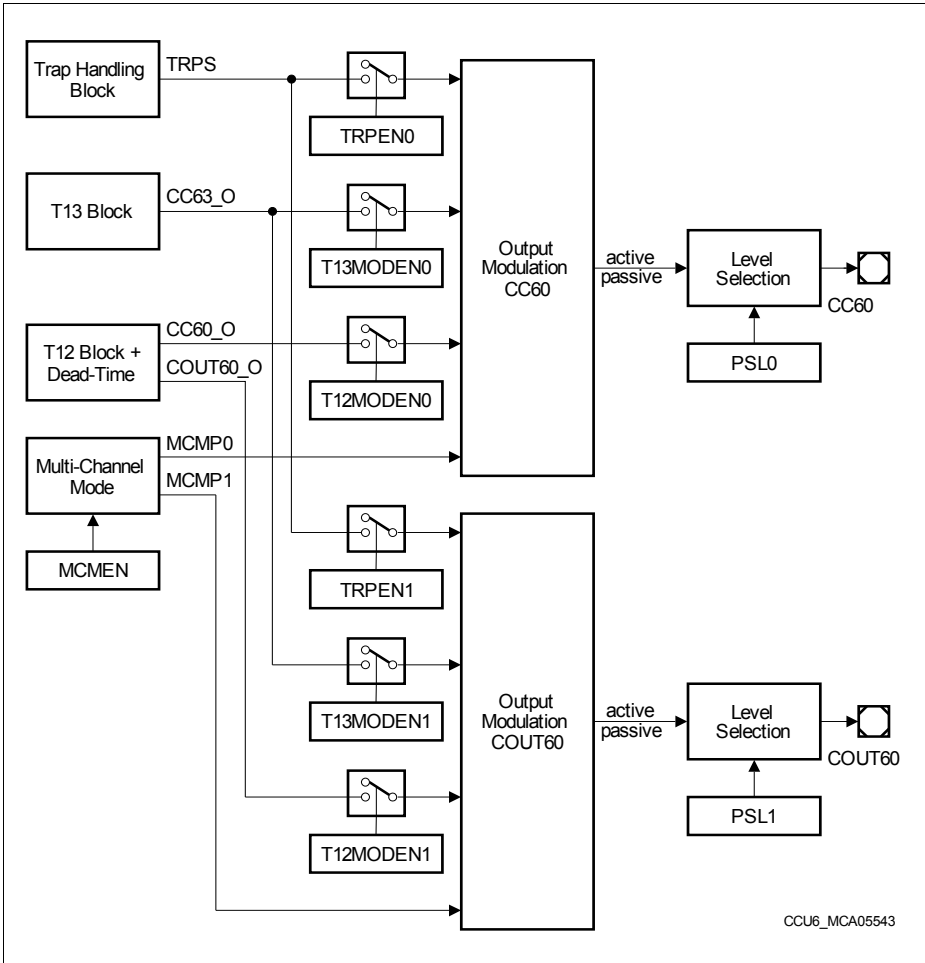


Figure 27-18 Output Modulation for Compare Channel CC60

27.2.5 T12 Capture Modes

Each of the three channels of the T12 Block can also be used to capture T12 time information in response to an external signal CC6xIN.

In capture mode, the interrupt event CC6x_R is detected when a rising edge is detected at the input CC6xIN, whereas the interrupt event CC6x_F is detected when a falling edge is detected.

There are a number of different modes for capture operation. In all modes, both of the registers of a channel are used. The selection of the capture modes is done via the **T12MSEL**.MSEL6x bit fields and can be selected individually for each of the channels.

Table 27-3 Capture Modes Overview

MSEL6x	Mode	Signal	Active Edge	CC6nSR Stored in	T12 Stored in
0100 _B	1	CC6xIN	Rising	–	CC6xR
		CC6xIN	Falling	–	CC6xSR
0101 _B	2	CC6xIN	Rising	CC6xR	CC6xSR
0110 _B	3	CC6xIN	Falling	CC6xR	CC6xSR
0111 _B	4	CC6xIN	Any	CC6xR	CC6xSR

Figure 27-19 illustrates **Capture Mode 1**. When a rising edge (0-to-1 transition) is detected at the corresponding input signal CC6xIN, the current contents of Timer T12 are captured into register CC6xR. When a falling edge (1-to-0 transition) is detected at the input signal CC6xIN, the contents of Timer T12 are captured into register CC6xSR.

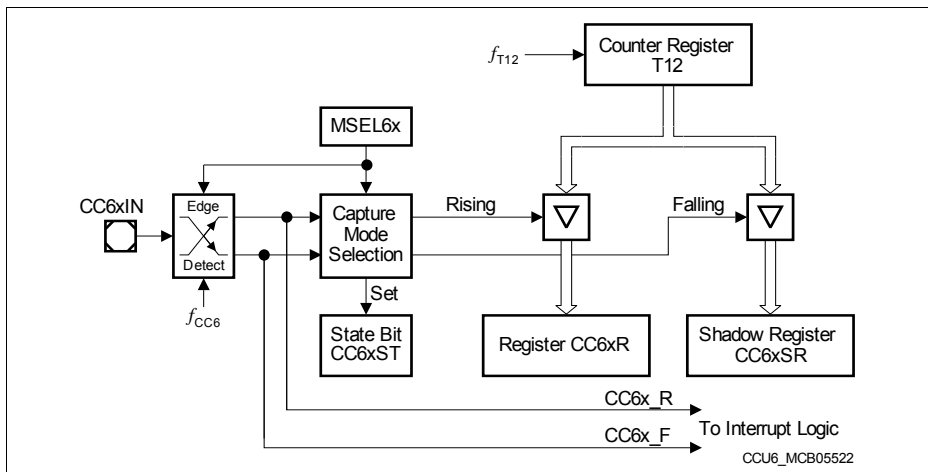


Figure 27-19 Capture Mode 1 Block Diagram

Capture/Compare Unit 6 (CCU6)

Capture Modes 2, 3 and 4 are shown in **Figure 27-20**. They differ only in the active edge causing the capture operation. In each of the three modes, when the selected edge is detected at the corresponding input signal CC6xIN, the current contents of the shadow register CC6xSR are transferred into register CC6xR, and the current Timer T12 contents are captured in register CC6xSR (simultaneous transfer). The active edge is a rising edge of CC6xIN for Capture Mode 2, a falling edge for Mode 3, and both, a rising or a falling edge for Capture Mode 4, as shown in **Table 27-3**. These capture modes are very useful in cases where there is little time between two consecutive edges of the input signal.

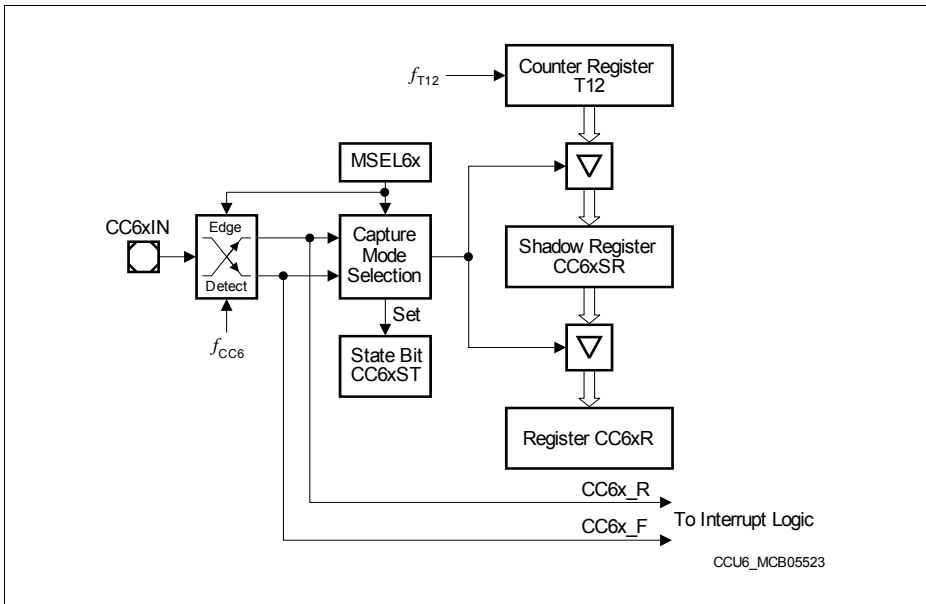


Figure 27-20 Capture Modes 2, 3 and 4 Block Diagram

Capture/Compare Unit 6 (CCU6)

Five further capture modes are called **Multi-Input Capture Modes**, as they use two different external inputs, signal CC6xIN and signal CCPOSx.

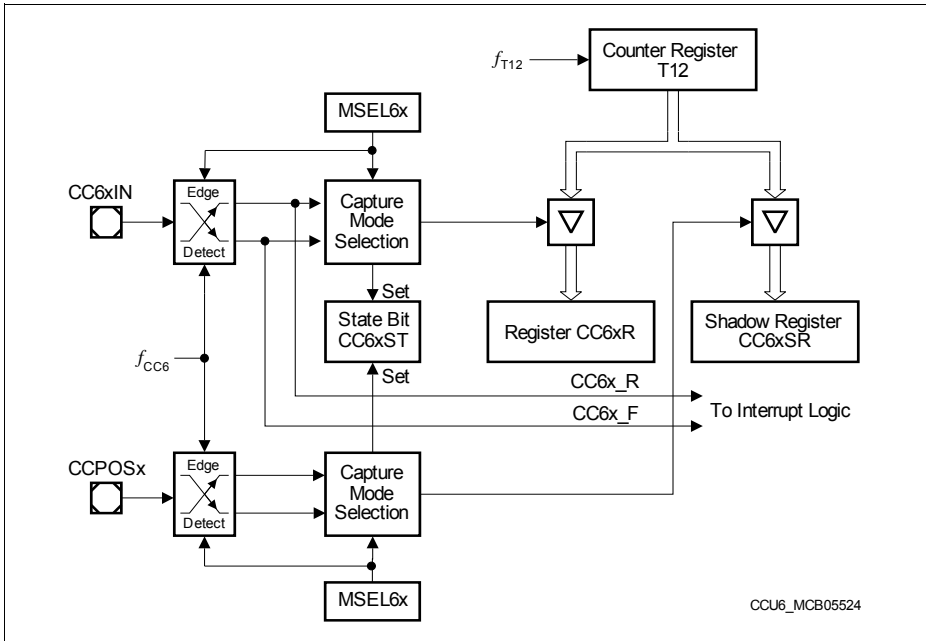


Figure 27-21 Multi-Input Capture Modes Block Diagram

In each of these modes, the current T12 contents are captured in register CC6xR in response to a selected event at signal CC6xIN, and in register CC6xSR in response to a selected event at signal CCPOSx. The possible events can be opposite input transitions, or the same transitions, or any transition at the two inputs. The different options are detailed in [Table 27-4](#).

In each of the various capture modes, the Channel State Bit, CC6xST, is set to 1 when the selected capture trigger event at signal CC6xIN or CCPOSx has occurred. The State Bit is not cleared by hardware, but can be cleared by software.

In addition, appropriate signal lines to the interrupt logic are activated, that can generate an interrupt request to the CPU. Regardless of the selected active edge, all edges detected at signal CC6xIN can lead to the activation of the appropriate interrupt request line (see also [Section 27.9](#)).

Table 27-4 Multi-Input Capture Modes Overview

MSEL6x	Mode	Signal	Active Edge	T12 Stored in
1010 _B	5	CC6xIN	Rising	CC6xR
		CCPOSx	Falling	CC6xSR
1011 _B	6	CC6xIN	Falling	CC6xR
		CCPOSx	Rising	CC6xSR
1100 _B	7	CC6xIN	Rising	CC6xR
		CCPOSx	Rising	CC6xSR
1101 _B	8	CC6xIN	Falling	CC6xR
		CCPOSx	Falling	CC6xSR
1110 _B	9	CC6xIN	Any	CC6xR
		CCPOSx	Any	CC6xSR
1111 _B	–	reserved (no capture or compare action)		

27.2.6 T12 Shadow Register Transfer

A special shadow transfer signal (T12_ST) can be generated to facilitate updating the period and compare values of the compare channels CC60, CC61, and CC62 synchronously to the operation of T12. Providing a shadow register for values defining one PWM period facilitates a concurrent update by software for all relevant parameters. The next PWM period can run with a new set of parameters. The generation of this signal is requested by software via bit **TCTR0.STE12** (set by writing 1 to the write-only bit **TCTR4.T12STR**, cleared by writing 1 to the write-only bit **TCTR4.T12STD**).

Figure 27-22 shows the shadow register structure and the shadow transfer signals, as well as on the read/write accessibility of the various registers.

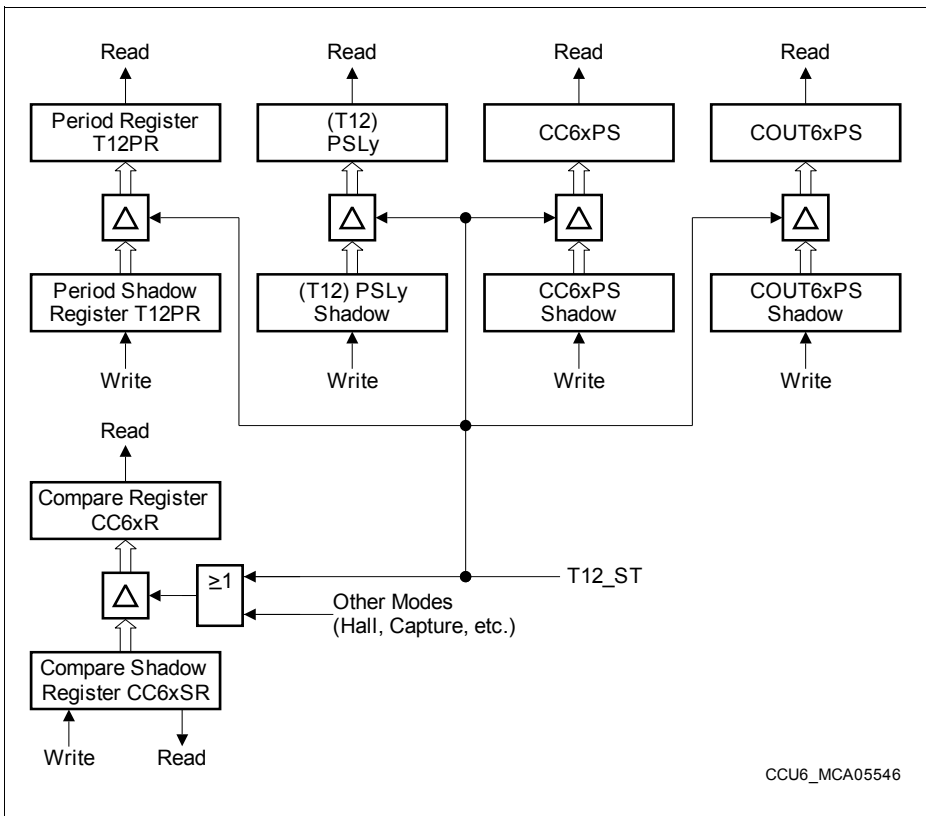


Figure 27-22 T12 Shadow Register Overview

Capture/Compare Unit 6 (CCU6)

A T12 shadow register transfer takes place (T12_ST active):

- while timer T12 is not running (T12R = 0), or
- STE12 = 1 and a Period-Match is detected while counting up, or
- STE12 = 1 and a One-Match is detected while counting down

When signal T12_ST is active, a shadow register transfer is triggered with the next cycle of the T12 clock. Bit STE12 is automatically cleared with the shadow register transfer.

27.2.7 Timer T12 Operating Mode Selection

The operating mode for the T12 channels are defined by the bit fields **T12MSEL.MSEL6x**.

Table 27-5 T12 Capture/Compare Modes Overview

MSEL6x	Selected Operating Mode
0000 _B , 1111 _B	Capture/Compare modes switched off
0001 _B , 0010 _B , 0011 _B	Compare mode, see Section 27.2.3 same behavior for all three codings
01XX _B	Double-Register Capture modes, see Section 27.2.5
1000 _B	Hall Sensor Mode, see Section 27.7 In order to properly enable this mode, all three MSEL6x fields have to be programmed to Hall Sensor mode.
1001 _B	Hysteresis-like compare mode, see Section 27.2.3.3
1010 _B , 1011 _B , 1100 _B , 1101 _B , 1110 _B	Multi-Input Capture modes, see Section 27.2.5

The clocking and counting scheme of the timers are controlled by the timer control registers **TCTR0** and **TCTR2**. Specific actions are triggered by write operations to register **TCTR4**.

27.2.8 T12 related Registers

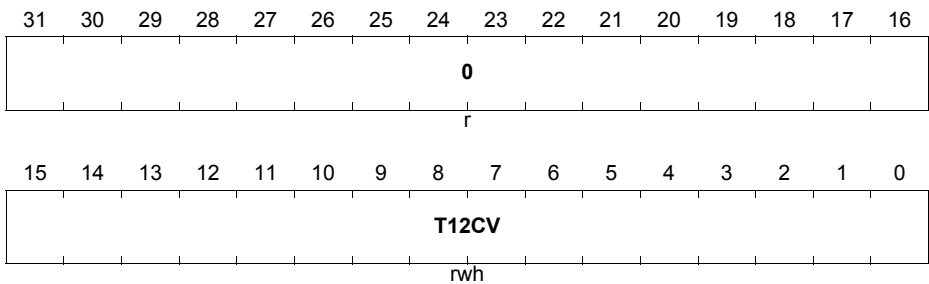
27.2.8.1 T12 Counter Register

Register T12 represents the counting value of timer T12. It can only be written while the timer T12 is stopped. Write actions while T12 is running are not taken into account. Register T12 can always be read by SW.

In edge-aligned mode, T12 only counts up, whereas in center-aligned mode, T12 can count up and down.

T12

Timer T12 Counter Register (20_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
T12CV	[15:0]	rwh	Timer 12 Counter Value This register represents the 16-bit counter value of Timer12.
0	[31:16]	r	Reserved; Returns 0 if read; should be written with 0.

Note: While timer T12 is stopped, the internal clock divider is reset in order to ensure reproducible timings and delays.

27.2.8.2 Period Register

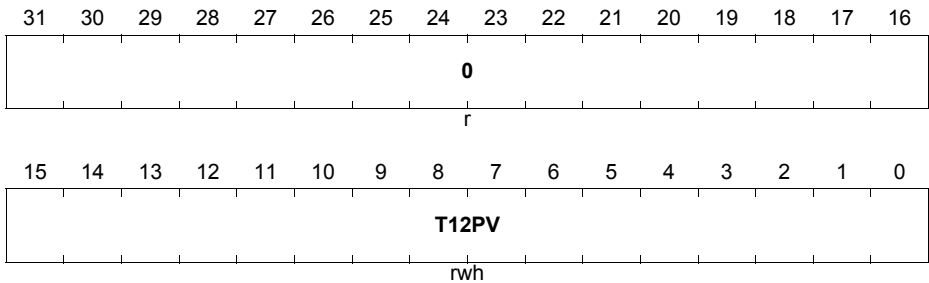
Register T12PR contains the period value for timer T12. The period value is compared to the actual counter value of T12 and the resulting counter actions depend on the defined counting rules. This register has a shadow register and the shadow transfer is controlled by bit STE12. A read action by SW delivers the value that is currently used for the compare action, whereas the write action targets a shadow register. The shadow register structure allows a concurrent update of all T12-related values.

T12PR

Timer 12 Period Register

(24_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
T12PV	[15:0]	rwh	T12 Period Value The value T12PV defines the counter value for T12 leading to a period-match. When reaching this value, the timer T12 is set to zero (edge-aligned mode) or changes its count direction to down counting (center-aligned mode).
0	[31:16]	r	Reserved; Returns 0 if read; should be written with 0.

Capture/Compare Unit 6 (CCU6)

27.2.8.3 Capture/Compare Registers

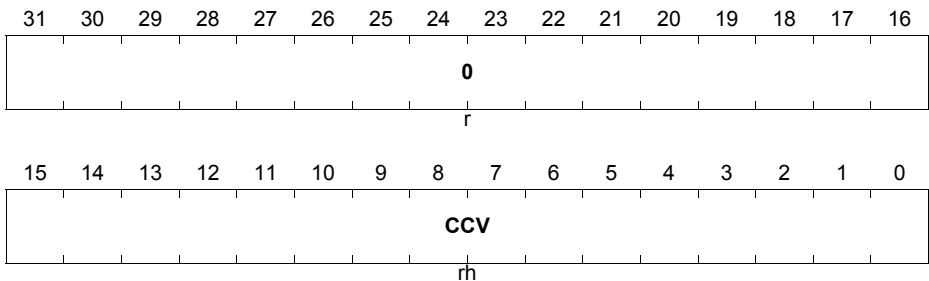
In compare mode, the registers CC6xR (x = 0 - 2) are the actual compare registers for T12. The values stored in CC6xR are compared (all three channels in parallel) to the counter value of T12. In capture mode, the current value of the T12 counter register is captured by registers CC6xR if the corresponding capture event is detected.

CC6xR (x = 0-2)

Capture/Compare Register for Channel CC6x

$$(30_H + 4 * x)$$

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CCV	[15:0]	rh	Capture/Compare Value In compare mode, the bit fields CCV contain the values, that are compared to the T12 counter value. In capture mode, the captured value of T12 can be read from these registers.
0	[31:16]	r	Reserved; Returns 0 if read; should be written with 0.

Capture/Compare Unit 6 (CCU6)

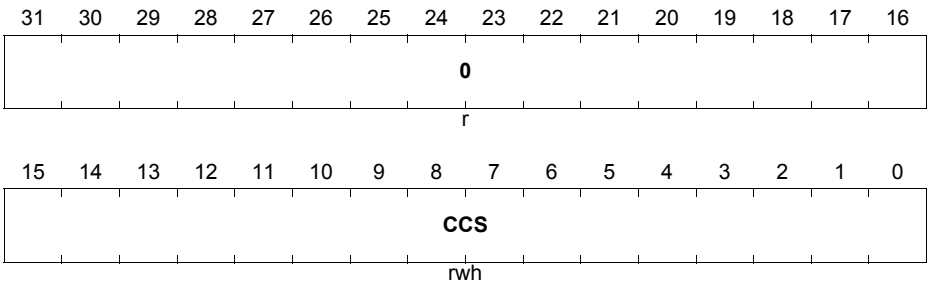
27.2.8.4 Capture/Compare Shadow Registers

The registers CC6xR can only be read by SW, the modification of the value is done by a shadow register transfer from register CC6xSR. The corresponding shadow registers CC6xSR can be read and written by SW. In capture mode, the value of the T12 counter register can also be captured by registers CC6xSR if the selected capture event is detected (depending on the selected capture mode).

CC6xSR (x=0-2)

Capture/Compare Shadow Reg. for Channel CC6x
(40_H+4*x)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CCS	[15:0]	rwh	Shadow Register for Channel x Capture/Compare Value In compare mode, the bit fields contents of CCS are transferred to the bit fields CCV for the corresponding channel during a shadow transfer. In capture mode, the captured value of T12 can be read from these registers.
0	[31:16]	r	Reserved; Returns 0 if read; should be written with 0.

Note: The shadow registers can also be written by SW in capture mode. In this case, the HW capture event wins over the SW write if both happen in the same cycle (the SW write is discarded).

27.2.8.5 Dead-time Control Register

Register T12DTC controls the dead-time generation for the timer T12 compare channels. Each channel can be independently enabled/disabled for dead-time generation. If enabled, the transition from passive state to active state is delayed by the value defined by bit field DTM.

The dead time counters are clocked with the same frequency as T12.

This structure allows symmetrical dead-time generation in center-aligned and in edge-aligned PWM mode. A duty cycle of 50% leads to CC6x, COUT6x switched on for: 0.5 * period - dead time.

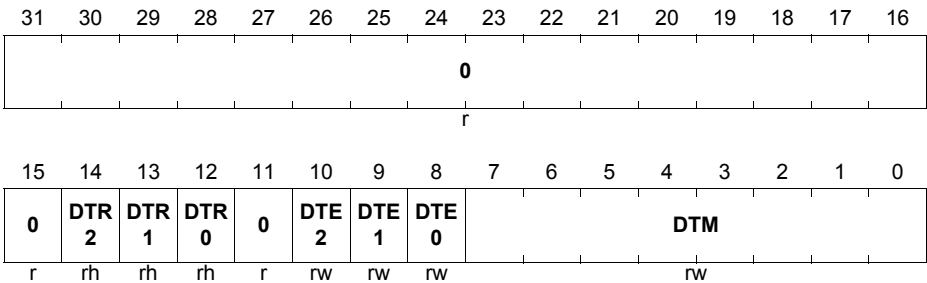
Note: The dead-time counters are not reset by bit T12RES, but by bit DTRES.

T12DTC

Dead-Time Control Register for Timer12

(28_H)

Reset Value: 0000 0000_H



Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
DTM	[7:0]	rw	Dead-Time Bit field DTM determines the programmable delay between switching from the passive state to the active state of the selected outputs. The switching from the active state to the passive state is not delayed.
DTE2, DTE1, DTE0	10, 9, 8	rw	Dead Time Enable Bits Bits DTE0..DTE2 enable and disable the dead time generation for each compare channel (0, 1, 2) of timer T12. 0_B Dead-Time Counter x is disabled. The corresponding outputs switch from the passive state to the active state (according to the actual compare status) without any delay. 1_B Dead-Time Counter x is enabled. The corresponding outputs switch from the passive state to the active state (according to the compare status) with the delay programmed in bit field DTM.
DTR2, DTR1, DTR0	14, 13, 12	rh	Dead Time Run Indication Bits Bits DTR0..DTR2 indicate the status of the dead time generation for each compare channel (0, 1, 2) of timer T12. 0_B Dead-Time Counter x is currently in the passive state. 1_B Dead-Time Counter x is currently in the active state.
0	11, [31:15]	r	Reserved; Returns 0 if read; should be written with 0.

27.2.9 Capture/Compare Control Registers

27.2.9.1 Channel State Bits

The Compare State Register CMPSTAT contains status bits monitoring the current capture and compare state and control bits defining the active/passive state of the compare channels.

CMPSTAT

Compare State Register (60_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T13 IM	C OUT 63PS	C OUT 62PS	CC 62PS	C OUT 61PS	CC 61PS	C OUT 60PS	CC 60PS	0	CC 63ST	CC POS 62	CC POS 61	CC POS 60	CC 62ST	CC 61ST	CC 60ST
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	r	rh	rh	rh	rh	rh	rh	rh

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
CC60ST, CC61ST, CC62ST, CC63ST ¹⁾	0, 1, 2, 6	rh	Capture/Compare State Bits Bits CC6xST monitor the state of the capture/compare channels. Bits CC6xST (x = 0, 1, 2) are related to T12, bit CC63ST is related to T13. 0_B In compare mode, the timer count is less than the compare value. In capture mode, the selected edge has not yet been detected since the bit has been cleared by SW the last time. 1_B In compare mode, the counter value is greater than or equal to the compare value. In capture mode, the selected edge has been detected.
CCPOS60, CCPOS61, CCPOS62	3, 4, 5	rh	Sampled Hall Pattern Bits Bits CCPOS6x (x = 0, 1, 2) are indicating the value of the input Hall pattern that has been compared to the current and expected value. The value is sampled when the event HCRDY (Hall Compare Ready) occurs. 0_B The input CCPOSx has been sampled as 0. 1_B The input CCPOSx has been sampled as 1.
CC60PS, CC61PS, CC62PS, COOUT60PS, COOUT61PS, COOUT62PS, COOUT63PS ²⁾	8, 10, 12, 9, 11, 13, 14	rwh	Passive State Select for Compare Outputs Bits CC6xPS, COOUT6xPS select the state of the corresponding compare channel, that is considered to be the passive state. During the passive state, the passive level (defined in register PSLR) is driven by the output pin. Bits CC6xPS, COOUT6xPS (x = 0, 1, 2) are related to T12, bit CC63PS is related to T13. 0_B The corresponding compare signal is in passive state while CC6xST is 0. 1_B The corresponding compare signal is in passive state while CC6xST is 1. In capture mode, these bits are not used.

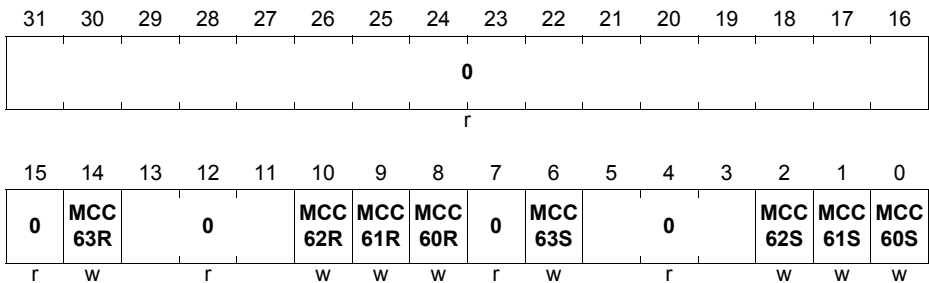
Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
T13IM 3)	15	rwh	T13 Inverted Modulation Bit T13IM inverts the T13 signal for the modulation of the CC6x and COU6x (x = 0, 1, 2) signals. 0 _B T13 output CC63_O is equal to <u>CC63ST</u> . 1 _B T13 output CC63_O is equal to <u>CC63ST</u> .
0	7, [31:16]	r	Reserved; Returns 0 if read; should be written with 0.

- 1) These bits are set and cleared according to the T12, T13 switching rules
- 2) These bits have shadow bits and are updated in parallel to the capture/compare registers of T12, T13 respectively. A read action targets the actually used values, whereas a write action targets the shadow bits.
- 3) This bit has a shadow bit and is updated in parallel to the compare and period registers of T13. A read action targets the actually used values, whereas a write action targets the shadow bit.

Capture/Compare Unit 6 (CCU6)

The Compare Status Modification Register CMPMODIF provides software-control (independent set and clear conditions) for the channel state bits CC6xST. This feature enables the user to individually change the status of the output lines by software, for example when the corresponding compare timer is stopped.

CMPMODIF
Compare State Modification Register
(64_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
MCC60S, MCC61S, MCC62S, MCC63S, MCC60R, MCC61R, MCC62R, MCC63R	0, 1, 2, 6, 8, 9, 10, 14	w	Capture/Compare Status Modification Bits These bits are used to bits to set (MCC6xS) or to clear (MCC6xR) the corresponding bits CC6xST by SW. This feature allows the user to individually change the status of the output lines by SW, e.g. when the corresponding compare timer is stopped. This allows a bit manipulation of CC6xST-bits by a single data write action. The following functionality of a write access to bits concerning the same capture/compare state bit is provided: [MCC6xR, MCC6xS] = 00 _B Bit CC6xST is not changed. 01 _B Bit CC6xST is set. 10 _B Bit CC6xST is cleared. 11 _B reserved
0	[5:3], 7, [13:11], [31:15]	r	Reserved; Returns 0 if read; should be written with 0.

27.2.9.2 T12 Mode Control Register

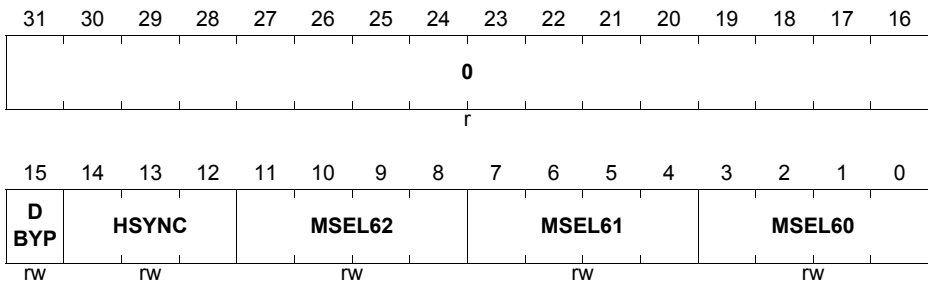
Register T12MSEL contains control bits to select the capture/compare functionality of the three channels of Timer T12.

T12MSEL

T12 Mode Select Register

(68_H)

Reset Value: 0000 0000_H

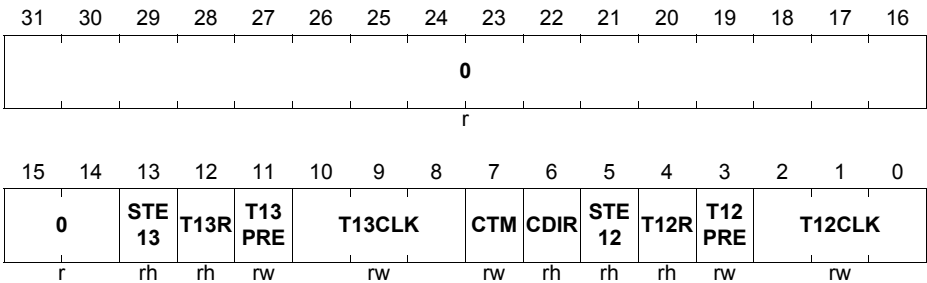


Field	Bits	Type	Description
MSEL60, MSEL61, MSEL62	[3:0], [7:4], [11:8]	rw	Capture/Compare Mode Selection These bit fields select the operating mode of the three T12 capture/compare channels. Each channel (x = 0, 1, 2) can be programmed individually for one of these modes (except for Hall Sensor Mode). Coding see Table 27-5 .
HSYNC	[14:12]	rw	Hall Synchronization Bit field HSYNC defines the source for the sampling of the Hall input pattern and the comparison to the current and the expected Hall pattern bit fields. Coding see Table 27-11 .
DBYP	15	rw	Delay Bypass DBYP controls whether the source signal for the sampling of the Hall input pattern (selected by HSYNC) is delayed by the Dead-Time Counter 0. 0 _B The bypass is not active. Dead-Time Counter 0 is generating a delay after the source signal becomes active. 1 _B The bypass is active. Dead-Time Counter 0 is not used for a delay.
0	[31:16]	r	Reserved; Returns 0 if read; should be written with 0.

27.2.9.3 Timer Control Registers

Register TCTR0 controls the basic functionality of both timers, T12 and T13.

Note: A write action to the bit fields T12CLK or T12PRE is only taken into account while the timer T12 is not running (T12R=0). A write action to the bit fields T13CLK or T13PRE is only taken into account while the timer T13 is not running (T13R=0).

TCTR0
Timer Control Register 0
(70_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
T12CLK	[2:0]	rw	Timer T12 Input Clock Select Selects the input clock for timer T12 that is derived from the peripheral clock according to the equation $f_{T12} = f_{CC6} / 2^{<T12CLK>}$. 000 _B $f_{T12} = f_{CC6}$ 001 _B $f_{T12} = f_{CC6} / 2$ 010 _B $f_{T12} = f_{CC6} / 4$ 011 _B $f_{T12} = f_{CC6} / 8$ 100 _B $f_{T12} = f_{CC6} / 16$ 101 _B $f_{T12} = f_{CC6} / 32$ 110 _B $f_{T12} = f_{CC6} / 64$ 111 _B $f_{T12} = f_{CC6} / 128$
T12PRE	3	rw	Timer T12 Prescaler Bit In order to support higher clock frequencies, an additional prescaler factor of 1/256 can be enabled for the prescaler for T12. 0 _B The additional prescaler for T12 is disabled. 1 _B The additional prescaler for T12 is enabled.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
T12R	4	rh	Timer T12 Run Bit¹⁾ T12R starts and stops timer T12. It is set/cleared by SW by setting bits T12RR or T12RS or it is cleared by HW according to the function defined by bit field T12SSC. 0 _B Timer T12 is stopped. 1 _B Timer T12 is running.
STE12	5	rh	Timer T12 Shadow Transfer Enable Bit STE12 enables or disables the shadow transfer of the T12 period value, the compare values and passive state select bits and levels from their shadow registers to the actual registers if a T12 shadow transfer event is detected. Bit STE12 is cleared by hardware after the shadow transfer. A T12 shadow transfer event is a period-match while counting up or a one-match while counting down. 0 _B The shadow register transfer is disabled. 1 _B The shadow register transfer is enabled.
CDIR	6	rh	Count Direction of Timer T12 This bit is set/cleared according to the counting rules of T12. 0 _B T12 counts up. 1 _B T12 counts down.
CTM	7	rw	T12 Operating Mode 0 _B Edge-aligned Mode: T12 always counts up and continues counting from zero after reaching the period value. 1 _B Center-aligned Mode: T12 counts down after detecting a period-match and counts up after detecting a one-match.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
T13CLK	[10:8]	rw	Timer T13 Input Clock Select Selects the input clock for timer T13 that is derived from the peripheral clock according to the equation $f_{T13} = f_{CC6} / 2^{<T13CLK>}$. 000 _B $f_{T13} = f_{CC6}$ 001 _B $f_{T13} = f_{CC6} / 2$ 010 _B $f_{T13} = f_{CC6} / 4$ 011 _B $f_{T13} = f_{CC6} / 8$ 100 _B $f_{T13} = f_{CC6} / 16$ 101 _B $f_{T13} = f_{CC6} / 32$ 110 _B $f_{T13} = f_{CC6} / 64$ 111 _B $f_{T13} = f_{CC6} / 128$
T13PRE	11	rw	Timer T13 Prescaler Bit In order to support higher clock frequencies, an additional prescaler factor of 1/256 can be enabled for the prescaler for T13. 0 _B The additional prescaler for T13 is disabled. 1 _B The additional prescaler for T13 is enabled.
T13R	12	rh	Timer T13 Run Bit²⁾ T13R starts and stops timer T13. It is set/cleared by SW by setting bits T13RR or T13RS or it is set/cleared by HW according to the function defined by bit fields T13SSC, T13TEC and T13TED. 0 _B Timer T13 is stopped. 1 _B Timer T13 is running.
STE13	13	rh	Timer T13 Shadow Transfer Enable Bit STE13 enables or disables the shadow transfer of the T13 period value, the compare value and passive state select bit and level from their shadow registers to the actual registers if a T13 shadow transfer event is detected. Bit STE13 is cleared by hardware after the shadow transfer. A T13 shadow transfer event is a period-match. 0 _B The shadow register transfer is disabled. 1 _B The shadow register transfer is enabled.
0	[31:14]	r	Reserved; Returns 0 if read; should be written with 0.

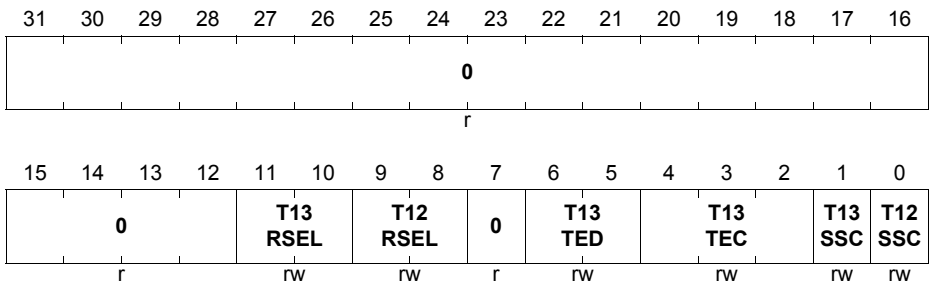
1) A concurrent set/clear action on T12R (from T12SSC, T12RR or T12RS) will have no effect. The bit T12R will remain unchanged.

Capture/Compare Unit 6 (CCU6)

- 2) A concurrent set/cleared action on T13R (from T13SSC, T13TEC, T13RR or T13RS) will have no effect. The bit T12R will remain unchanged.

Capture/Compare Unit 6 (CCU6)

Register TCTR2 controls the single-shot and the synchronization functionality of both timers T12 and T13. Both timers can run in single-shot mode. In this mode they stop their counting sequence automatically after one counting period with a count value of zero. The single-shot mode and the synchronization feature of T13 to T12 allow the generation of events with a programmable delay after well-defined PWM actions of T12.

TCTR2
Timer Control Register 2
(74_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
T12SSC	0	rw	Timer T12 Single Shot Control This bit controls the single shot-mode of T12. 0 _B The single-shot mode is disabled, no HW action on T12R. 1 _B The single shot mode is enabled, the bit T12R is cleared by HW if - T12 reaches its period value in edge-aligned mode - T12 reaches the value 1 while down counting in center-aligned mode. In parallel to the clear action of bit T12R, the bits CC6xST (x=0, 1, 2) are cleared.
T13SSC	1	rw	Timer T13 Single Shot Control This bit controls the single shot-mode of T13. 0 _B No HW action on T13R 1 _B The single-shot mode is enabled, the bit T13R is cleared by HW if T13 reaches its period value. In parallel to the clear action of bit T13R, the bit CC63ST is cleared.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
T13TEC	[4:2]	rw	T13 Trigger Event Control Bit field T13TEC selects the trigger event to start T13 (automatic set of T13R for synchronization to T12 compare signals) according to following combinations: 000 _B no action 001 _B set T13R on a T12 compare event on channel 0 010 _B set T13R on a T12 compare event on channel 1 011 _B set T13R on a T12 compare event on channel 2 100 _B set T13R on any T12 compare event (ch. 0, 1, 2) 101 _B set T13R upon a period-match of T12 110 _B set T13R upon a zero-match of T12 (while counting up) 111 _B set T13R on any edge of inputs CCPOSx
T13TED	[6:5]	rw	Timer T13 Trigger Event Direction¹⁾ Bit field T13TED delivers additional information to control the automatic set of bit T13R in the case that the trigger action defined by T13TEC is detected. 00 _B reserved, no action 01 _B while T12 is counting up 10 _B while T12 is counting down 11 _B independent on the count direction of T12
T12RSEL	[9:8]	rw	Timer T12 External Run Selection Bit field T12RSEL defines the event of signal T12HR that can set the run bit T12R by HW. 00 _B The external setting of T12R is disabled. 01 _B Bit T12R is set if a rising edge of signal T12HR is detected. 10 _B Bit T12R is set if a falling edge of signal T12HR is detected. 11 _B Bit T12R is set if an edge of signal T12HR is detected.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
T13RSEL	[11:10]	rw	Timer T13 External Run Selection Bit field T13RSEL defines the event of signal T13HR that can set the run bit T13R by HW. 00 _B The external setting of T13R is disabled. 01 _B Bit T13R is set if a rising edge of signal T13HR is detected. 10 _B Bit T13R is set if a falling edge of signal T13HR is detected. 11 _B Bit T13R is set if an edge of signal T13HR is detected.
0	7, [31:12]	r	Reserved; Returns 0 if read; should be written with 0;

1) Example:

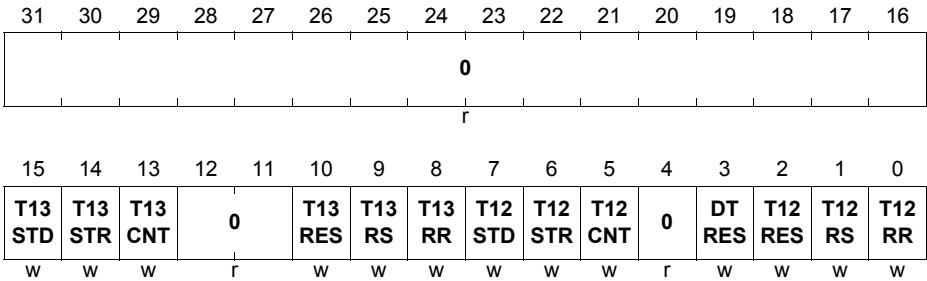
If the timer T13 is intended to start at any compare event on T12 (T13TEC=100) the trigger event direction can be programmed to

- counting up >> a T12 channel 0, 1, 2 compare match triggers T13R only while T12 is counting up
- counting down >> a T12 channel 0, 1, 2 compare match triggers T13R only while T12 is counting down
- independent from bit CDIR >> each T12 channel 0, 1, 2 compare match triggers T13R

The timer count direction is taken from the value of bit CDIR. As a result, if T12 is running in edge-aligned mode (counting up only), T13 can only be started automatically if bit field T13TED=01 or 11.

Capture/Compare Unit 6 (CCU6)

Register TCTR4 provides software-control (independent set and clear conditions) for the run bits T12R and T13R. Furthermore, the timers can be reset (while running) and bits STE12 and STE13 can be controlled by software. Reading these bits always returns 0.

TCTR4
Timer Control Register 4
(78_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
T12RR	0	w	Timer T12 Run Reset Setting this bit clears the T12R bit. 0 _B T12R is not influenced. 1 _B T12R is cleared, T12 stops counting.
T12RS	1	w	Timer T12 Run Set Setting this bit sets the T12R bit. 0 _B T12R is not influenced. 1 _B T12R is set, T12 starts counting.
T12RES	2	w	Timer T12 Reset 0 _B No effect on T12. 1 _B The T12 counter register is cleared to zero. The switching of the output signals is according to the switching rules. Setting of T12RES has no impact on bit T12R.
DTRES	3	w	Dead-Time Counter Reset 0 _B No effect on the dead-time counters. 1 _B The three dead-time counter channels are cleared to zero.
T12CNT	5	w	Timer T12 Count Event 0 _B No action 1 _B If enabled (PISEL2), timer T12 counts one step.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
T12STR	6	w	Timer T12 Shadow Transfer Request 0 _B No action 1 _B STE12 is set, enabling the shadow transfer.
T12STD	7	w	Timer T12 Shadow Transfer Disable 0 _B No action 1 _B STE12 is cleared without triggering the shadow transfer.
T13RR	8	w	Timer T13 Run Reset Setting this bit clears the T13R bit. 0 _B T13R is not influenced. 1 _B T13R is cleared, T13 stops counting.
T13RS	9	w	Timer T13 Run Set Setting this bit sets the T13R bit. 0 _B T13R is not influenced. 1 _B T13R is set, T13 starts counting.
T13RES	10	w	Timer T13 Reset 0 _B No effect on T13. 1 _B The T13 counter register is cleared to zero. The switching of the output signals is according to the switching rules. Setting of T13RES has no impact on bit T13R.
T13CNT	13	w	Timer T13 Count Event 0 _B No action 1 _B If enabled (PISEL2), timer T13 counts one step.
T13STR	14	w	Timer T13 Shadow Transfer Request 0 _B No action 1 _B STE13 is set, enabling the shadow transfer.
T13STD	15	w	Timer T13 Shadow Transfer Disable 0 _B No action 1 _B STE13 is cleared without triggering the shadow transfer.
0	4, [12:11], [31:16]	r	reserved; returns 0 if read; should be written with 0;

Note: A simultaneous write of a 1 to bits that set and clear the same bit will trigger no action. The corresponding bit will remain unchanged.

27.3 Operating Timer T13

Timer T13 is implemented similarly to Timer T12, but only with one channel in compare mode. A 16-bit up-counter is connected to a channel register via a comparator, that generates a signal when the counter contents match the contents of the channel register. A variety of control functions facilitate the adaptation of the T13 structure to different application needs. In addition, T13 can be started synchronously to timer T12 events.

This section provides information about:

- T13 overview (see [Section 27.3.1](#))
- Counting scheme (see [Section 27.3.2](#))
- Compare mode (see [Section 27.3.3](#))
- Compare output path (see [Section 27.3.4](#))
- Shadow register transfer (see [Section 27.3.5](#))
- T13 counter register description (see [Section 27.3.6](#))

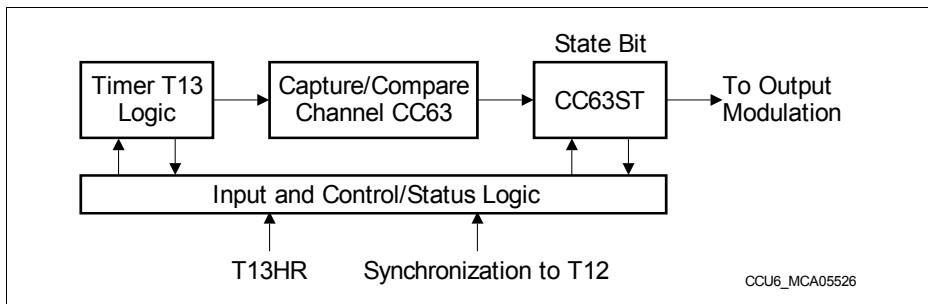


Figure 27-23 Overview Diagram of the Timer T13 Block

27.3.1 T13 Overview

Figure 27-24 shows a detailed block diagram of Timer T13. The functions of the timer T12 block are controlled by bits in registers **TCTR0**, **TCTR2**, and **PISEL2**.

Timer T13 receives its input clock, f_{T13} , from the module clock f_{CC6} via a programmable prescaler and an optional 1/256 divider or from an input signal T13HR. T13 can only count up (similar to the Edge-Aligned mode of T12).

Via a comparator, the timer T13 Counter Register **T13** is connected to the Period Register **T13PR**. This register determines the maximum count value for T13. When T13 reaches the period value, signal T13_PM (T13 Period Match) is generated and T13 is cleared to 0000_H with the next T13 clock edge. The Period Register receives a new period value from its Shadow Period Register, T13PS, that is loaded via software. The transfer of a new period value from the shadow register into T13PR is controlled via the 'T13 Shadow Transfer' control signal, T13_ST. The generation of this signal depends on the associated control bit STE13. Providing a shadow register for the period value as

Capture/Compare Unit 6 (CCU6)

well as for other values related to the generation of the PWM signal facilitates a concurrent update by software for all relevant parameters (refer to [Table 27.3.5](#)). Another signal indicates whether the counter contents are equal to 0000_H (T13_ZM). A Single-Shot control bit, T13SSC, enables an automatic stop of the timer when the current counting period is finished (see [Figure 27-26](#)).

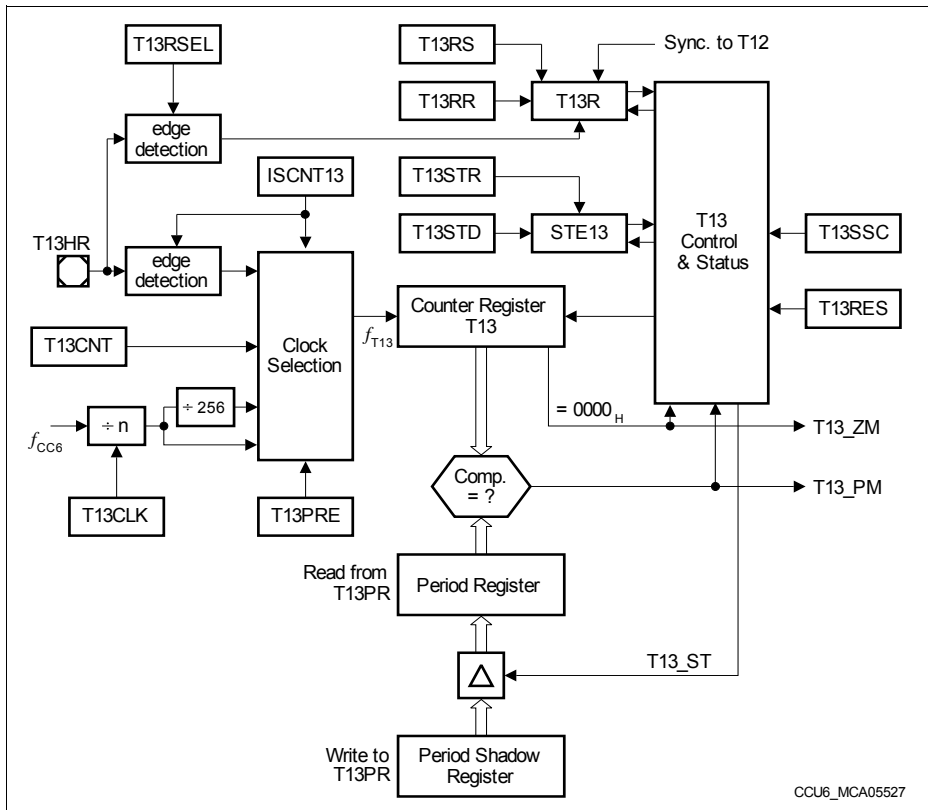


Figure 27-24 T13 Counter Logic and Period Comparators

The start or stop of T13 is controlled by the Run bit, T13R. This control bit can be set by software via the associated set/clear bits T13RS or T13RR in register [TCTR4](#), or it is cleared by hardware according to preselected conditions (single-shot mode). The timer T13 run bit T13R must not be set while the applied T13 period value is zero. Bit T13R can be set automatically if an event of T12 is detected to synchronize T13 timings to T12 events, e.g. to generate a programmable delay via T13 after an edge of a T12 compare channel before triggering an AD conversion (T13 can trigger ADC

Capture/Compare Unit 6 (CCU6)

conversions).

Timer T13 can be cleared to 0000_H via control bit T13RES. Setting this write-only bit only clears the timer contents, but has no further effects, e.g., it does not stop the timer.

The generation of the T13 shadow transfer control signal, T13_ST, is enabled via bit STE13. This bit can be set or cleared by software indirectly through its associated set/reset control bits T13STR and T13STD.

Two bit fields, T13TEC and T13TED, control the synchronization of T13 to Timer T12 events. T13TEC selects the trigger event, while T13TED determines for which T12 count direction the trigger should be active.

While Timer T13 is running, write accesses to the count register T13 are not taken into account. If T13 is stopped, write actions to register T13 are immediately taken into account.

Note: The T13 Period Register and its associated shadow register are located at the same physical address. A write access to this address targets the Shadow Register, while a read access reads from the actual period register.

27.3.2 T13 Counting Scheme

This section describes the clocking and the counting capabilities of T13.

27.3.2.1 Clock Selection

In **Timer Mode** (**PISEL2**.ISCNT13 = 00_B), the input clock f_{T13} of Timer T13 is derived from the internal module clock f_{CC6} through a programmable prescaler and an optional 1/256 divider. The resulting prescaler factors are listed in **Table 27-6**. The prescaler of T13 is cleared while T13 is not running (**TCTR0**.T13R = 0) to ensure reproducible timings and delays.

Table 27-6 Timer T13 Input Clock Options

T13CLK	Resulting Input Clock f_{T13} Prescaler Off (T13PRE = 0)	Resulting Input Clock f_{T13} Prescaler On (T13PRE = 1)
000 _B	f_{CC6}	$f_{CC6} / 256$
001 _B	$f_{CC6} / 2$	$f_{CC6} / 512$
010 _B	$f_{CC6} / 4$	$f_{CC6} / 1024$
011 _B	$f_{CC6} / 8$	$f_{CC6} / 2048$
100 _B	$f_{CC6} / 16$	$f_{CC6} / 4096$
101 _B	$f_{CC6} / 32$	$f_{CC6} / 8192$
110 _B	$f_{CC6} / 64$	$f_{CC6} / 16384$
111 _B	$f_{CC6} / 128$	$f_{CC6} / 32768$

In **Counter Mode**, timer T13 counts one step:

- If a 1 is written to **TCTR4**.T13CNT and **PISEL2**.ISCNT13 = 01_B
- If a rising edge of input signal T13HR is detected and **PISEL2**.ISCNT13 = 10_B
- If a falling edge of input signal T13HR is detected and **PISEL2**.ISCNT13 = 11_B

27.3.2.2 T13 Counting

The period of the timer is determined by the value in the period Register T13PR according to the following formula:

$$T13_{PER} = \langle \text{Period-Value} \rangle + 1; \text{ in } T13 \text{ clocks } (f_{T13}) \quad (27.3)$$

Timer T13 can only count up, comparable to the Edge-Aligned mode of T12. This leads to very simple 'counting rule' for the T13 counter:

- The counter is cleared with the next T13 clock edge if a Period-Match is detected. The counting direction is always upwards.

The behavior of T13 is illustrated in **Figure 27-25**.

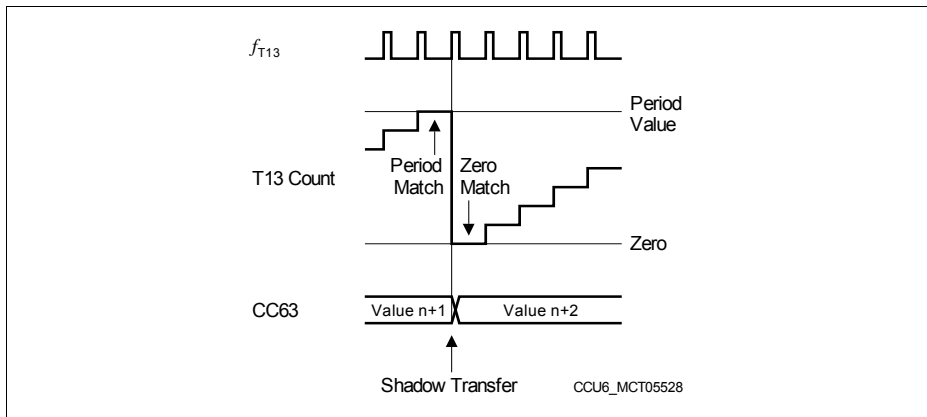


Figure 27-25 T13 Counting Sequence

27.3.2.3 Single-Shot Mode

In Single-Shot Mode, the timer run bit T13R is cleared by hardware. If bit T13SSC = 1, the timer T13 will stop when the current timer period is finished.

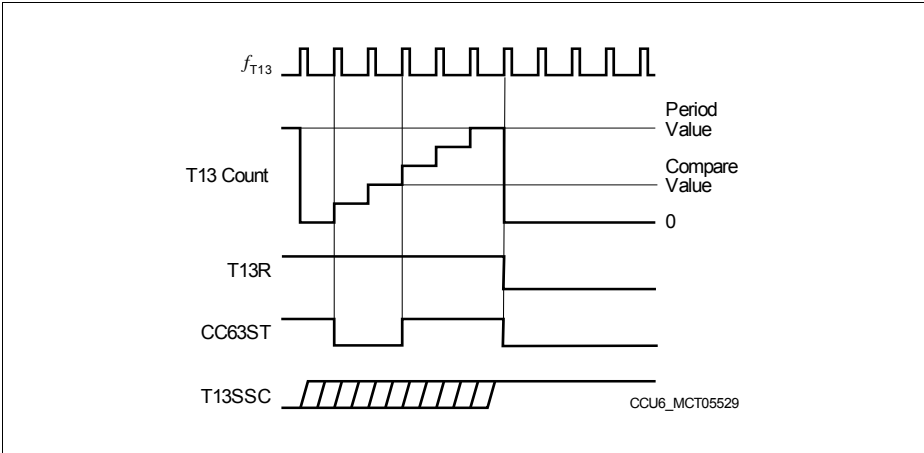


Figure 27-26 Single-Shot Operation of Timer T13

27.3.2.4 Synchronization to T12

Timer T13 can be synchronized to a T12 event. Bit fields T13TEC and T13TED select the event that is used to start Timer T13. The selected event sets bit T13R via HW, and T13 starts counting. Combined with the Single-Shot mode, this feature can be used to generate a programmable delay after a T12 event.

Figure 27-27 shows an example for the synchronization of T13 to a T12 event. Here, the selected event is a compare-match (compare value = 2) while counting up. The clocks of T12 and T13 can be different (other prescaler factor); the figure shows an example in which T13 is clocked with half the frequency of T12.

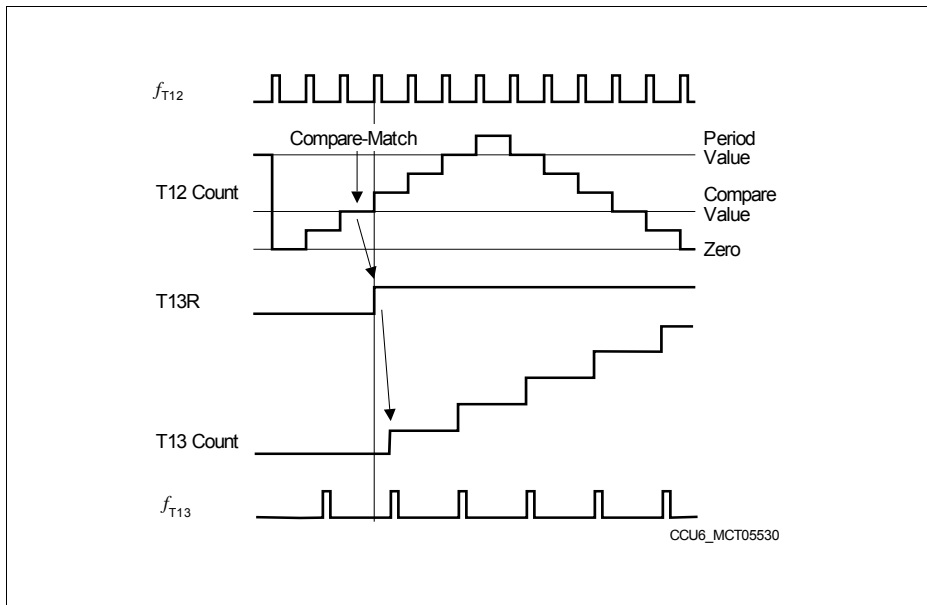


Figure 27-27 Synchronization of T13 to T12 Compare Match

Bit field T13TEC selects the trigger event to start T13 (automatic set of T13R for synchronization to T12 compare signals) according to the combinations shown in [Table 27-7](#). Bit field T13TED additionally specifies for which count direction of T12 the selected trigger event should be regarded (see [Table 27-8](#)).

Table 27-7 T12 Trigger Event Selection

T13TEC	Selected Event
000 _B	None
001 _B	T12 Compare Event on Channel 0 (CM_CC60)
010 _B	T12 Compare Event on Channel 1 (CM_CC61)
011 _B	T12 Compare Event on Channel 2 (CM_CC62)
100 _B	T12 Compare Event on any Channel (0, 1, 2)
101 _B	T12 Period-Match (T12_PM)
110 _B	T12 Zero-Match while counting up (T12_ZM and CDIR = 0)
111 _B	Any Hall State Change

Table 27-8 T12 Trigger Event Additional Specifier

T13TED	Selected Event Specifier
00 _B	Reserved, no action
01 _B	Selected event is active while T12 is counting up (CDIR = 0)
10 _B	Selected event is active while T12 is counting down (CDIR = 1)
11 _B	Selected event is active independently of the count direction of T12

27.3.3 T13 Compare Mode

Associated with Timer T13 is one compare channel, that can perform compare operations with regard to the contents of the T13 counter.

Figure 27-23 gives an overview on the T13 channel in Compare Mode. The channel is connected to the T13 counter register via an equal-to comparator, generating a compare match signal when the contents of the counter matches the contents of the compare register.

The channel consists of the comparator and a double register structure - the actual compare register, **CC63R**, feeding the comparator, and an associated shadow register, **CC63SR**, that is preloaded by software and transferred into the compare register when signal T13 shadow transfer, T13_ST, gets active. Providing a shadow register for the compare value as well as for other values related to the generation of the PWM signal facilitates a concurrent update by software for all relevant parameters.

Associated with the channel is a State Bit, **CMPSTAT.CC63ST**, holding the status of the compare operation. **Figure 27-28** gives an overview on the logic for the State Bit.

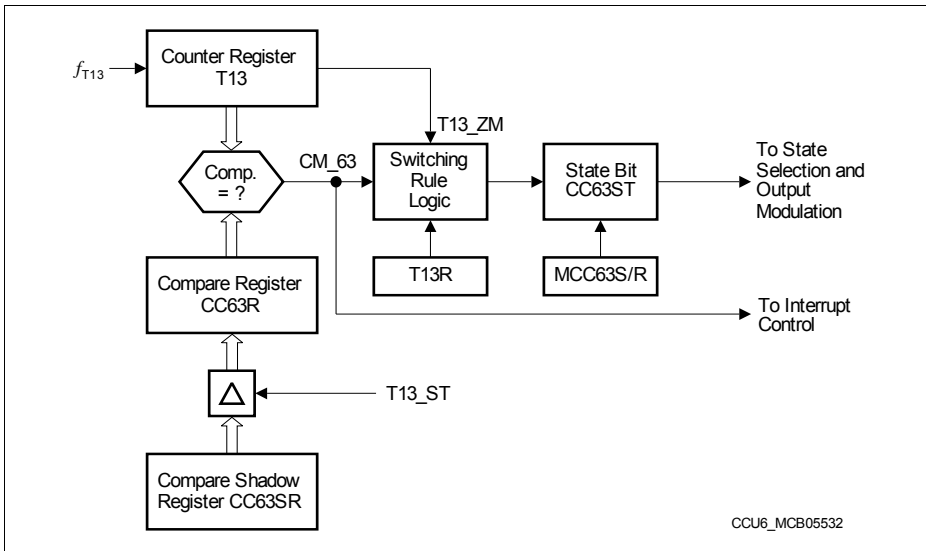


Figure 27-28 T13 State Bit Block Diagram

A compare interrupt event CM_63 is signaled when a compare match is detected. The actual setting of a State Bit has no influence on the interrupt generation.

The inputs to the switching rule logic for the CC63ST bit are the timer run bit (T13R), the timer zero-match signal (T13_ZM), and the actual individual compare-match signal CM_63. In addition, the state bit can be set or cleared by software via bits MCC63S and

Capture/Compare Unit 6 (CCU6)

MCC63R in register **CMPMODIF**.

A modification of the State Bit CC63ST by hardware is only possible while Timer T13 is running ($T13R = 1$). If this is the case, the following switching rules apply for setting and resetting the State Bit in Compare Mode:

State Bit **CC63ST** is set to 1

- with the next T13 clock (f_{T13}) after a compare-match (T13 is always counting up) (i.e., when the counter is incremented above the compare value);
- with the next T13 clock (f_{T13}) after a zero-match AND a parallel compare-match.

State Bit **CC63ST** is cleared to 0

- with the next T13 clock (f_{T13}) after a zero-match AND NO parallel compare-match.

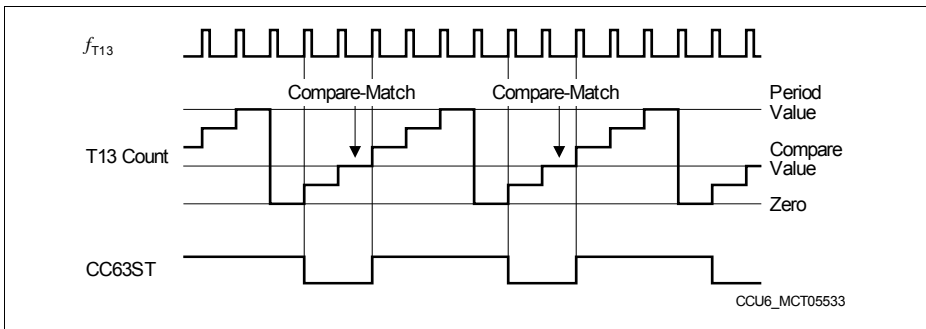


Figure 27-29 T13 Compare Operation

27.3.4 Compare Mode Output Path

Figure 27-30 gives an overview on the signal path from the channel State Bit CC63ST to its output pin COUT63. As illustrated, a user can determine the desired output behavior in relation to the current state of CC63ST. Please refer to [Section 27.2.4.3](#) for detailed information on the output modulation for T12 signals.

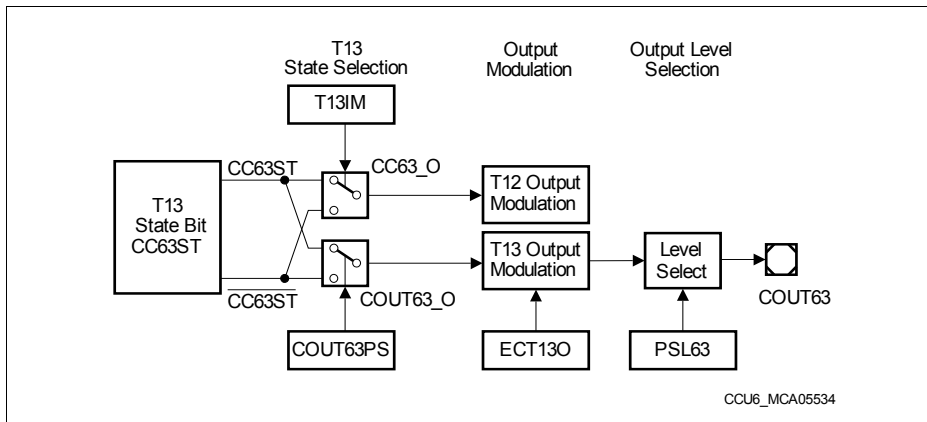


Figure 27-30 Channel 63 Output Path

The output line COUT63_O can generate a T13 PWM at the output pin COUT63. The signal CC63_O can be used to modulate the T12-related output signals with a T13 PWM. In order to decouple COUT63 from the internal modulation, the compare state leading to an active signal can be selected independently by bits T13IM and COUT63PS.

The last block of the data path is the Output Modulation block. Here, the modulation source T13 and the trap functionality are combined and control the actual level of the output pin COUT63 (see [Figure 27-31](#)):

- The **T13 related compare signal** COUT63_O delivered by the T13 state selection with the enable bit **MODCTR.ECT130**
- The **trap state** TRPS with an individual enable bit **TRPCTR.TRPEN13**

If the modulation input signal COUT63_O is enabled (ECT130 = 1) and is at passive state, the modulated is also in passive state. If the modulation input is not enabled, the output is in passive state.

If the Trap State is active (TRPS = 1), then the output enabled for the trap signal (by TRPEN13 = 1) is set to the passive state.

The output of the modulation control block is connected to a level select block. It offers the option to determine the actual output level of a pin, depending on the state of the output line (decoupling of active/passive state and output polarity) as specified by the Passive State Select bit **PSLR.PSL63**. If the modulated output signal is in the passive

Capture/Compare Unit 6 (CCU6)

state, the level specified directly by PSL63 is output. If it is in the active state, the inverted level of PSL63 is output. This allows the user to adapt the polarity of an active output signal to the connected circuitry.

The PSL63 bit has a shadow register to allow for updates with the T13 shadow transfer signal (T13_ST) without undesired pulses on the output lines. A read action returns the actually used value, whereas a write action targets the shadow bit. Providing a shadow register for the PSL value as well as for other values related to the generation of the PWM signal facilitates a concurrent update by software for all relevant parameters.

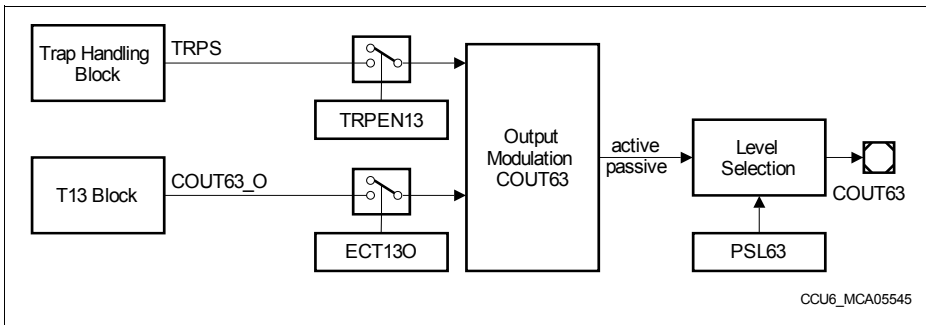


Figure 27-31 T13 Output Modulation

27.3.5 T13 Shadow Register Transfer

A special shadow transfer signal (T13_ST) can be generated to facilitate updating the period and compare values of the compare channel CC63 synchronously to the operation of T13. Providing a shadow register for values defining one PWM period facilitates a concurrent update by software for all relevant parameters. The next PWM period can run with a new set of parameters. The generation of this signal is requested by software via bit **TCTR0.STE13** (set by writing 1 to the write-only bit **TCTR4.T13STR**, cleared by writing 1 to the write-only bit **TCTR4.T13STD**).

When signal T13_ST is active, a shadow register transfer is triggered with the next cycle of the T13 clock. Bit STE13 is automatically cleared with the shadow register transfer.

A T13 shadow register transfer takes place (T13_ST active):

- while timer T13 is not running (T13R = 0), or
- STE13 = 1 and a Period-Match is detected while T13R = 1

Capture/Compare Unit 6 (CCU6)

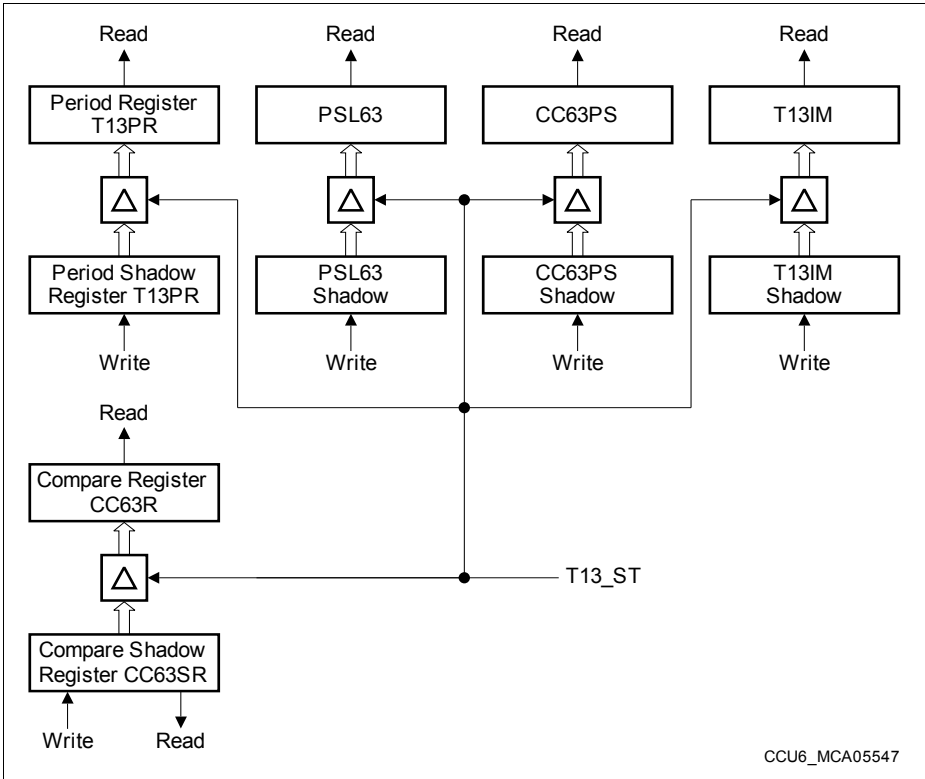


Figure 27-32 T13 Shadow Register Overview

27.3.6 T13 related Registers

27.3.6.1 T13 Counter Register

The generation of the patterns for a single channel pulse width modulation (PWM) is based on timer T13. The registers related to timer T13 can be concurrently updated (with well-defined conditions) in order to ensure consistency of the PWM signal. T13 can be synchronized to several timer T12 events.

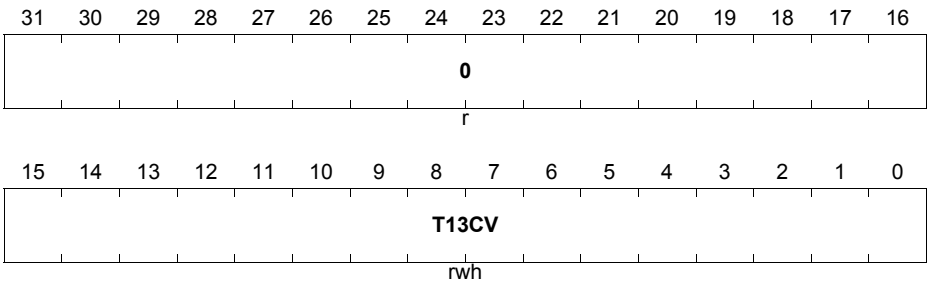
Timer T13 only supports compare mode on its compare channel CC63.

Register T13 represents the counting value of timer T13. It can only be written while the timer T13 is stopped. Write actions while T13 is running are not taken into account. Register T13 can always be read by SW.

Timer T13 only supports edge-aligned mode (counting up).

T13

Timer T13 Counter Register (50_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
T13CV	[15:0]	rwh	Timer 13 Counter Value This register represents the 16-bit counter value of Timer13.
0	[31:16]	r	Reserved; Returns 0 if read; should be written with 0.

Note: While timer T13 is stopped, the internal clock divider is reset in order to ensure reproducible timings and delays.

27.3.6.2 Period Register

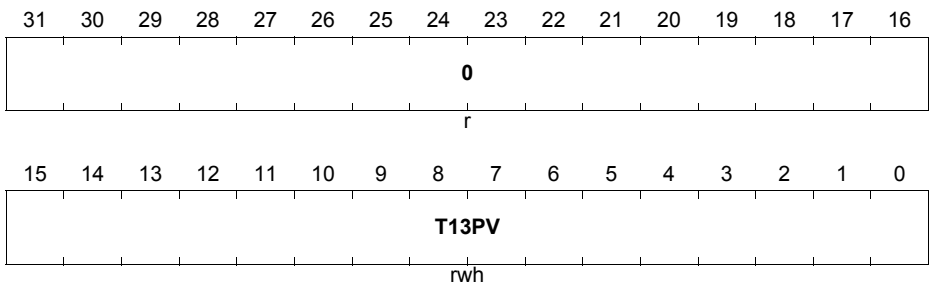
Register T13PR contains the period value for timer T13. The period value is compared to the actual counter value of T13 and the resulting counter actions depend on the defined counting rules. This register has a shadow register and the shadow transfer is controlled by bit STE13. A read action by SW delivers the value currently used for the compare action, whereas the write action targets a shadow register. The shadow register structure allows a concurrent update of all T13-related values.

T13PR

Timer 13 Period Register

(54_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
T13PV	[15:0]	rwh	T13 Period Value The value T13PV defines the counter value for T13 leading to a period-match. When reaching this value, the timer T13 is set to zero.
0	[31:16]	r	Reserved; Returns 0 if read; should be written with 0.

27.3.6.3 Compare Register

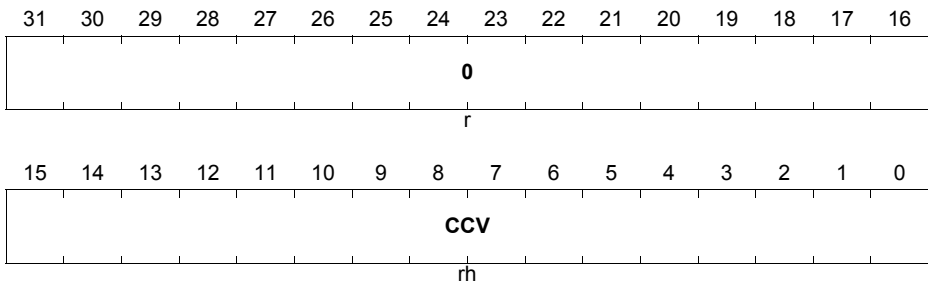
Registers CC63R is the actual compare register for T13. The values stored in CC63R is compared to the counter value of T13. The State Bit CC63ST is located in register **CMPSTAT**.

CC63R

Compare Register for T13

(58_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CCV	[15:0]	rh	Channel CC63 Compare Value The bit field CCV contains the value, that is compared to the T13 counter value.
0	[31:16]	r	Reserved; Returns 0 if read; should be written with 0.

27.3.6.4 Compare Shadow Register

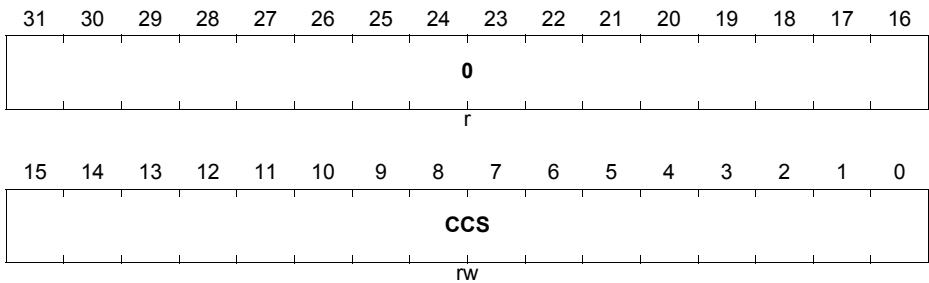
The register CC63R can only be read by SW, the modification of the value is done by a shadow register transfer from register CC63SR. The corresponding shadow register CC63SR can be read and written by SW.

CC63SR

Compare Shadow Register for T13

(5C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CCS	[15:0]	rw	Shadow Register for Channel CC63 Compare Value The bit field contents of CCS is transferred to the bit field CCV during a shadow transfer.
0	[31:16]	r	Reserved; Returns 0 if read; should be written with 0.

27.4 Synchronous Start Feature

The T12 and T13 timers can be started synchronously through the T12HR and T13HR inputs of all CCU6x kernels.

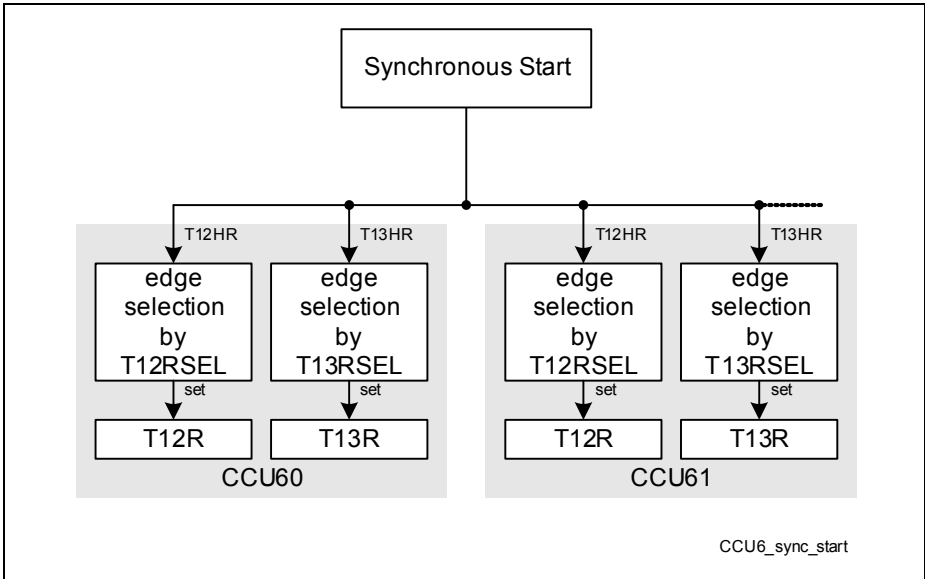


Figure 27-33 Synchronization Concept

27.5 Trap Handling

The trap functionality permits the PWM outputs to react on the state of the input signal $\overline{\text{CTRAP}}$. This functionality can be used to switch off the power devices if the trap input becomes active (e.g. to perform an emergency stop). The trap handling and the effect on the output modulation are controlled by the bits in the trap control register **TRPCTR**. The trap flags TRPF and TRPS are located in register **IS** and can be set/cleared by SW by writing to registers **ISS** and **ISR**.

Figure 27-34 gives an overview on the trap function.

The Trap Flag TRPF monitors the trap input and initiates the entry into the Trap State. The Trap State Bit TRPS determines the effect on the outputs and controls the exit of the Trap State.

When a trap condition is detected ($\overline{\text{CTRAP}} = 0$) and the input is enabled ($\text{TRPPEN} = 1$), both, the Trap Flag TRPF and the Trap State Bit TRPS, are set to 1 (trap state active). The output of the Trap State Bit TRPS leads to the Output Modulation Blocks (for T12 and for T13) and can there deactivate the outputs (set them to the passive state). Individual enable control bits for each of the six T12-related outputs and the T13-related output facilitate a flexible adaptation to the application needs.

There are a number of different ways to exit the Trap State. This offers SW the option to select the best operation for the application. Exiting the Trap State can be done either immediately when the trap condition is removed ($\overline{\text{CTRAP}} = 1$ or $\text{TRPPEN} = 0$), or under software control, or synchronously to the PWM generated by either Timer T12 or Timer T13.

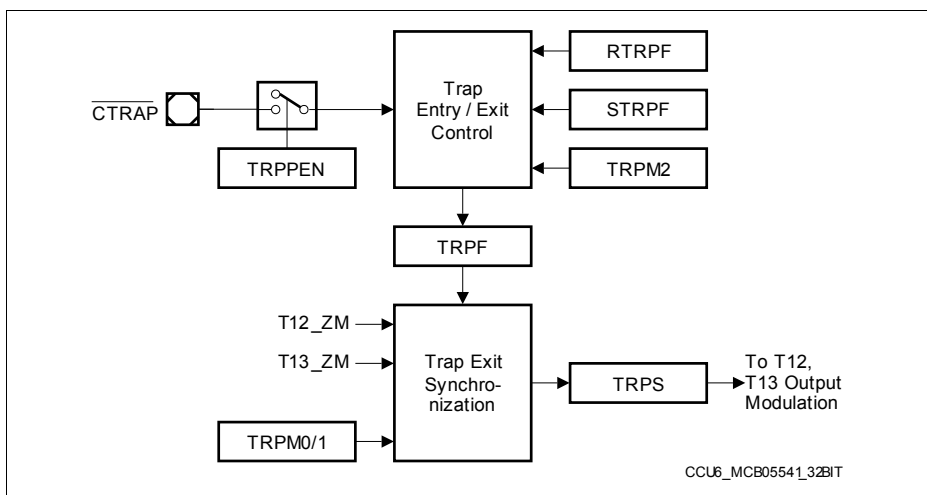


Figure 27-34 Trap Logic Block Diagram

Capture/Compare Unit 6 (CCU6)

Clearing of TRPF is controlled by the mode control bit TRPM2. If $TRPM2 = 0$, TRPF is automatically cleared by HW when \overline{CTRAP} returns to the inactive level ($\overline{CTRAP} = 1$) or if the trap input is disabled ($TRPPEN = 0$). When $TRPM2 = 1$, TRPF must be reset by SW after \overline{CTRAP} has become inactive.

Clearing of TRPS is controlled by the mode control bits TRPM1 and TRPM0 (located in the Trap Control Register TRPCTR). A reset of TRPS terminates the Trap State and returns to normal operation. There are three options selected by TRPM1 and TRPM0. One is that the Trap State is left immediately when the Trap Flag TRPF is cleared, without any synchronization to timers T12 or T13. The other two options facilitate the synchronization of the termination of the Trap State to the count periods of either Timer T12 or Timer T13. **Figure 27-35** gives an overview on the associated operation.

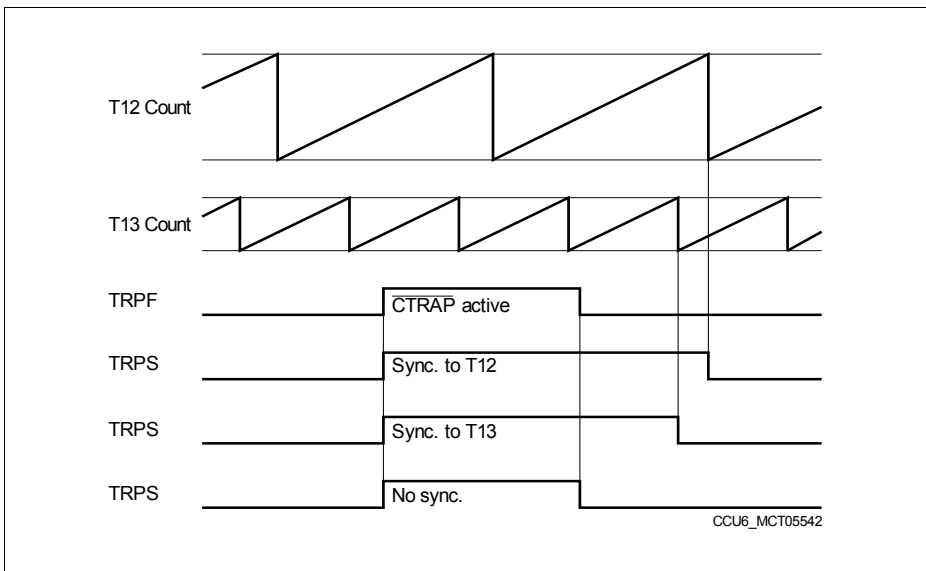


Figure 27-35 Trap State Synchronization (with $TRPM2 = 0$)

27.6 Multi-Channel Mode

The Multi-Channel mode offers the possibility to modulate all six T12-related output signals with one instruction. The bits in bit field **MCMOUT.MCMP** are used to specify the outputs that may become active. If Multi-Channel mode is enabled (bit **MODCTR.MCMEN** = 1), only those outputs may become active, that have a 1 at the corresponding bit position in bit field **MCMP**.

This bit field has its own shadow bit field **MCMOUTS.MCMPS**, that can be written by software. The transfer of the new value in **MCMP** to the bit field **MCMP** can be triggered by, and synchronized to, T12 or T13 events. This structure permits the software to write the new value, that is then taken into account by the hardware at a well-defined moment and synchronized to a PWM signal. This avoids unintended pulses due to unsynchronized modulation sources.

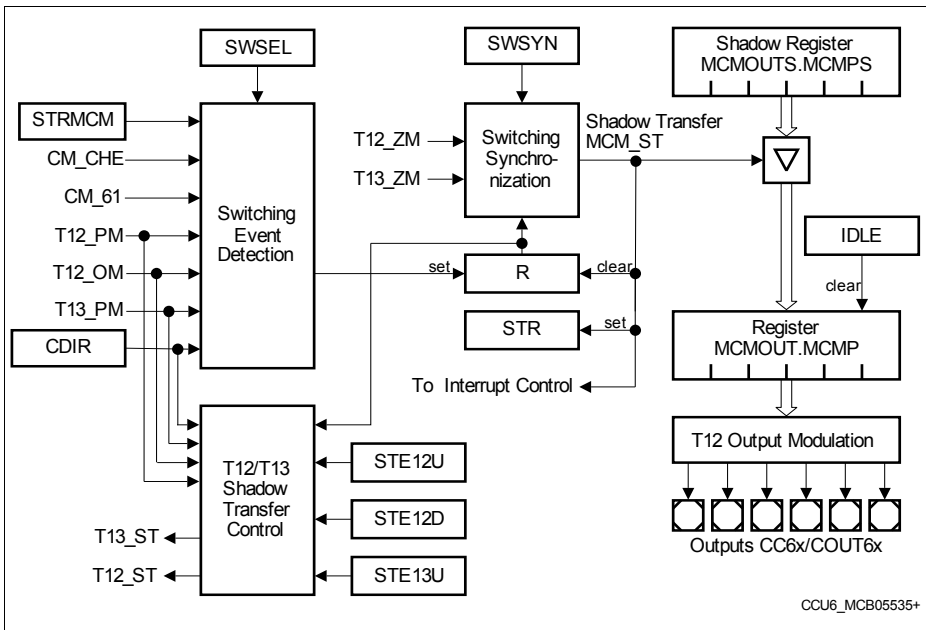


Figure 27-36 Multi-Channel Mode Block Diagram

Figure 27-36 shows the functional blocks for the Multi-Channel operation, controlled by bit fields in register **MCMCTR**. The event that triggers the update of bit field **MCMP** is chosen by **SWSEL**. In order to synchronize the update of **MCMP** to a PWM generated by T12 or T13, bit field **SWSYN** allows the selection of the synchronization event leading to the transfer from **MCMPS** to **MCMP**. Due to this structure, an update takes place with a new PWM period. A reminder flag **R** is set when the selected switching event occurs

Capture/Compare Unit 6 (CCU6)

(the event is not necessarily synchronous to the modulating PWM), and is cleared when the transfer takes place. This flag can be monitored by software to check for the status of this logic block. If the shadow transfer from MCMPS to MCMP takes place, bit **IS.STR** becomes set and an interrupt can be generated.

In addition to the Multi-Channel shadow transfer event MCM_ST, the shadow transfers for T12 (T12_ST) and T13 (T13_ST) can be generated to allow concurrent updates of applied duty cycles for T12 and/or T13 modulation and Multi-Channel patterns.

If it is explicitly desired, the update takes place immediately with the occurrence of the selected event when the direct synchronization mode is selected. The update can also be requested by software by writing to bit field MCMPS with the shadow transfer request bit STRMCM = 1. The option to trigger an update by SW is possible for all settings of SWSEL.

By using the direct mode and bit STRMCM = 1, the update takes place completely under software control.

Table 27-9 Multi-Channel Mode Switching Event Selection

SWSEL	Selected Event (see register MCMCTR)
000 _B	No automatic event detection
001 _B	Correct Hall Event (CM_CHE) detected at input signals CCPOSx without additional delay
010 _B	T13 Period-Match (T13_PM)
011 _B	T12 One-Match while counting down (T12_OM and CDIR = 1)
100 _B	T12 Compare Channel 1 Event while counting up (CM_61 and CDIR = 0) to support the phase delay function by CC61 for block commutation mode.
101 _B	T12 Period-Match while counting up (T12_PM and CDIR = 0)
110 _B , 111 _B	Reserved, no action

Table 27-10 Multi-Channel Mode Switching Synchronization

SWSYN	Synchronization Event (see register MCMCTR)
00 _B	Direct Mode: the trigger event directly causes the shadow transfer
01 _B	T13 Zero-Match (T13_ZM), the MCM shadow transfer is synchronized to a T13 PWM
10 _B	T12 Zero-Match (T12_ZM), the MCM shadow transfer is synchronized to a T12 PWM
11 _B	Reserved, no action

27.7 Hall Sensor Mode

For Brushless DC-Motors in block commutation mode, the Multi-Channel Mode has been introduced to provide efficient means for switching pattern generation. These patterns need to be output in relation to the angular position of the motor. For this, usually Hall sensors or Back-EMF sensing are used to determine the angular rotor position. The CCU6 provides three inputs, CCPOS0, CCPOS1, and CCPOS2, that can be used as inputs for the Hall sensors or the Back-EMF detection signals.

There is a strong correlation between the motor position and the output modulation pattern. When a certain position of the motor has been reached, indicated by the sampled Hall sensor inputs (the Hall pattern), the next, pre-determined Multi-Channel Modulation pattern has to be output. Because of different machine types, the modulation pattern for driving the motor can vary. Therefore, it is wishful to have a wide flexibility in defining the correlation between the Hall pattern and the corresponding Modulation pattern. Furthermore, a hardware mechanism significantly reduces the CPU for block-commutation.

The CCU6 offers the flexibility by having a register containing the currently assumed Hall pattern (CURH), the next expected Hall pattern (EXPH) and the corresponding output pattern (MCMP). A new Modulation pattern is output when the sampled Hall inputs match the expected ones (EXPH). To detect the next rotation phase (segment for block commutation), the CCU6 monitors the Hall inputs for changes. When the next expected Hall pattern is detected, the next corresponding Modulation pattern is output.

To increase for noise immunity (to a certain extend), the CCU6 offers the possibility to introduce a sampling delay for the Hall inputs. Some changes of the Hall inputs are not leading to the expected Hall pattern, because they are only short spikes due to noise. The Hall pattern compare logic compares the Hall inputs to the next expected pattern and also to the currently assumed pattern to filter out spikes.

For the Hall and Modulation output patterns, a double-register structure is implemented. While register **MCMOUT** holds the actually used values, its shadow register **MCMOUTS** can be loaded by software from a pre-defined table, holding the appropriate Hall and Modulation patterns for the given motor control.

A transfer from the shadow register into register MCMOUT can take place when a correct Hall pattern change is detected. Software can then load the next values into register MCMOUTS. It is also possible by software to force a transfer from MCMOUTS into MCMOUT.

Note: The Hall input signals CCPOSx and the CURH and EXPH bit fields are arranged in the following order:

CCPOS0 corresponds to CURH.0 (LSB) and EXPH.0 (LSB)

CCPOS1 corresponds to CURH.1 and EXPH.1

CCPOS2 corresponds to CURH.2 (MSB) and EXPH.2 (MSB)

27.7.1 Hall Pattern Evaluation

The Hall sensor inputs CCPOSx can be permanently monitored via an edge detection block (with the module clock f_{CC6}). In order to suppress spikes on the Hall inputs due to noise in rugged inverter environment, two optional noise filtering methods are supported by the Hall logic (both methods can be combined).

- Noise filtering with delay:

For this function, the mode control bit fields MSEL6x for all T12 compare channels must be programmed to 1000_B and DBYP = 0. The selected event triggers Dead-Time Counter 0 to generate a programmable delay (defined by bit field DTM). When the delay has elapsed, the evaluation signal HCRDY becomes activated.

Output modulation with T12 PWM signals is not possible in this mode.
- Noise filtering by synchronization to PWM:

The Hall inputs are not permanently monitored by the edge detection block, but samples are taken only at defined points in time during a PWM period. This can be used to sample the Hall inputs when the switching noise (due to PWM) does not disturb the Hall input signals.

If neither the delay function of Dead-Time Counter 0 is not used for the Hall pattern evaluation nor the Hall mode for Brushless DC-Drive control is enabled, the timer T12 block is available for PWM generation and output modulation.

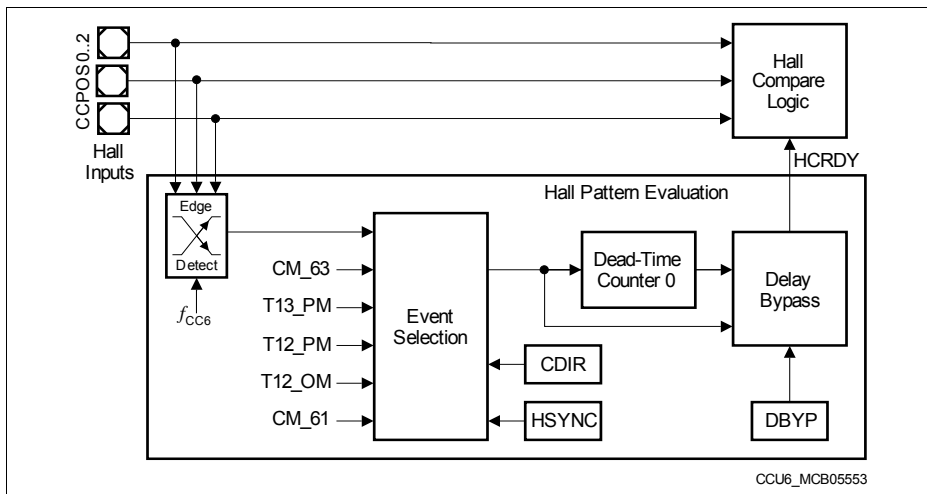


Figure 27-37 Hall Pattern Evaluation

If the evaluation signal HCRDY (Hall Compare Ready, see [Figure 27-38](#)) becomes activated, the Hall inputs are sampled and the Hall compare logic starts the evaluation of the Hall inputs.

Capture/Compare Unit 6 (CCU6)

Figure 27-37 illustrates the events for Hall pattern evaluation and the noise filter logic, **Table 27-11** summarizes the selectable trigger input signals.

Table 27-11 Hall Sensor Mode Trigger Event Selection

HSYNC	Selected Event (see register T12MSEL)
000 _B	Any edge at any of the inputs CCPOSx, independent from any PWM signal (permanent check).
001 _B	A T13 Compare-Match (CM_63).
010 _B	A T13 Period-Match (T13_PM).
011 _B	Hall sampling triggered by HW sources is switched off.
100 _B	A T12 Period-Match while counting up (T12_PM and CDIR = 0).
101 _B	A T12 One-Match while counting down (T12_OM and CDIR = 1).
110 _B	A T12 Compare-Match of compare channel CC61 while counting up (CM_61 and CDIR = 0).
111 _B	A T12 Compare-Match of compare channel CC61 while counting down (CM_61 and CDIR = 1).

27.7.2 Hall Pattern Compare Logic

Figure 27-38 gives an overview on the double-register structure and the pattern compare logic. Software writes the next modulation pattern (MCMPS) and the corresponding current (CURHS) and expected (EXPHS) Hall patterns into the shadow register MCMOUTS. Register MCMOUT holds the actually used values CURH and EXPH. The modulation pattern MCMPS is provided to the T12 Output Modulation block. The current (CURH) and expected (EXPH) Hall patterns are compared to the sampled Hall sensor inputs (visible in register **CMPSTAT**). Sampling of the inputs and the evaluation of the comparator outputs is triggered by the evaluation signal HCRDY (Hall Compare Ready), that is detailed in the next section.

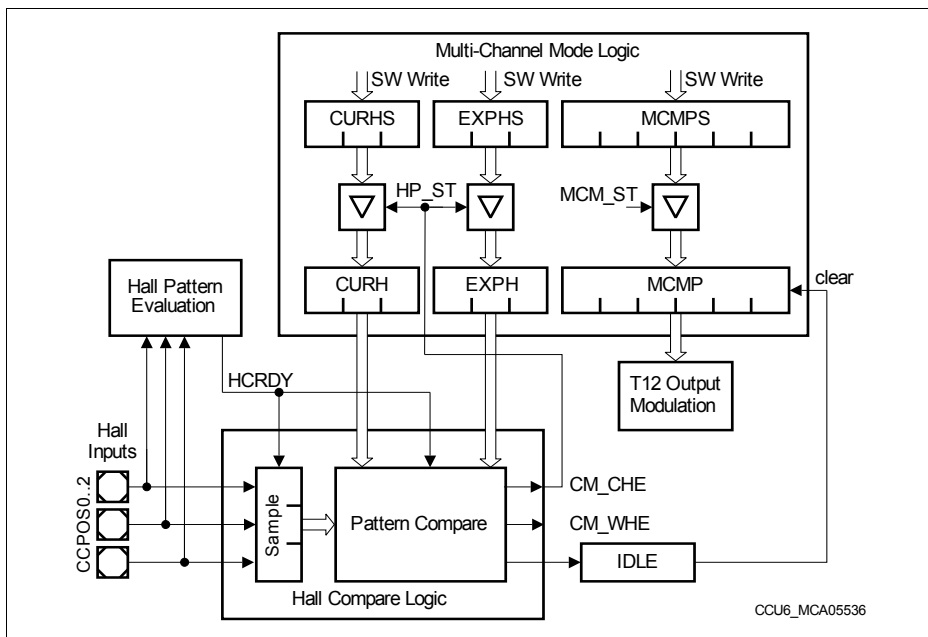


Figure 27-38 Hall Pattern Compare Logic

- If the sampled Hall pattern matches the value programmed in CURH, the detected transition was a spike (no Hall event) and no further actions are necessary.
- If the sampled Hall pattern matches the value programmed in EXPH, the detected transition was the expected event (correct Hall event CM_CHE) and the MCMPS value has to change.
- If the sampled Hall pattern matches neither CURH nor EXPH, the transition was due to a major error (wrong Hall event CM_CWE) and can lead to an emergency shut down (IDLE).

Capture/Compare Unit 6 (CCU6)

At every correct Hall event (CM_CHE), the next Hall patterns are transferred from the shadow register MCMOUTS into MCMOUT (Hall pattern shadow transfer HP_ST), and a new Hall pattern with its corresponding output pattern can be loaded (e.g. from a predefined table in memory) by software into MCMOUTS. For the Modulation patterns, signal MCM_ST is used to trigger the transfer.

Loading this shadow register can also be done by writing MCMOUTS.STRHP = 1 (for EXPH and CURH) or MCMOUTS.STRMCM = 1 (for MCM).

27.7.3 Hall Mode Flags

Depending on the Hall pattern compare operation, a number of flags are set in order to indicate the status of the module and to trigger further actions and interrupt requests.

Flag **IS.CHE** (Correct Hall Event) is set by signal CM_CHE when the sampled Hall pattern matches the expected one (EXPH). This flag can also be set by SW by setting bit **ISS.SCHE** = 1. If enabled by bit **IEN.ENCHE** = 1, the set signal for CHE can also generate an interrupt request to the CPU. Bit field **INP.INPCHE** defines which service request output becomes activated in case of an interrupt request. To clear flag CHE, SW needs to write **ISR.RCHE** = 1.

Flag **IS.WHE** indicates a Wrong Hall Event. Its handling for flag setting and resetting as well as interrupt request generation are similar to the mechanism for flag CHE.

The implementation of flag STR is done in the same way as for CHE and WHE. This flag is set by HW by the shadow transfer signal MCM_ST (see also [Figure 27-36](#)).

Please note that for flags CHE, WHE, and STR, the interrupt request generation is triggered by the set signal for the flag. That means, a request can be generated even if the flag is already set. There is no need to clear the flag in order to enable further interrupt requests.

The implementation for the IDLE flag is different. It is set by HW through signal CM_WHE if enabled by bit ENIDLE. Software can also set the flag via bit SIDLE. As long as bit IDLE is set, the modulation pattern field MCM is cleared to force the outputs to the passive state. Flag IDLE must be cleared by software by writing RIDLE = 1 in order to return to normal operation. To fully restart from IDLE mode, the transfer requests for the bit fields in register MCMOUTS to register MCMOUT have to be initiated by software via bits STRMCM and STRHP in register MCMOUTS. In this way, the release from IDLE mode is under software control, but can be performed synchronously to the PWM signal.

Capture/Compare Unit 6 (CCU6)

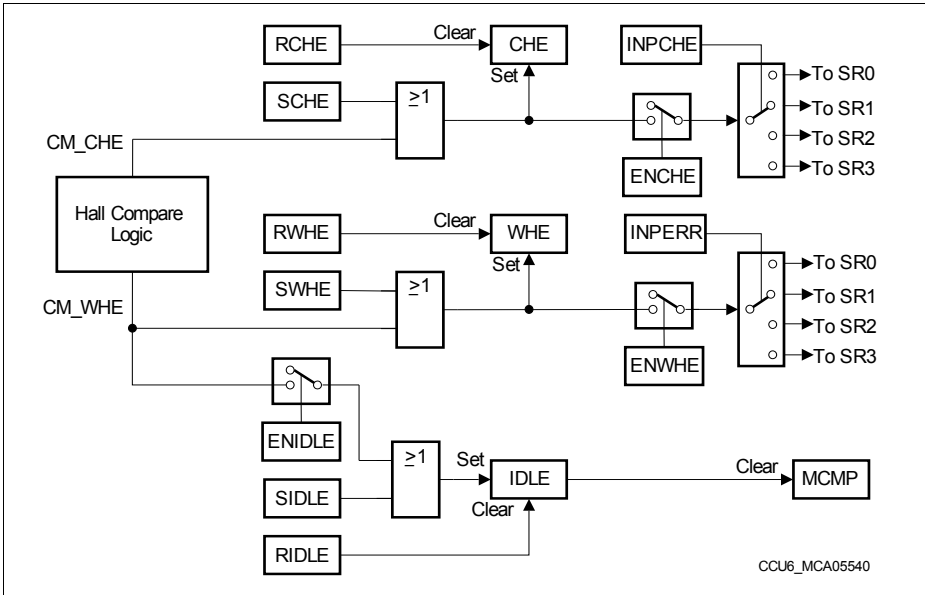


Figure 27-39 Hall Mode Flags

27.7.4 Hall Mode for Brushless DC-Motor Control

The CCU6 provides a mode for the Timer T12 Block especially targeted for convenient control of block commutation patterns for Brushless DC-Motors. This mode is selected by setting all **T12MSEL.MSEL6x** bit fields of the three T12 Channels to 1000_B. In this mode, illustrated in **Figure 27-40**, channel CC60 is placed in capture mode to measure the time elapsed between the last two correct Hall events, channel CC61 in compare mode to provide a programmable phase delay between the Hall event and the application of a new PWM output pattern, and channel CC62 also in compare mode as first time-out criterion. A second time-out criterion can be built by the T12 period match event.

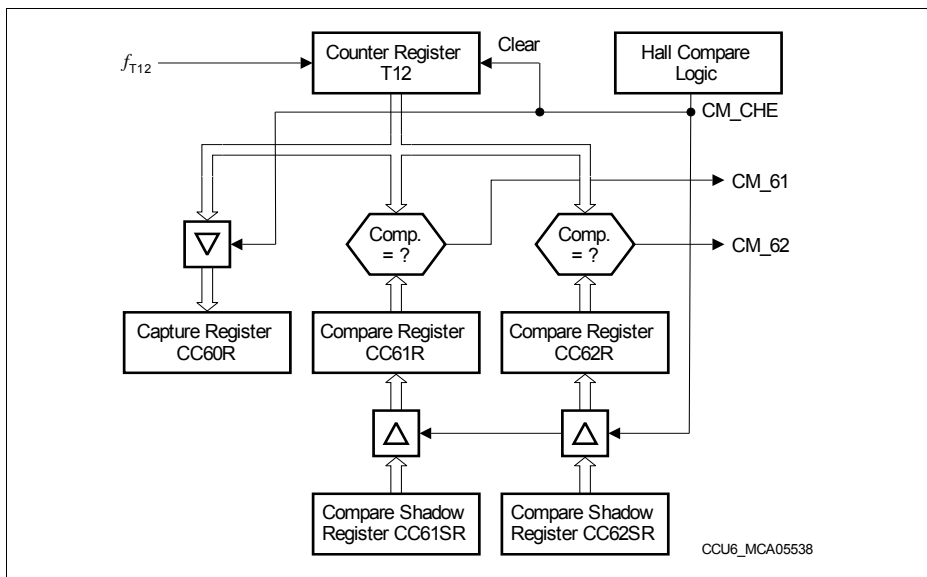


Figure 27-40 T12 Block in Hall Sensor Mode

The signal CM_CHE from the Hall compare logic is used to transfer the new compare values from the shadow registers CC6xSR into the actual compare registers CC6xR, performs the shadow transfer for the T12 period register, to capture the current T12 contents into register CC60R, and to clear T12.

Note: In this mode, the shadow transfer signal T12_ST is not generated. Not all shadow bits, such as the PSLy bits, will be transferred to their main registers. To program the main registers, SW needs to write to these registers while Timer T12 is stopped. In this case, a SW write actualizes both registers.

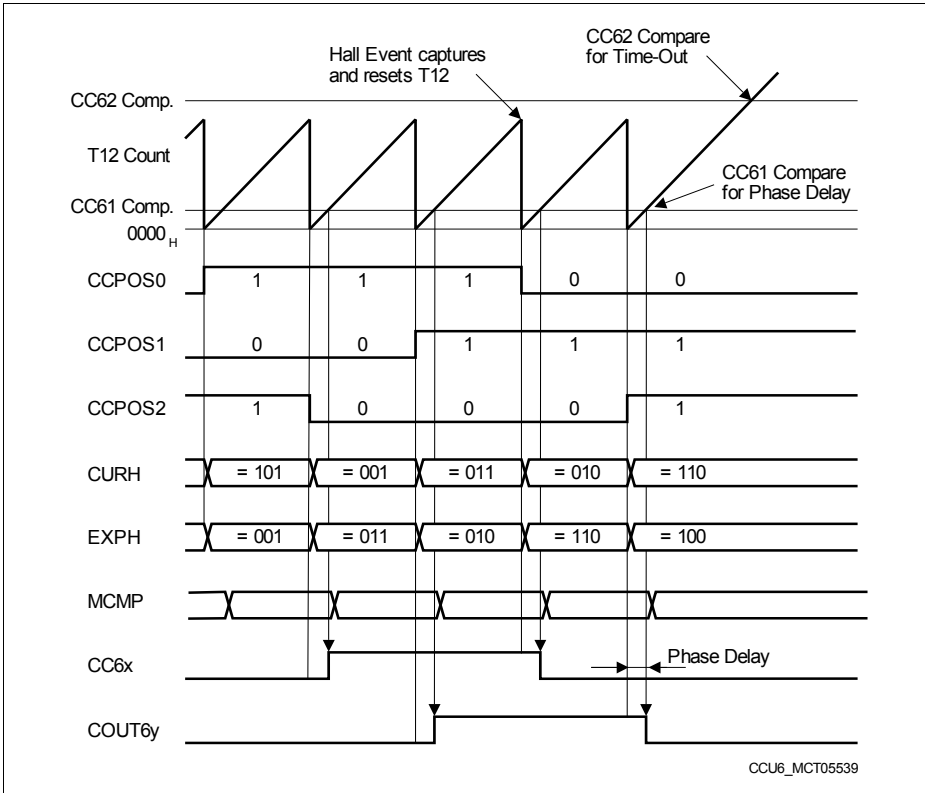


Figure 27-41 Brushless DC-Motor Control Example (all MSEL6x = 1000_B)

After the detection of an expected Hall pattern (CM_CHE active), the T12 count value is captured into channel CC60 (representing the actual rotor speed by measuring the elapsed time between the last two correct Hall events), and T12 is reset. When the timer reaches the compare value in channel CC61, the next multi-channel state is switched by triggering the shadow transfer of bit field MCMP (if enabled in bit field SWEN). This trigger event can be combined with the synchronization of the next multi-channel state to the PWM source (to avoid spikes on the output lines, see Section 27.6). This compare function of channel CC61 can be used as a phase delay from the position sensor input signals to the switching of the output signals, that is necessary if a sensorless back-EMF technique or Hall sensors are used. The compare value in channel CC62 can be used as a time-out trigger (interrupt), indicating that the actual motor speed is far below the desired destination value. An abnormal load change can be detected with this feature and PWM generation can be disabled.

27.8 Modulation Control Registers

27.8.1 Modulation Control

This register contains bits enabling the modulation of the corresponding output signal by PWM pattern generated by the timers T12 and T13. Furthermore, the multi-channel mode can be enabled as additional modulation source for the output signals.

MODCTR

Modulation Control Register

(80_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECT 130	0	T13MODEN						MCM EN	0	T12MODEN					
rw	r	rw						rw	r	rw					

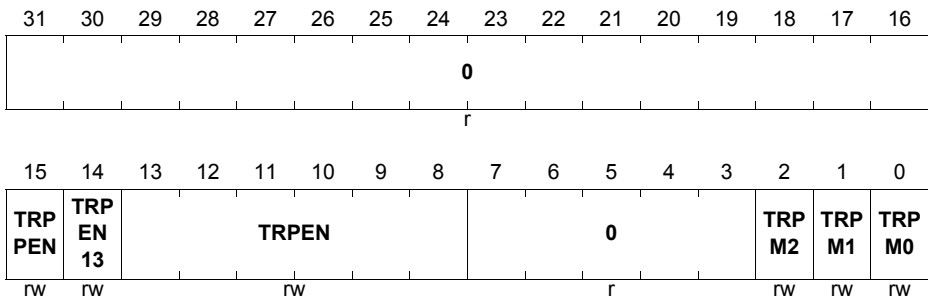
Field	Bits	Type	Description
T12MODEN	[5:0]	rw	<p>T12 Modulation Enable</p> <p>These bits enable the modulation of the corresponding output signal by a PWM pattern generated by timer T12.</p> <p>T12MODEN0 = MODCTR.0 for output CC60 T12MODEN1 = MODCTR.1 for output COUT60 T12MODEN2 = MODCTR.2 for output CC61 T12MODEN3 = MODCTR.3 for output COUT61 T12MODEN4 = MODCTR.4 for output CC62 T12MODEN5 = MODCTR.5 for output COUT62</p> <p>0_B The modulation of the corresponding output signal by a T12 PWM pattern is disabled.</p> <p>1_B The modulation of the corresponding output signal by a T12 PWM pattern is enabled.</p>
MCMEN	7	rw	<p>Multi-Channel Mode Enable</p> <p>0_B The modulation of the corresponding output signal by a multi-channel pattern according to bit field MCMOUT is disabled.</p> <p>1_B The modulation of the corresponding output signal by a multi-channel pattern according to bit field MCMOUT is enabled.</p>
T13MODEN	[13:8]	rw	<p>T13 Modulation Enable</p> <p>These bits enable the modulation of the corresponding output signal by the PWM pattern CC63_O generated by timer T13.</p> <p>T13MODEN0 = MODCTR.8 for output CC60 T13MODEN1 = MODCTR.9 for output COUT60 T13MODEN2 = MODCTR.10 for output CC61 T13MODEN3 = MODCTR.11 for output COUT61 T13MODEN4 = MODCTR.12 for output CC62 T13MODEN5 = MODCTR.13 for output COUT62</p> <p>0_B The modulation of the corresponding output signal by a T13 PWM pattern is disabled.</p> <p>1_B The modulation of the corresponding output signal by a T13 PWM pattern is enabled.</p>

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
ECT130	15	rw	Enable Compare Timer T13 Output 0_B The output COUT63 is in the passive state. 1_B The output COUT63 is enabled for the PWM signal generated by T13.
0	6, 14, [31:16]	r	Reserved; Returns 0 if read; should be written with 0.

27.8.2 Trap Control Register

The register TRPCTR controls the trap functionality. It contains independent enable bits for each output signal and control bits to select the behavior in case of a trap condition. The trap condition is a low level on the \overline{CTRAP} input pin, that is monitored (inverted level) by bit IS.TRPF. While TRPF=1 (trap input active), the trap state bit IS.TRPS is set to 1.

TRPCTR
Trap Control Register
(84_H)
Reset Value: 0000 0000_H


Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
TRPM1, TRPM0	1, 0	rw	<p>Trap Mode Control Bits 1, 0</p> <p>These two bits define the behavior of the selected outputs when leaving the trap state after the trap condition has become inactive again.</p> <p>A synchronization to the timer driving the PWM pattern avoids unintended pulses when leaving the trap state.</p> <p>The combination [TRPM1, TRPM0] leads to:</p> <p>00_B The trap state is left (return to normal operation) after TRPF has become 0 again when a zero-match of T12 (while counting up) is detected (synchronization to T12).</p> <p>01_B The trap state is left (return to normal operation) after TRPF has become 0 again when a zero-match of T13 is detected (synchronization to T13).</p> <p>10_B reserved</p> <p>11_B The trap state is left (return to normal operation) immediately after TRPF has become 0 again without any synchronization to T12 or T13.</p>
TRPM2	2	rw	<p>Trap Mode Control Bit 2</p> <p>This bit defines how the trap flag TRPF can be cleared after the trap input condition ($\overline{CTRAP} = 0$ and $\overline{TRPPEN} = 1$) is no longer valid (either by $\overline{CTRAP} = 1$ or by $\overline{TRPPEN} = 0$).</p> <p>0_B Automatic Mode: Bit TRPF is cleared by HW if the trap input condition is no longer valid.</p> <p>1_B Manual Mode: Bit TRPF stays 0 after the trap input condition is no longer valid. It has to be cleared by SW by writing $ISR.RTRPF = 1$.</p>

Capture/Compare Unit 6 (CCU6)

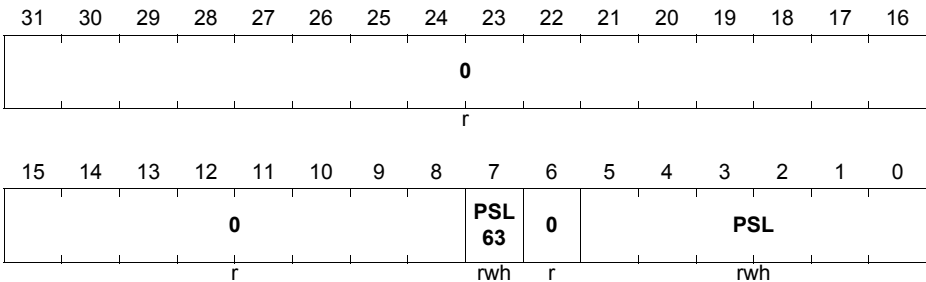
Field	Bits	Type	Description
TRPEN	[13:8]	rw	Trap Enable Control Setting a bit enables the trap functionality for the following corresponding output signals: TRPEN0 = TRPCTR.8 for output CC60 TRPEN1 = TRPCTR.9 for output COUT60 TRPEN2 = TRPCTR.10 for output CC61 TRPEN3 = TRPCTR.11 for output COUT61 TRPEN4 = TRPCTR.12 for output CC62 TRPEN5 = TRPCTR.13 for output COUT62 0_B The trap functionality of the corresponding output signal is disabled. The output state is independent from bit IS.TRPS. 1_B The trap functionality of the corresponding output signal is enabled. The output state is set to the passive while IS.TRPS=1.
TRPEN13	14	rw	Trap Enable Control for Timer T13 0_B The trap functionality for output COUT63 is disabled. The output state is independent from bit IS.TRPS. 1_B The trap functionality for output COUT63 is enabled. The output state is set to the passive while IS.TRPS=1.
TRPPEN	15	rw	Trap Pin Enable This bit enables the input (pin) function for the trap generation. An interrupt can only be generated if a falling edge is detected at pin \overline{CTRAP} while TRPPEN = 1. 0_B The CCU6 trap functionality based on the input \overline{CTRAP} is disabled. A CCU6 trap can only be generated by SW by setting bit TRPF. 1_B The CCU6 trap functionality based on the input \overline{CTRAP} is enabled. A CCU6 trap can be generated by SW by setting bit TRPF or by $\overline{CTRAP}=0$.
0	[7:3], [31:16]	r	Reserved; Returns 0 if read; should be written with 0.

27.8.3 Passive State Level Register

Register PSLR defines the passive state level of the PWM outputs of the module. The passive state level is the value that is driven during the passive state of the output. During the active state, the corresponding output pin drives the active state level, that is the inverted passive state level. The passive state level permits to adapt the driven output levels to the driver polarity (inverted, not inverted) of the connected power stage. The bits in this register have shadow bit fields to permit a concurrent update of all PWM-related parameters (bit field PSL is updated with T12_ST, whereas PSL63 is updated with T13_ST). The actually used values can be read (attribute “rh”), whereas the shadow bits can only be written (attribute “w”).

PSLR

Passive State Level Register (88_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
PSL	[5:0]	rwh	Compare Outputs Passive State Level These bits define the passive level driven by the module outputs during the passive state. PSL0 = PSLR.0 for output CC60 PSL1 = PSLR.1 for output COUT60 PSL2 = PSLR.2 for output CC61 PSL3 = PSLR.3 for output COUT61 PSL4 = PSLR.4 for output CC62 PSL5 = PSLR.5 for output COUT62 0 _B The passive level is 0. 1 _B The passive level is 1.
PSL63	7	rwh	Passive State Level of Output COUT63 This bit defines the passive level driven by the module output COUT63 during the passive state. 0 _B The passive level is 0. 1 _B The passive level is 1.
0	6, [31:8]	r	Reserved; Returns 0 if read; should be written with 0.

27.8.4 Multi-Channel Mode Registers

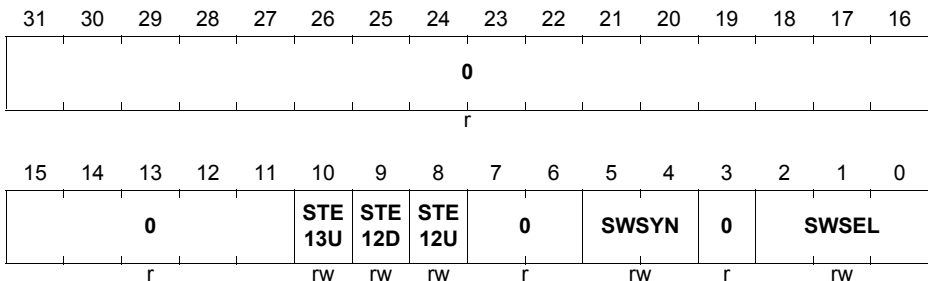
Register MCMCTR contains control bits for the multi-channel functionality.

MCMCTR

Multi-Channel Mode Control Register

(94_H)

Reset Value: 0000 0000_H



Capture/Compare Unit 6 (CCU6)

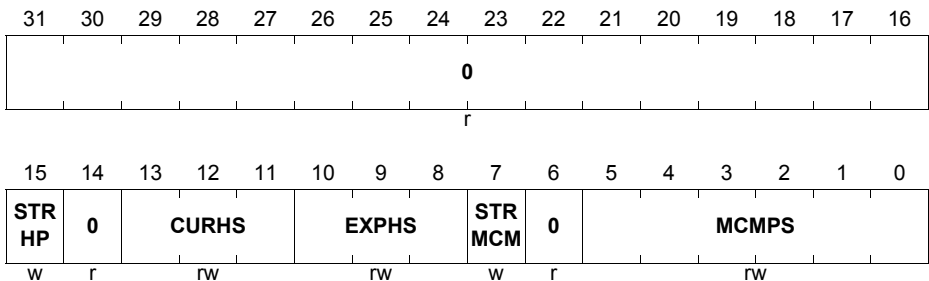
Field	Bits	Type	Description
SWSEL	[2:0]	rw	<p>Switching Selection</p> <p>Bit field SWSEL selects one of the following trigger request sources (next multi-channel event) for the shadow transfer MCM_ST from MCMP5 to MCMP. The trigger request is stored in the reminder flag R until the shadow transfer is done and flag R is cleared automatically with the shadow transfer. The shadow transfer takes place synchronously with an event selected in bit field SWSYN.</p> <p>000_B No trigger request will be generated 001_B Correct Hall pattern detected (CM_CHE) 010_B T13 period-match detected (while counting up) 011_B T12 one-match (while counting down) 100_B T12 channel 1 compare-match detected (phase delay function) 101_B T12 period match detected (while counting up) 110_B reserved, no trigger request will be generated 111_B reserved, no trigger request will be generated</p>
SWSYN	[5:4]	rw	<p>Switching Synchronization</p> <p>Bit field SWSYN defines the synchronization mechanism of the shadow transfer event MCM_ST if it has been requested before (flag R set by an event selected by SWSEL) and if MCMEN = 1. This feature permits the synchronization of the outputs to the PWM source, that is used for modulation (T12 or T13).</p> <p>00_B Direct; the trigger event immediately leads to the shadow transfer 01_B A T13 zero-match triggers the shadow transfer 10_B A T12 zero-match (while counting up) triggers the shadow transfer 11_B reserved; no action</p>
STE12U	8	rw	<p>Shadow Transfer Enable for T12 Upcounting</p> <p>This bit enables the shadow transfer T12_ST if flag MCMOUT.R is set or becomes set while a T12 period match is detected while counting up.</p> <p>0_B No action 1_B The T12_ST shadow transfer mechanism is enabled if MCMEN = 1.</p>

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
STE12D	9	rw	Shadow Transfer Enable for T12 Downcounting This bit enables the shadow transfer T12_ST if flag MCMOUT.R is set or becomes set while a T12 one match is detected while counting down. 0 _B No action 1 _B The T12_ST shadow transfer mechanism is enabled if MCMEN = 1.
STE13U	10	rw	Shadow Transfer Enable for T13 Upcounting This bit enables the shadow transfer T13_ST if flag MCMOUT.R is set or becomes set while a T13 period match is detected. 0 _B No action 1 _B The T13_ST shadow transfer mechanism is enabled if MCMEN = 1.
0	3, [7:6], [31:11]	r	Reserved; Returns 0 if read; should be written with 0.

Capture/Compare Unit 6 (CCU6)

Register MCMOUTS contains bits used as pattern input for the multi-channel mode and the Hall mode. This register is a shadow register (that can be read and written) for register MCMOUT, indicating the currently active signals.

MCMOUTS
Multi-Channel Mode Output Shadow Register
(8C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
MCMPS	[5:0]	rw	Multi-Channel PWM Pattern Shadow Bit field MCMPS is the shadow bit field for bit field MCMP. The multi-channel shadow transfer is triggered by MCM_ST according to the transfer conditions defined by register MCMCTR.
STRMCM	7	w	Shadow Transfer Request for MCMPS Writing STRMCM = 1 leads to an immediate activation of MCM_ST to update bit field MCMP by the value of MCMPS. When read, this bit always delivers 0. 0 _B No action. 1 _B Bit field MCMP is updated.
EXPHS	[10:8]	rw	Expected Hall Pattern Shadow Bit field EXPHS is the shadow bit field for bit field EXPH. The shadow transfer takes place when a correct Hall event is detected (CM_CHE).
CURHS	[13:11]	rw	Current Hall Pattern Shadow Bit field CURHS is the shadow bit field for bit field CURH. The shadow transfer takes place when a correct Hall event is detected (CM_CHE).

Capture/Compare Unit 6 (CCU6)

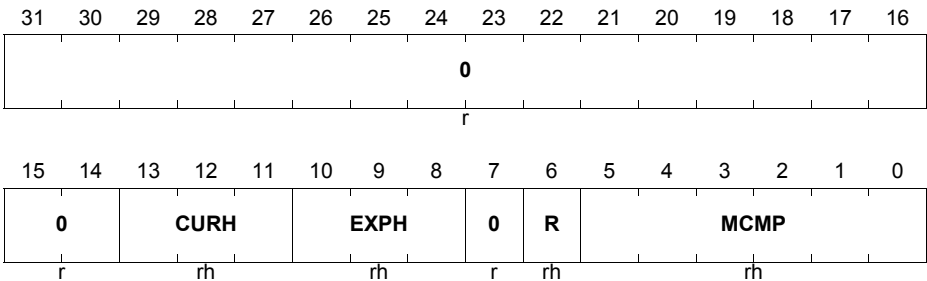
Field	Bits	Type	Description
STRHP	15	w	Shadow Transfer Request for the Hall Pattern Writing STRHP = 1 leads to an immediate activation of HP_ST to update bit fields EXPH and CURH by EXPHS and CURHS. When read, this bit always delivers 0. 0 _B No action. 1 _B Bit fields EXPH and CURH are updated.
0	6, 14, [31:16]	r	Reserved; Returns 0 if read; should be written with 0.

MCMOUT

Multi-Channel Mode Output Register

(90_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
MCMP	[5:0]	rh	<p>Multi-Channel PWM Pattern</p> <p>Bit field MCMP defines the output pattern for the multi-channel mode. If this mode is enabled by MODCTR.MCMEN = 1, the output state of all T12 related PWM outputs can be modified. This bit field is 0 while IS.IDLE = 1.</p> <p>MCMP0 = MCMOUT.0 for output CC60 MCMP1 = MCMOUT.1 for output COUT60 MCMP2 = MCMOUT.2 for output CC61 MCMP3 = MCMOUT.3 for output COUT61 MCMP4 = MCMOUT.4 for output CC62 MCMP5 = MCMOUT.5 for output COUT62</p> <p>0_B The output is set to the passive state. A PWM generated by T12 or T13 are not taken into account.</p> <p>1_B The output can be in the active state, depending on the enabled PWM modulation signals generated by T12, T13 and the trap state.</p>
R	6	rh	<p>Reminder Flag</p> <p>This flag indicates that the shadow transfer from MCMP5 to MCMP has been requested by the selected trigger source. It is cleared when the shadow transfer takes place or while MCMEN=0.</p> <p>0_B A shadow transfer MCM_ST is not requested.</p> <p>1_B A shadow transfer MCM_ST is requested, but has not yet been executed, because the selected synchronization condition has not yet occurred.</p>
EXPH	[10:8]	rh	<p>Expected Hall Pattern</p> <p>Bit field EXPH is updated by a shadow transfer HP_ST from bit field EXPHS.</p> <p>If HCRDY = 1, EXPH is compared to the sampled CCPOSx inputs in order to detect the occurrence of the next desired (=expected) hall pattern or a wrong pattern. If the sampled hall pattern at the hall input pins is equal to bit field EXPH, a correct Hall event has been detected (CM_CHE).</p>

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
CURH	[13:11]	rh	Current Hall Pattern Bit field CURH is updated by a shadow transfer HP_ST from bit field CURHS. If HCRDY = 1, CURH is compared to the sampled CCPOSx inputs in order to detect a spike. If the sampled Hall pattern at the Hall input pins is equal to bit field CURH, no Hall event has been detected. If the sampled Hall input pattern is neither equal to CURH nor equal to EXPH, the Hall event was not the desired one and may be due to a fatal error (e.g. blocked rotor, etc.). In this case, a wrong Hall event has been detected (CM_WHE).
0	7, [31:14]	r	Reserved; Returns 0 if read; should be written with 0.

27.9 Interrupt Handling

This section describes the interrupt handling of the CCU6 module.

27.9.1 Interrupt Structure

The HW interrupt event or the SW setting of the corresponding interrupt set bit (in register ISS) sets the event indication flags (in register IS) and can trigger the interrupt generation. The interrupt pulse is generated independently from the interrupt status flag in register IS (it is not necessary to clear the related status bit to be able to generate another interrupt). The interrupt flag can be cleared by SW by writing to the corresponding bit in register ISR.

If enabled by the related interrupt enable bit in register IEN, an interrupt pulse can be generated on one of the four service request outputs (SR0 to SR3) of the module. If more than one interrupt source is connected to the same interrupt node pointer (in register INP), the requests are logically OR-combined to one common service request output (see [Figure 27-42](#)).

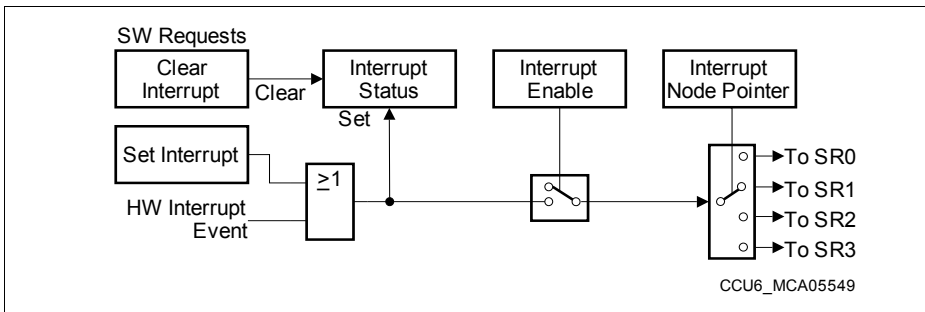


Figure 27-42 General Interrupt Structure

The available interrupt events in the CCU6 are shown in [Figure 27-43](#).

Capture/Compare Unit 6 (CCU6)

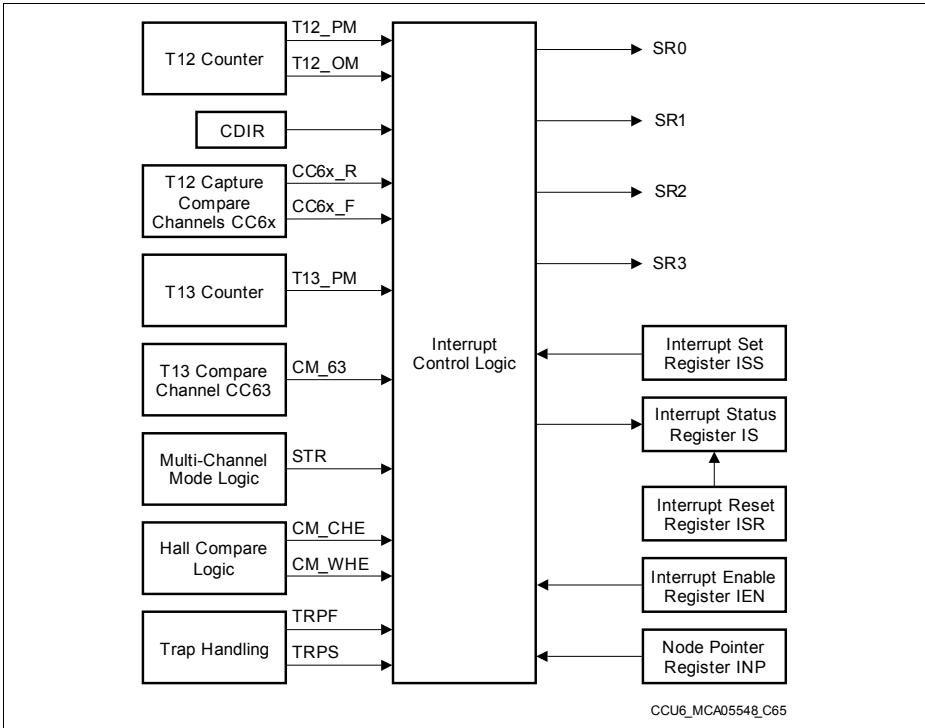


Figure 27-43 Interrupt Sources and Events

27.9.2 Interrupt Registers

27.9.2.1 Interrupt Status Register

Register IS contains the individual interrupt request bits. This register can only be read, write actions have no impact on the contents of this register. The SW can set or clear the bits individually by writing to the registers ISS (to set the bits) or to register ISR (to clear the bits).

The interrupt generation is independent from the value of the bits in register IS, e.g. the interrupt will be generated (if enabled) even if the corresponding bit is already set. The trigger for an interrupt generation is the detection of a set condition (by HW or SW) for the corresponding bit in register IS.

In compare mode (and hall mode), the timer-related interrupts are only generated while the timer is running ($T1xR=1$). In capture mode, the capture interrupts are also generated while the timer T12 is stopped.

Note: Not all bits in register IS can generate an interrupt. Other status bits have been added, that have a similar structure for their set and clear actions. It is recommended that SW checks the interrupt bits bit-wisely (instead of common OR over the bits).

IS

Interrupt Status Register

(A0_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STR	IDLE	WHE	CHE	TRP S	TRP F	T13 PM	T13 CM	T12 PM	T12 OM	ICC 62F	ICC 62R	ICC 61F	ICC 61R	ICC 60F	ICC 60R
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
ICC60R, ICC61R, ICC62R	0, 2, 4	rh	Capture, Compare-Match Rising Edge Flag This bit indicates that event CC6x_R has been detected. This event occurs in compare mode when a compare-match is detected while T12 is counting up (CM_6x and CDIR = 0) and in capture mode when a rising edge is detected at the related input CC6xIN. 0 _B The event has not yet been detected. 1 _B The event has been detected.
ICC60F, ICC61F, ICC62F	1, 3, 5	rh	Capture, Compare-Match Falling Edge Flag This bit indicates that event CC6x_F has been detected. This event occurs in compare mode when a compare-match is detected while T12 is counting down (CM_6x and CDIR = 1) and in capture mode when a falling edge is detected at the related input CC6xIN. 0 _B The event has not yet been detected. 1 _B The event has been detected.
T12OM	6	rh	Timer T12 One-Match Flag This bit indicates that a timer T12 one-match while counting down (T12_OM and CDIR = 1) has been detected. 0 _B The event has not yet been detected. 1 _B The event has been detected.
T12PM	7	rh	Timer T12 Period-Match Flag This bit indicates that a timer T12 period-match while counting up (T12_PM and CDIR = 0) has been detected. 0 _B The event has not yet been detected. 1 _B The event has been detected.
T13CM	8	rh	Timer T13 Compare-Match Flag This bit indicates that a timer T13 compare-match (CM_63) has been detected. 0 _B The event has not yet been detected. 1 _B The event has been detected.
T13PM	9	rh	Timer T13 Period-Match Flag This bit indicates that a timer T13 period-match (T13_PM) has been detected. 0 _B The event has not yet been detected. 1 _B The event has been detected.

Capture/Compare Unit 6 (CCU6)

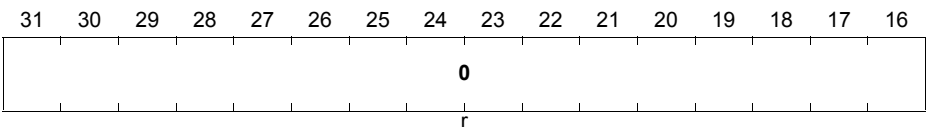
Field	Bits	Type	Description
TRPF	10	rh	Trap Flag This bit indicates if a trap condition (input $\overline{\text{CTRAP}} = 0$ or by SW) is / has been detected. If $\text{TRPM2} = 0$, it becomes cleared automatically if $\overline{\text{CTRAP}} = 1$ or $\text{TRPEN} = 0$, whereas if $\text{TRPM2} = 1$, it has to be cleared by writing $\text{RTRPF} = 1$. 0_{B} The trap condition has not been detected. 1_{B} The trap condition is / has been detected.
TRPS	11	rh	Trap State¹⁾ This bit indicates the actual trap state. It is set if $\text{TRPF} = 1$ and becomes cleared according to the mode selected in register TRPCTR . 0_{B} The trap state is not active. 1_{B} The trap state is active.
CHE	12	rh	Correct Hall Event This bit indicates that a correct Hall event (CM_CHE) has been detected. 0_{B} The event has not yet been detected. 1_{B} The event has been detected.
WHE	13	rh	Wrong Hall Event This bit indicates that a wrong Hall event (CM_WHE) has been detected. 0_{B} The event has not yet been detected. 1_{B} The event has been detected.
IDLE	14	rh	IDLE State If enabled by $\text{ENIDLE} = 1$, this bit is set together with bit WHE and it has to be cleared by SW. 0_{B} No action. 1_{B} Bit field MCMP is cleared, the selected outputs are set to passive state.
STR	15	rh	Multi-Channel Mode Shadow Transfer Request This bit indicates that a shadow transfer from MCMPS to MCMP (MCM_ST) has taken place. 0_{B} The event has not yet been detected. 1_{B} The event has been detected.
0	[31:16]	r	Reserved; Returns 0 if read; should be written with 0.

Capture/Compare Unit 6 (CCU6)

- 1) During the trap state, the selected outputs are set to the passive state. The logic level driven during the passive state is defined by the corresponding bit in register PSLR. Bits TRPS=1 and TRPF=0 can occur if the trap condition is no longer active but the selected synchronization has not yet taken place.

27.9.2.2 Interrupt Status Set Register

Register ISS contains individual interrupt request set bits to generate a CCU6 interrupt request by software. Writing a 1 sets the bit(s) in register IS at the corresponding bit position(s) and can generate an interrupt event (if available and enabled). All bit positions read as 0.

ISS
Interrupt Status Set Register
(A4_H)
Reset Value: 0000 0000_H


15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S
STR	IDLE	WHE	CHE	WHC	TRP F	T13 PM	T13 CM	T12 PM	T12 OM	CC 62F	CC 62R	CC 61F	CC 61R	CC 60F	CC 60R
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Field	Bits	Type	Description
SCC60R, SCC61R, SCC62R	0, 2, 4	w	Set Capture, Compare-Match Rising Edge Flag 0 _B No action 1 _B Bit CC6xR will be set.
SCC60F, SCC61F, SCC62F	1, 3, 5	w	Set Capture, Compare-Match Falling Edge Flag 0 _B No action 1 _B Bit CC6xF will be set.
ST12OM	6	w	Set Timer T12 One-Match Flag 0 _B No action 1 _B Bit T12OM will be set.
ST12PM	7	w	Set Timer T12 Period-Match Flag 0 _B No action 1 _B Bit T12PM will be set.
ST13CM	8	w	Set Timer T13 Compare-Match Flag 0 _B No action 1 _B Bit T13CM will be set.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
ST13PM	9	w	Set Timer T13 Period-Match Flag 0 _B No action 1 _B Bit T13PM will be set.
STRPF	10	w	Set Trap Flag 0 _B No action 1 _B Bits TRPF and TRPS will be set.
SWHC	11	w	Software Hall Compare 0 _B No action 1 _B The Hall compare action is triggered.
SCHE	12	w	Set Correct Hall Event Flag 0 _B No action 1 _B Bit CHE will be set.
SWHE	13	w	Set Wrong Hall Event Flag 0 _B No action 1 _B Bit WHE will be set.
SIDLE	14	w	Set IDLE Flag 0 _B No action 1 _B Bit IDLE will be set.
SSSTR	15	w	Set STR Flag 0 _B No action 1 _B Bit STR will be set.
0	[31:16]	r	Reserved; Returns 0 if read; should be written with 0.

27.9.2.3 Status Reset Register

Register ISR contains bits to individually clear the interrupt event flags by software. Writing a 1 clears the bit(s) in register IS at the corresponding bit position(s). All bit positions read as 0.

ISR

Interrupt Status Reset Register (A8_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	R	R	R	0	R	R	R	R	R	R	R	R	R	R	R
STR	IDLE	WHE	CHE		TRP	T13	T13	T12	T12	CC	CC	CC	CC	CC	CC
					F	PM	CM	PM	OM	62F	62R	61F	61R	60F	60R
w	w	w	w	r	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RCC60R, RCC61R, RCC62R	0, 2, 4	w	Reset Capture, Compare-Match Rising Edge Flag 0 _B No action 1 _B Bit CC6xR will be cleared.
RCC60F, RCC61F, RCC62F	1, 3, 5	w	Reset Capture, Compare-Match Falling Edge Flag 0 _B No action 1 _B Bit CC6xF will be cleared.
RT12OM	6	w	Reset Timer T12 One-Match Flag 0 _B No action 1 _B Bit T12OM will be cleared.
RT12PM	7	w	Reset Timer T12 Period-Match Flag 0 _B No action 1 _B Bit T12PM IS will be cleared.
RT13CM	8	w	Reset Timer T13 Compare-Match Flag 0 _B No action 1 _B Bit T13CM will be cleared.
RT13PM	9	w	Reset Timer T13 Period-Match Flag 0 _B No action 1 _B Bit T13PM will be cleared.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
RTRPF	10	w	Reset Trap Flag 0 _B No action 1 _B Bit TRPF will be cleared (not taken into account while input $\overline{\text{CTRAP}}=0$ and $\text{TRPPEN}=1$.
RCHE	12	w	Reset Correct Hall Event Flag 0 _B No action 1 _B Bit CHE will be cleared.
RWHE	13	w	Reset Wrong Hall Event Flag 1 _B No action 0 _B Bit WHE will be cleared.
RIDLE	14	w	Reset IDLE Flag 0 _B No action 1 _B Bit IDLE will be cleared.
RSTR	15	w	Reset STR Flag 0 _B No action 1 _B Bit STR will be cleared.
0	11, [31:16]	r	Reserved; Returns 0 if read; should be written with 0.

27.9.2.4 Interrupt Enable Register

Register IEN contains the interrupt enable bits and a control bit to enable the automatic idle function in the case of a wrong hall pattern.

IEN
Interrupt Enable Register
(B0_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN STR	EN IDLE	EN WHE	EN CHE	0	EN TRP F	EN T13 PM	EN T13 CM	EN T12 PM	EN T12 OM	EN CC 62F	EN CC 62R	EN CC 61F	EN CC 61R	EN CC 60F	EN CC 60R
rw	rw	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENCC60R, ENCC61R, ENCC62R	0, 2, 4	rw	Capture, Compare-Match Rising Edge Interrupt Enable for Channel CC6x 0 _B No interrupt will be generated if the set condition for bit CC6xR in register IS occurs. 1 _B An interrupt will be generated if the set condition for bit CC6xR in register IS occurs. The service request output that will be activated is selected by bit field INPCC6x.
ENCC60F, ENCC61F, ENCC62F	1, 3, 5	rw	Capture, Compare-Match Falling Edge Interrupt Enable for Channel CC6x 0 _B No interrupt will be generated if the set condition for bit CC6xF in register IS occurs. 1 _B An interrupt will be generated if the set condition for bit CC6xF in register IS occurs. The service request output that will be activated is selected by bit field INPCC6x.

Capture/Compare Unit 6 (CCU6)

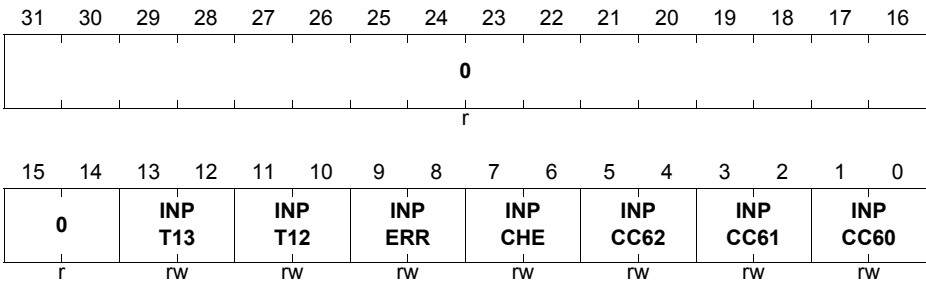
Field	Bits	Type	Description
ENT12OM	6	rw	<p>Enable Interrupt for T12 One-Match</p> <p>0_B No interrupt will be generated if the set condition for bit T12OM in register IS occurs.</p> <p>1_B An interrupt will be generated if the set condition for bit T12OM in register IS occurs. The service request output that will be activated is selected by bit field INPT12.</p>
ENT12PM	7	rw	<p>Enable Interrupt for T12 Period-Match</p> <p>0_B No interrupt will be generated if the set condition for bit T12PM in register IS occurs.</p> <p>1_B An interrupt will be generated if the set condition for bit T12PM in register IS occurs. The service request output that will be activated is selected by bit field INPT12.</p>
ENT13CM	8	rw	<p>Enable Interrupt for T13 Compare-Match</p> <p>0_B No interrupt will be generated if the set condition for bit T13CM in register IS occurs.</p> <p>1_B An interrupt will be generated if the set condition for bit T13CM in register IS occurs. The service request output that will be activated is selected by bit field INPT13.</p>
ENT13PM	9	rw	<p>Enable Interrupt for T13 Period-Match</p> <p>0_B No interrupt will be generated if the set condition for bit T13PM in register IS occurs.</p> <p>1_B An interrupt will be generated if the set condition for bit T13PM in register IS occurs. The service request output that will be activated is selected by bit field INPT13.</p>
ENTRPF	10	rw	<p>Enable Interrupt for Trap Flag</p> <p>0_B No interrupt will be generated if the set condition for bit TRPF in register IS occurs.</p> <p>1_B An interrupt will be generated if the set condition for bit TRPF in register IS occurs. The service request output that will be activated is selected by bit field INPERR.</p>

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
ENCHE	12	rw	Enable Interrupt for Correct Hall Event 0_B No interrupt will be generated if the set condition for bit CHE in register IS occurs. 1_B An interrupt will be generated if the set condition for bit CHE in register IS occurs. The service request output that will be activated is selected by bit field INPCHE.
ENWHE	13	rw	Enable Interrupt for Wrong Hall Event 0_B No interrupt will be generated if the set condition for bit WHE in register IS occurs. 1_B An interrupt will be generated if the set condition for bit WHE in register IS occurs. The service request output that will be activated is selected by bit field INPERR.
ENIDLE	14	rw	Enable Idle This bit enables the automatic entering of the idle state (bit IDLE will be set) after a wrong hall event has been detected (bit WHE is set). During the idle state, the bit field MCMP is automatically cleared. 0_B The bit IDLE is not automatically set when a wrong hall event is detected. 1_B The bit IDLE is automatically set when a wrong hall event is detected.
ENSTR	15	rw	Enable Multi-Channel Mode Shadow Transfer Interrupt 0_B No interrupt will be generated if the set condition for bit STR in register IS occurs. 1_B An interrupt will be generated if the set condition for bit STR in register IS occurs. The service request output that will be activated is selected by bit field INPCHE.
0	11, [31:16]	r	Reserved; Returns 0 if read; should be written with 0.

27.9.2.5 Interrupt Node Pointer Register

Register INP contains the interrupt node pointers allowing a flexible interrupt handling. These bit fields define which service request output will be activated if the corresponding interrupt event occurs and the interrupt generation for this event is enabled.

Capture/Compare Unit 6 (CCU6)
INP
Interrupt Node Pointer Register
(AC_H)
Reset Value: 0000 3940_H


Field	Bits	Type	Description
INPCC60, INPCC61, INPCC62	[1:0], [3:2], [5:4]	rw	Interrupt Node Pointer for Channel CC6x Interrupts This bit field defines the service request output activated due to a set condition for bit CC6xR (if enabled by bit ENCC6xR) or for bit CC6xF (if enabled by bit ENCC6xF). 00 _B Service request output SR0 is selected. 01 _B Service request output SR1 is selected. 10 _B Service request output SR2 is selected. 11 _B Service request output SR3 is selected.
INPCHE	[7:6]	rw	Interrupt Node Pointer for the CHE Interrupt This bit field defines the service request output activated due to a set condition for bit CHE (if enabled by bit ENCHE) or for bit STR (if enabled by bit ENSTR). Coding see INPCC6x.
INPERR	[9:8]	rw	Interrupt Node Pointer for Error Interrupts This bit field defines the service request output activated due to a set condition for bit TRPF (if enabled by bit ENTRPF) or for bit WHE (if enabled by bit ENWHE). Coding see INPCC6x.
INPT12	[11:10]	rw	Interrupt Node Pointer for Timer12 Interrupts This bit field defines the service request output activated due to a set condition for bit T12OM (if enabled by bit ENT12OM) or for bit T12PM (if enabled by bit ENT12PM). Coding see INPCC6x.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
INPT13	[13:12]	rw	Interrupt Node Pointer for Timer13 Interrupt This bit field defines the service request output activated due to a set condition for bit T13CM (if enabled by bit ENT13CM) or for bit T13PM (if enabled by bit ENT13PM). Coding see INPCC6x.
0	[31:14]	r	Reserved; Returns 0 if read; should be written with 0.

27.10 General Module Operation

This section provides information about the:

- Input Selection (see [Section 27.10.1](#))
- Input Monitoring (see [Section 27.10.2](#))
- OCDS Suspend Functionality (see [Section 27.10.3](#))
- OCDS Trigger Bus (OTGB) Interface (see [Section 27.10.4](#))
- General Register Description (see [Section 27.10.5](#))
- BPI Register Description (see [Section 27.10.6](#))

27.10.1 Input Selection

Each CCU6 input signal can be selected from a vector of four or eight possible inputs by programming the port input select registers [PISEL0](#) and [PISEL2](#). This permits to adapt the pin functionality of the device to the application requirements.

The output pins for the module output signals are chosen in the ports.

Naming convention:

The input vector CC60IN[D:A] for input signal CC60IN is composed of the signals CC60INA to CC60IND.

Note: All functional inputs of the CCU6 are synchronized to f_{CC6} before they affect the module internal logic. The resulting delay of $2/f_{CC6}$ and for asynchronous signals an additional uncertainty of $1/f_{CC6}$ have to be taken into account for precise timing calculation. An edge of an input signal can only be correctly detected if the high phase and the low phase of the input signal are both longer than $1/f_{CC6}$.

27.10.2 Input Monitoring

A selected event which occurs at each CCU6 input signal can be monitored through **IMON.x**. Every input signal can be included for the detection of a lost bit event (**IMON.LBE**) if enabled through its individual lost indicator enable bits (**LI.yEN**). The lost bit event occurs if a selected event occurs again with the previous event captured (**IMON.x** remains set) and its lost indicator is enabled for at least one of the monitored input signals. The lost bit event can be enabled (**LI.LBEEN**) for an interrupt to be generated at one of the SRx line, selected through **LI.INPLBE**. The LBE output signal of the kernel can be connected to a capture input to indicate when does the lost bit event happens.

The lost bit event can be used as a kind of interrupt or event watchdog to monitor if an action related to an event has been processed before a second event of the same type occurs. Like this, if a certain event is treated by an interrupt that should be monitored, the related indication flag has to be cleared by SW. If the SW has not yet cleared the flag and the event occurs again, the event is considered as being lost and another interrupt can be generated to inform the system about the loss. This can be also used to indicate that input events occur too often and the main task has not enough time to treat them.

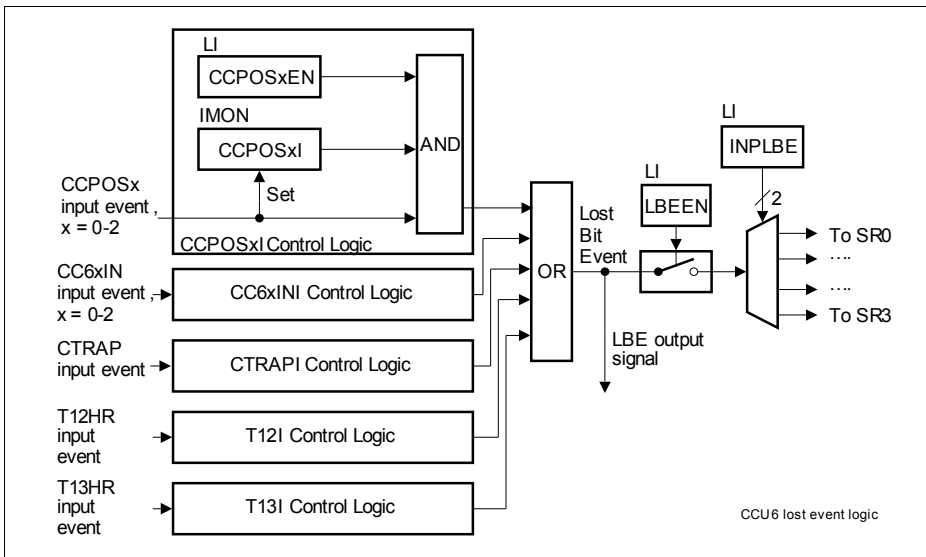


Figure 27-44 Lost Event Logic

27.10.3 OCDS Suspend

The behavior of CCU6 upon an OCDS suspend request is controlled by the **OCS** register. CCU6 supports both Hard Suspend Mode and Soft Suspend Mode.

Hard Suspend Mode

In Hard Suspend Mode the CCU6 kernel clock is switched off immediately. Reading and writing of registers is possible but will enable the kernel clock for a few cycles.

Attention: Register accesses with clocking in Hard Suspend Mode can have unintended side effects like signals becoming and staying active. This can affect also other modules, so a CCU6 kernel reset might not be sufficient to bring the system into a defined state.

Soft Suspend Mode

In Soft Suspend Mode CCU6 may finalize specific actions before it enters the suspended state with **OCS.SUSSTA** set. This can be advantageous for applications (e.g. motor control) where critical system states could occur if the kernel clock would be switched off immediately upon an OCDS suspend request.

Soft Suspend mode does not influence the kernel clock. Reading and writing of registers is possible without the side effects that may occur when reading/writing registers in Hard Suspend Mode.

For CCU6, two basic soft suspend options (**Stop Mode 0**, **Stop Mode 1**) are available, selected via bit field **OCS.SUS**.

In addition, the internal functional blocks (T12, T13, Hall logic, Trap logic) may individually be programmed via their Sensitivity Bits in register **KSCSR** to accept or ignore a suspend request. If the request sensitivity is disabled, the block continues normal operation. If the request sensitivity is enabled, the block operates as specified for the selected stop mode. The mapping of the CCU6 functional blocks to their Sensitivity Bits is shown in **Table 27-12**.

Stop Mode 0

In Stop Mode 0, if selected to be stopped, the Hall and Trap logic stops immediately, while Timer T12 and/or T13 continues normal operation (if running) until it reaches the end of the PWM period; then it stops (same stop condition as in single shot mode). When Timer T12 stops, the capture inputs CC6xIN are frozen.

Stop Mode 1

In Stop Mode 1, if selected to be stopped, the internal functional blocks (T12, T13, Hall logic, Trap logic) will stop immediately upon a suspend request.

Capture/Compare Unit 6 (CCU6)

If the sensitivity bits for Timer T12 or T13 are set, the corresponding output lines enabled for the trap condition are set to their passive values (similar to a trap condition).

Table 27-12 summarizes the reaction of the CCU6 functional blocks to OCDS suspend requests.

Table 27-12 CCU6 Functional Blocks

Block	Reaction to OCDS Suspend Request	Sensitivity Bit
0	Timer T12: if bit SB0 = 1 _B , - in Stop Mode 0 Timer T12 continues (if running) until the end of the PWM period. Then it stops and the CCxIN input stages are frozen. - in Stop Mode 1 Timer T12 stops immediately and the CC6xIN input stages are frozen. Output lines CC6x, COU6x with enabled trap functionality are set to their passive states.	KSCSR.SB0
1	Timer T13: if bit SB1 = 1 _B , - in Stop Mode 0 Timer T13 continues (if running) until the end of the PWM period. Then it stops. - in Stop Mode 1 Timer T13 stops immediately. If trap functionality for COU63 is enabled, it is set to the passive state.	KSCSR.SB1
2	Hall Logic: if bit SB2 = 1 _B , the hall logic is stopped immediately and the CCPOSx input stages are frozen (same behavior for Stop Mode 0 and 1).	KSCSR.SB2
3	Trap Logic: if bit SB3 = 1 _B , the trap logic is stopped immediately and the CTRAP input stage is frozen (same behavior for Stop Mode 0 and 1).	KSCSR.SB3

27.10.4 OCDS Trigger Bus (OTGB) Interface

The CCU6 kernel provides a set of 16 internal status signals that are combined to a Trigger Set. The CCU6 Trigger Set is shown in [Table 27-13](#). It is output on OTGB0 or OTGB1 controlled by the [OCS](#) register.

Table 27-13 TS16_CCU6 Trigger Set CCU6

Bits	Name	Description
0	T12pre	T12 prescaler output
1	T13pre	T13 prescaler output
2	T12_cm0	T12 compare match channel 0
3	T12_cm1	T12 compare match channel 1
4	T12_cm2	T12 compare match channel 2
5	T12_pm	T12 period match
6	T12_om	T12 one match
7	T12_cdir	T12 count direction
8	T12_zm	T12 zero match
9	T13_cm	T13 compare match
10	T13_pm	T13 period match
11	T13_zm	T13 zero match
12	mcm_st	MCM shadow transfer
13	che	Correct Hall event
14	whe	Wrong Hall event
15	trpf	Trap flag

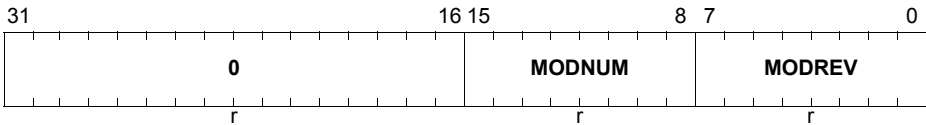
27.10.5 General Registers

27.10.5.1 ID Register

The CCU6 Module Identification Register ID contains read-only information about the module identification number and its revision.

ID

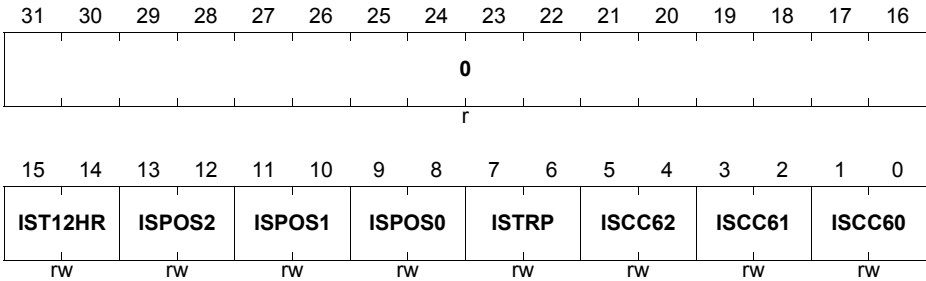
Module Identification Register (08_H) **Reset Value: 0000 54XX_H**



Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number MODREV defines the module revision number. The value of a module revision starts with 01 _H (first revision), 02 _H , 03 _H , ... up to FF _H .
MODNUM	[15:8]	r	Module Number Value This bit field defines the module identification number for the CCU6: 54 _H
0	[31:16]	r	Reserved Read as 0.

27.10.5.2 Port Input Select Registers

Registers PISEL0 and PISEL2 contain bit fields selecting the actual input signal for the module inputs.

PISEL0
Port Input Select Register 0
(10_H)
Reset Value: 0000 0000_H


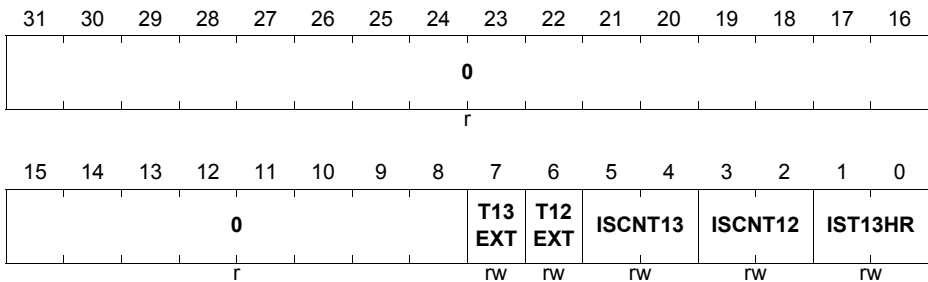
Field	Bits	Type	Description
ISCC60	[1:0]	rw	Input Select for CC60 This bit field defines the input signal used as CC60 capture input. 00 _B The signal CC60INA is selected. 01 _B The signal CC60INB is selected. 10 _B The signal CC60INC is selected. 11 _B The signal CC60IND is selected.
ISCC61	[3:2]	rw	Input Select for CC61 This bit field defines the input signal used as CC61 capture input. 00 _B The signal CC61INA is selected. 01 _B The signal CC61INB is selected. 10 _B The signal CC61INC is selected. 11 _B The signal CC61IND is selected.
ISCC62	[5:4]	rw	Input Select for CC62 This bit field defines the input signal used as CC62 capture input. 00 _B The signal CC62INA is selected. 01 _B The signal CC62INB is selected. 10 _B The signal CC62INC is selected. 11 _B The signal CC62IND is selected.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
ISTRP	[7:6]	rw	Input Select for CTRAP This bit field defines the input signal used as CTRAP input. 00 _B The signal CTRAPA is selected. 01 _B The signal CTRAPB is selected. 10 _B The signal CTRAPC is selected. 11 _B The signal CTRAPD is selected.
ISPOS0	[9:8]	rw	Input Select for CCPOS0 This bit field defines the input signal used as CCPOS0 input. 00 _B The signal CCPOS0A is selected. 01 _B The signal CCPOS0B is selected. 10 _B The signal CCPOS0C is selected. 11 _B The signal CCPOS0D is selected.
ISPOS1	[11:10]	rw	Input Select for CCPOS1 This bit field defines the input signal used as CCPOS1 input. 00 _B The signal CCPOS1A is selected. 01 _B The signal CCPOS1B is selected. 10 _B The signal CCPOS1C is selected. 11 _B The signal CCPOS1D is selected.
ISPOS2	[13:12]	rw	Input Select for CCPOS2 This bit field defines the input signal used as CCPOS2 input. 00 _B The signal CCPOS2A is selected. 01 _B The signal CCPOS2B is selected. 10 _B The signal CCPOS2C is selected. 11 _B The signal CCPOS2D is selected.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
IST12HR	[15:14]	rw	Input Select for T12HR This bit field defines the input signal used as T12HR input. 00 _B Either signal T12HRA (if T12EXT = 0) or T12HRE (if T12EXT = 1) is selected. 01 _B Either signal T12HRB (if T12EXT = 0) or T12HRF (if T12EXT = 1) is selected. 10 _B Either signal T12HRC (if T12EXT = 0) or T12HRG (if T12EXT = 1) is selected. 11 _B Either signal T12HRD (if T12EXT = 0) or T12HRH (if T12EXT = 1) is selected.
0	[31:16]	r	Reserved Returns 0 if read, should be written with 0.

PISEL2
Port Input Select Register 2
(14_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
IST13HR	[1:0]	rw	<p>Input Select for T13HR</p> <p>This bit field defines the input signal used as T13HR input.</p> <p>00_B Either signal T13HRA (if T13EXT = 0) or T13HRE (if T13EXT = 1) is selected.</p> <p>01_B Either signal T13HRB (if T13EXT = 0) or T13HRF (if T13EXT = 1) is selected.</p> <p>10_B Either signal T13HRC (if T13EXT = 0) or T13HRG (if T13EXT = 1) is selected.</p> <p>11_B Either signal T13HRD (if T13EXT = 0) or T13HRH (if T13EXT = 1) is selected.</p>
ISCNT12	[3:2]	rw	<p>Input Select for T12 Counting Input</p> <p>This bit field defines the input event leading to a counting action of T12.</p> <p>00_B The T12 prescaler generates the counting events. Bit TCTR4.T12CNT is not taken into account.</p> <p>01_B Bit TCTR4.T12CNT written with 1 is a counting event. The T12 prescaler is not taken into account.</p> <p>10_B The timer T12 is counting each rising edge detected in the selected T12HR signal.</p> <p>11_B The timer T12 is counting each falling edge detected in the selected T12HR signal.</p>
ISCNT13	[5:4]	rw	<p>Input Select for T13 Counting Input</p> <p>This bit field defines the input event leading to a counting action of T13.</p> <p>00_B The T13 prescaler generates the counting events. Bit TCTR4.T13CNT is not taken into account.</p> <p>01_B Bit TCTR4.T13CNT written with 1 is a counting event. The T13 prescaler is not taken into account.</p> <p>10_B The timer T13 is counting each rising edge detected in the selected T13HR signal.</p> <p>11_B The timer T13 is counting each falling edge detected in the selected T13HR signal.</p>
T12EXT	6	rw	<p>Extension for T12HR Inputs</p> <p>This bit extends the 2-bit field IST12HR.</p> <p>0_B One of the signals T12HR[D:A] is selected.</p> <p>1_B One of the signals T12HR[H:E] is selected.</p>

Capture/Compare Unit 6 (CCU6)

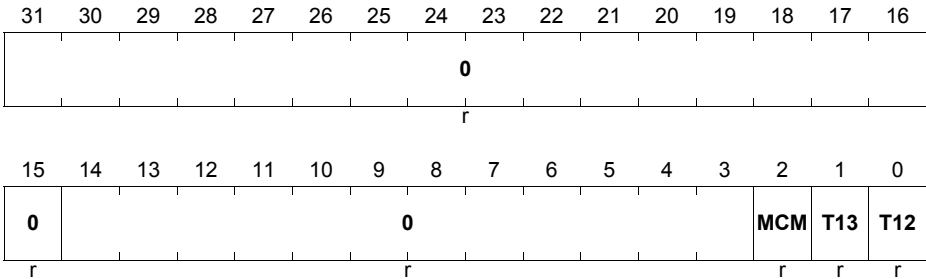
Field	Bits	Type	Description
T13EXT	7	rw	Extension for T13HR Inputs This bit extends the 2-bit field IST13HR. 0 _B One of the signals T13HR[D:A] is selected. 1 _B One of the signals T13HR[H:E] is selected.
0	[31:8]	r	Reserved; Returns 0 if read; should be written with 0.

27.10.5.3 Module Configuration Register

The module configuration register contains bits describing the functionality that is available in the CCU6 module.

MCFG

Module Configuration Register (04_H) **Reset Value: 0000 0007_H**



Field	Bits	Type	Description
T12	0	r	T12 Available This bit indicates if the T12 block is available. 0 _B The T12 block is not available. A write access to T12PR is ignored. 1 _B The T12 block is available. A write access to T12PR is executed.
T13	1	r	T13 Available This bit indicates if the T13 block is available. 0 _B The T13 block is not available. A write access to T13PR is ignored. 1 _B The T13 block is available. A write access to T13PR is executed.
MCM	2	r	Multi-Channel Mode Available This bit indicates if the multi-channel mode functionality is available. 0 _B The multi-channel mode functionality is not available. A write access to MCMOUTS is ignored. 1 _B The multi-channel mode functionality is available. A write access to MCMOUTS is executed.
0	[31:3]	r	Reserved; read as 0; should be written with 0.

27.10.5.4 Input Monitoring Register

The input monitoring register monitors the occurrence of a selected event for the input signals. If a hardware event triggers the setting of bit IMON.x and the same bit is written with 1 via software at the same time, then the corresponding bit is cleared (software overrules hardware). The lost bit event is indicated if an event is detected again at one or more input signals with its lost indicator enabled.

Note: The register is only applicable in capture modes if the edges are selected through T12MSEL.MSEL6x.

IMON

Input Monitoring Register

(98_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			T13 HRI	T12 HRI	CTR API	CC 62INI	CC 61INI	CC 60INI	CC POS 2I	CC POS 1I	CC POS 0I	LBE			
			r	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Capture/Compare Unit 6 (CCU6)

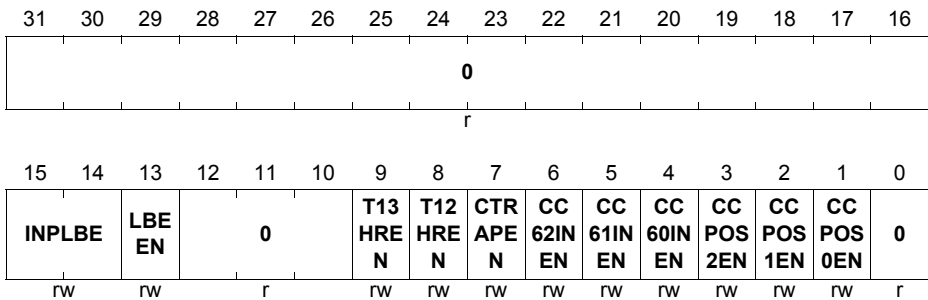
Field	Bits	Type	Description
LBE	0	rwh	<p>Lost Bit Event</p> <p>This bit determines if a lost bit event has occurred. A lost bit event occurs when a selected event occurs again with the previous event captured (IMON.x remains set) and its lost indicator is enabled, for at least one of the monitored input signals. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect.</p> <p>0_B The lost bit event has not occurred. 1_B The lost bit event has occurred.</p>
CCPOSxI (x = 0 -2)	x + 1	rwh	<p>Event indication for input signal CCPOSx</p> <p>The bit determines if the selected event has occurred via an edge detection. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect.</p> <p>0_B A selected event has not occurred. 1_B Edge detection indicates a selected event has occurred.</p> <p><i>Note: The dedicated edge is indicated for a selected event if Hysteretic-like Control or Capture modes are initialized in T12MSEL.MSEL6x. If these modes are not selected, then all edges will be indicated as an event for the inputs.</i></p>
CC6xINI (x = 0 -2)	x + 4	rwh	<p>Event indication for input signal CC6xIN</p> <p>The bit determines if the selected event has occurred via an edge detection. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect.</p> <p>0_B A selected event has not occurred. 1_B Edge detection indicates a selected event has occurred.</p>
CTRAPI	7	rwh	<p>Event indication for input signal CTRAP</p> <p>The bit determines if the selected event has occurred via an edge detection. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect.</p> <p>0_B An event has not occurred. 1_B Edge detection indicates an event has occurred.</p>

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
T12HRI	8	rwh	Event indication for input signal T12HR The bit determines if the selected event has occurred via an edge detection. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect. 0 _B An event has not occurred. 1 _B Edge detection indicates an event has occurred.
T13HRI	9	rwh	Event indication for input signal T13HR The bit determines if the selected event has occurred via an edge detection. The bit can be cleared by writing a 1 to the same bit position, while writing a 0 has no effect. 0 _B An event has not occurred. 1 _B Edge detection indicates an event has occurred.
0	[31:10]	r	Reserved; Returns 0 if read; should be written with 0.

27.10.5.5 Lost Indicator Register

The lost indicator register has the lost indicator enable bits for its detected event at the input signals. The lost bit event can then be enabled as an output signal through one of the service request lines.

LI
Lost Indicator Register (9C_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
CCPOSxEN (x = 0 -2)	x + 1	rw	Lost Indicator Enable for input signal CCPOSx This bit determines if the monitored event at the input signal is enabled for the detection of a lost bit event. 0 _B Input signal is disabled for a lost bit event detection. 1 _B Input signal is enabled for a lost bit event detection.
CC6xINEN (x = 0 -2)	x + 4	rw	Lost Indicator Enable for input signal CC6xIN This bit determines if the monitored event at the input signal is enabled for the detection of a lost bit event. 0 _B Input signal is disabled for a lost bit event detection. 1 _B Input signal is enabled for a lost bit event detection.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
CTRAPEN	7	rw	Lost Indicator Enable for input signal CTRAP This bit determines if the monitored event at the input signal is enabled for the detection of a lost bit event. 0 _B Input signal is disabled for a lost bit event detection. 1 _B Input signal is enabled for a lost bit event detection.
T12HREN	8	rw	Lost Indicator Enable for input signal T12HR This bit determines if the monitored event at the input signal is enabled for the detection of a lost bit event. 0 _B Input signal is disabled for a lost bit event detection. 1 _B Input signal is enabled for a lost bit event detection.
T13HREN	9	rw	Lost Indicator Enable for input signal T13HR This bit determines if the monitored event at the input signal is enabled for the detection of a lost bit event. 0 _B Input signal is disabled for a lost bit event detection. 1 _B Input signal is enabled for a lost bit event detection.
LBEEN	13	rw	Interrupt Enable for Lost Bit Event This bit determines if a SRx line is activated if lost bit event is detected. 0 _B Lost bit event is disabled for the activation of a SRx line. 1 _B Lost bit event is enabled for the activation of a SRx line.
INPLBE	[15:14]	rw	Interrupt Node Pointer for lost bit event This bit field defines which service request output line is selected to output an lost event alert for an enabled lost bit event. 00 _B Service request output SR0 is selected. 01 _B Service request output SR1 is selected. 10 _B Service request output SR2 is selected. 11 _B Service request output SR3 is selected.
0	0, [12:10], [31:16]	r	Reserved; Returns 0 if read; should be written with 0.

27.10.5.6 Kernel State Control Sensitivity Register

The kernel state control sensitivity register bits define which internal block is affected by Stop Modes 0 and 1, as described in section [Soft Suspend Mode](#).

The Kernel State Control Sensitivity Register (KSCSR) is cleared by Debug Reset.

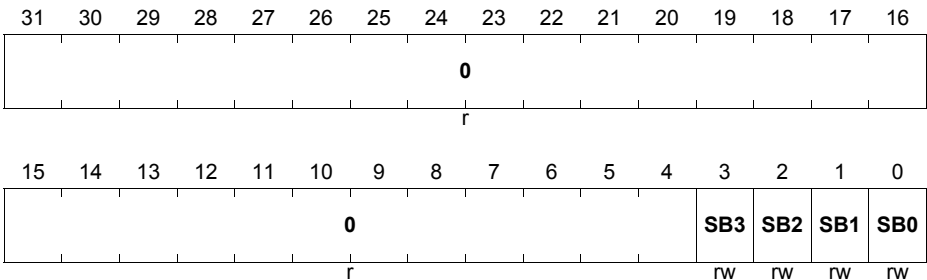
The register can only be written while the OCDS is enabled (OCDS enable = '1').

KSCSR

Kernel State Control Sensitivity Register

(1C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SB0, SB1, SB2, SB3	0, 1, 2, 3	rw	Sensitivity Block x This bit defines if block x of the CCU6 kernel is sensitive to Stop Mode 0 or Stop Mode 1. The functional definition of the blocks is given in Table 27-12 . 0 _B Block x is not sensitive to Stop Mode 0 or Stop Mode 1. It continues normal operation without respecting the defined stop condition. 1 _B Block x is sensitive to Stop Mode 0 or Stop Mode 1. It is respecting the defined stop condition.
0	[31:4]	r	Reserved; Returns 0 if read; should be written with 0.

27.10.6 BPI Registers

This section describes the registers of the BPI (Bus Peripheral Interface). [Figure 27-45](#) shows all registers associated with the BPI for one CCU6 kernel.

BPI Registers Overview

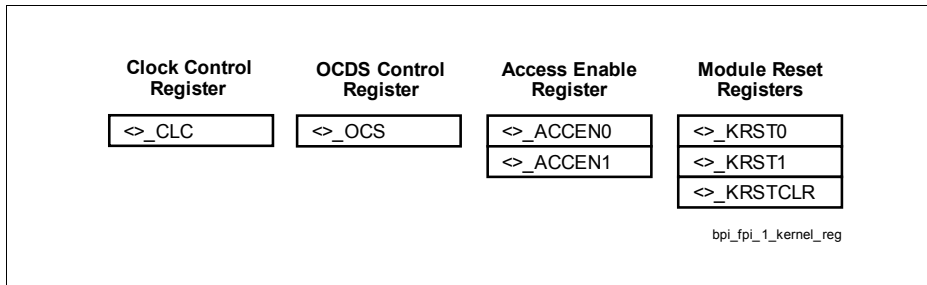


Figure 27-45 BPI Registers

[Table 27-14](#) gives an overview of the BPI registers, including their access modes and reset class.

Table 27-14 Registers Overview - BPI Registers

Register Short Name	Description	Offset Addr.	Access Mode		Reset	Page Num.
			Read	Write		
CLC	Clock Control Register	00 _H	U, SV	SV, E, P	Application Reset	27-130
-	Kernel Registers	-	-	-	-	-
OCS	OCDS Control and Status Register	E8 _H	U, SV	SV, P	Debug Reset	27-131
KRSTCLR	Reset Status Clear Register	EC _H	U, SV	SV, E, P	Application Reset	27-137
KRST1	Reset Control Register 1	F0 _H	U, SV	SV, E, P	Application Reset	27-136
KRST0	Reset Control Register 0	F4 _H	U, SV	SV, E, P	Application Reset	27-135

Table 27-14 Registers Overview - BPI Registers

Register Short Name	Description	Offset Addr.	Access Mode		Reset	Page Num.
			Read	Write		
ACCEN1	Access Enable Register 1	F8 _H	U, SV	SV, SE	Application Reset	27-134
ACCEN0	Access Enable Register 0	FC _H	U, SV	SV, SE	Application Reset	27-133

Note: Register bits marked "r" in the following register description are virtual registers and do not contain flip-flops. They are always read as 0.

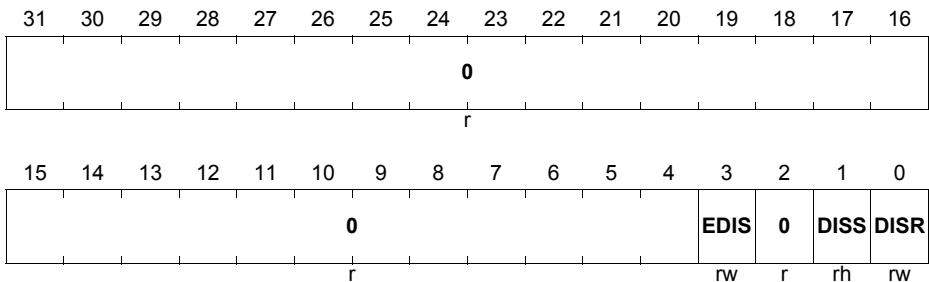
27.10.6.1 System Registers

Clock Control Register (CLC)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI for the CCU6. Where a module kernel is connected to the CLC clock control interface, CLC controls the f_{CCU6} module clock signal and Sleep Mode for the module.

CLC

Clock Control Register

(00_H)
Reset Value: 0000 0003_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module. 0 _B Module disable is not requested. 1 _B Module disable is requested.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module. 0 _B Module is enabled. 1 _B Module is disabled.
EDIS	3	rw	Sleep Mode Enable Control Used to control module's sleep mode. 0 _B Sleep Mode request is regarded. Module is enabled to go into Sleep Mode. 1 _B Sleep Mode request is disregarded: Sleep Mode cannot be entered upon a request.

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
0	[31:16], [15:4], 2	r	Reserved Read as 0; should be written with 0.

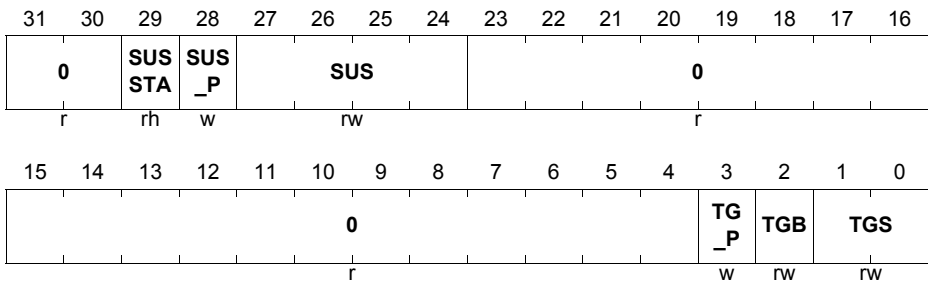
Note: Upon an accepted Sleep Mode request (with EDIS = '1'), or upon a disable request (DISR = '1'), the CCU6 kernel clock is switched off immediately. Therefore, software should ensure that the system controlled by the CCU6 kernel has reached a safe state before triggering a Sleep Mode or module disable request.

OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The OCS register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS register value will not change. When OCDS is disabled the OCS suspend control is ineffective.

Write access is 32 bit wide only and requires Supervisor Mode.

OCS
OCDS Control and Status Register (E8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
TGS	[1:0]	rw	Trigger Set for OTGB0/1 0 _H No Trigger Set output 1 _H Trigger Set TS16_CCU6 (Table 27-13) others, reserved
TGB	2	rw	OTGB0/1 Bus Select 0 _B Trigger Set is output on OTGB0 1 _B Trigger Set is output on OTGB1

Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
TG_P	3	w	TGS, TGB Write Protection TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately. 2 _H Soft suspend, Stop Mode 0 3 _H Soft suspend, Stop Mode 1 others , reserved Effects of soft suspend options on CCU6 Functional Blocks are described in section Soft Suspend Mode
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:4], [31:30]	r	Reserved Read as 0; must be written with 0.

Access Enable Register 0 (ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ..., EN31 -> TAG ID 011111_B.

1) The BPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers.

ACCEN0
Access Enable Register 0
(FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Field	Bits	Type	Description
ENn (n = 0-31)	n	r/w	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register 1 (ACCEN1)

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

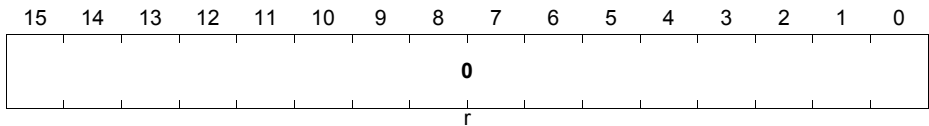
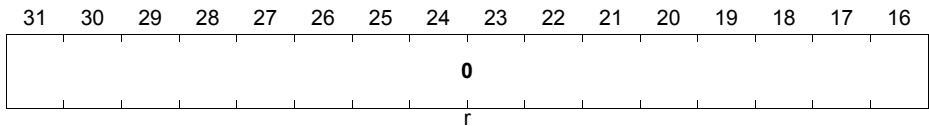
Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000_B, EN1 -> TAG ID 100001_B, ..., EN31 -> TAG ID 111111_B.

ACCEN1

Access Enable Register 1

(F8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

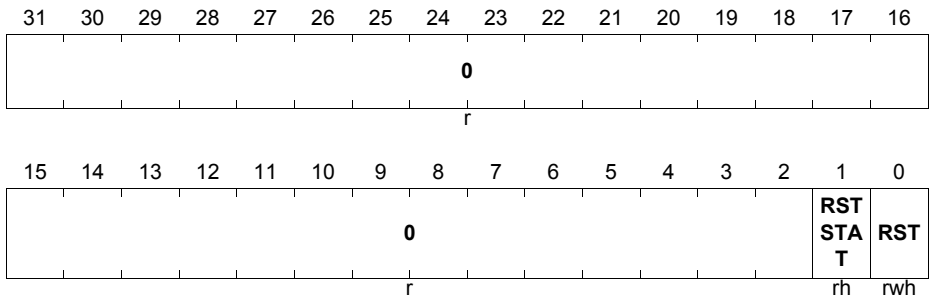
Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers related to the module kernel. The RST bit will be re-set (cleared to '0') by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI in the same clock cycle the RST bit is re-set by the BPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the CLR bit in the related KRSTCLR register.

Note: During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge

*Note: A kernel reset, initiated via registers KRST0/KRST1, does not affect the port logic. Because register **PSLR** is set to its default value, the passive level 0 is selected for the outputs CC60..CC62 and COUT60..COUT63 upon a kernel reset. Therefore, software must ensure appropriate levels for the external system at the port pins in case they are different from the default values selected via PSLR.*

KRST0
Kernel Reset Register 0
(F4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI after the kernel reset was executed.
RSTSTAT	1	rh	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI after the execution of a kernel reset in the same clock cycle both reset bits. 0 _B No kernel reset was executed 1 _B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers (KRSTx1.RST and KRSTx0.RST) related to the module kernel that should be reset. The RST bit will be re-set (cleared to '0') by the BPI with the end of the BPI kernel reset sequence.

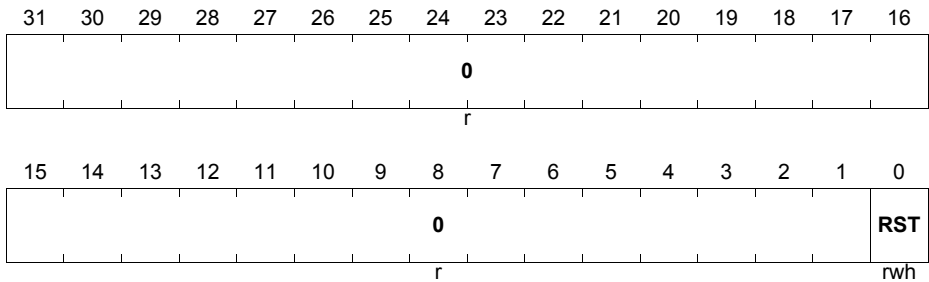
Capture/Compare Unit 6 (CCU6)

KRST1

Kernel Reset Register 1

(F0_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers are set.</p> <p>0_B No kernel reset was requested</p> <p>1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI after the kernel reset was executed.</p>
0	[31:1]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Kernel Reset Status Clear Register (KRSTCLR)

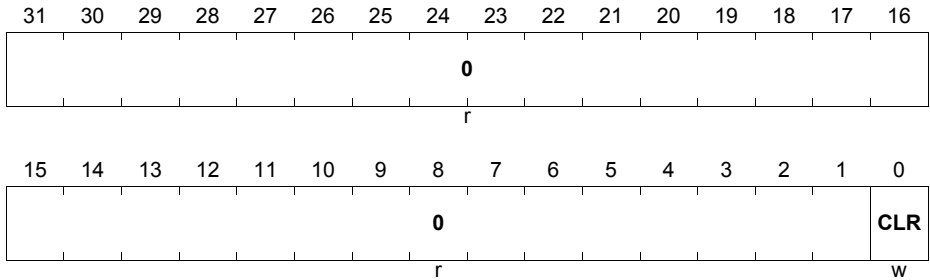
The Kernel Reset Status Clear Register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

Capture/Compare Unit 6 (CCU6)

KRSTCLR

Kernel Reset Status Clear Register (EC_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

27.11 Implementation

This chapter describes the implementation of the CCU6 module in the TC26x B-Step device.

- Address map (see [Section 27.11.1](#))
- Module output select (see [Section 27.11.2](#))
- Synchronous start (see [Section 27.11.3](#))
- Digital Connections (see [Section 27.11.4](#))

27.11.1 Address Map

There are two CCU6 kernels in the TC26x B-Step, namely CCU60 and CCU61. The CCU6061 module consists of CCU60 and CCU61 kernels .

Table 27-15 Registers Address Space

Module	Base Address	End Address	Note
CCU60	F000 2A00 _H	F000 2AFF _H	CCU6061 module includes CCU60 and CCU61 kernels
CCU61	F000 2B00 _H	F000 2BFF _H	CCU6061 module includes CCU60 and CCU61 kernels

Table 27-16 Registers Overview - CCU6 Module Registers

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Reset	Page Num.
			Read	Write		
CCU6061 Module Registers (only available in the address range of CCU60)						
CCU60_MO SEL	CCU60 Module Output Select Register	0C _H	U, SV	U, SV, P	Application Reset	27-139

27.11.1.1 Module Registers

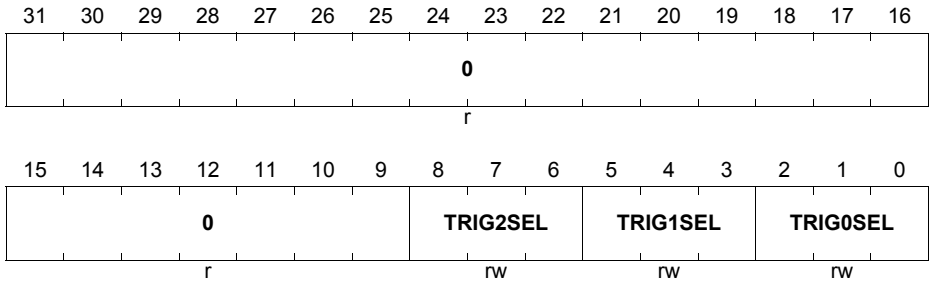
Module Output Select Register

MOSEL contains bit fields to select the output signal from module CCU6061 for the trigger signals to the A/D converters in the VADC module.

CCU60_MOSEL

CCU60 Module Output Select Register(0C_H)

Reset Value: 0000 0000_H



Capture/Compare Unit 6 (CCU6)

Field	Bits	Type	Description
TRIG0SEL	[2:0]	rw	Output Trigger Select for CCU6061 TRIG0 This bit field defines the output signal from the CCU6061 module used as the trigger signal to the VADC inputs. 000 _B The signal CCU60_COUT63 is selected. 001 _B The signal CCU61_COUT63 is selected. 010 _B The signal CCU60_CC60 is selected. 011 _B The signal CCU61_CC60 is selected. 100 _B The signal CCU60_SR1 is selected. 101 _B The signal CCU61_SR1 is selected. 110 _B The signal CCU60_SR3 is selected. 111 _B The signal CCU61_SR3 is selected.
TRIG1SEL	[5:3]	rw	Output Trigger Select for CCU6061 TRIG1 This bit field defines the output signal from the CCU6061 module used as the trigger signal to the VADC inputs. 000 _B The signal CCU60_COUT63 is selected. 001 _B The signal CCU61_COUT63 is selected. 010 _B The signal CCU60_CC61 is selected. 011 _B The signal CCU61_CC61 is selected. 100 _B The signal CCU60_SR1 is selected. 101 _B The signal CCU61_SR1 is selected. 110 _B The signal CCU60_SR3 is selected. 111 _B The signal CCU61_SR3 is selected.
TRIG2SEL	[8:6]	rw	Output Trigger Select for CCU6061 TRIG2 This bit field defines the output signal from the CCU6061 module used as the trigger signal to the VADC inputs. 000 _B The signal CCU60_COUT63 is selected. 001 _B The signal CCU61_COUT63 is selected. 010 _B The signal CCU60_CC62 is selected. 011 _B The signal CCU61_CC62 is selected. 100 _B The signal CCU60_SR1 is selected. 101 _B The signal CCU61_SR1 is selected. 110 _B The signal CCU60_SR3 is selected. 111 _B The signal CCU61_SR3 is selected.
0	[31:9]	r	Reserved Returns 0 if read, should be written with 0.

Note: CCU60_MOSEL is located in the address space of kernel CCU60. Therefore, when a reset of kernel CCU60 is triggered via registers CCU60_KRST0 and CCU60_KRST1, register CCU60_MOSEL is also reset.

Capture/Compare Unit 6 (CCU6)

*CCU60_MOSEL is **not** reset when a reset of kernel CCU61 is triggered via registers CCU61_KRST0 and CCU61_KRST1.*

*Because CCU60_MOSEL controls both signals from kernel CCU60 and CCU61, the output signals TRIG0..2 will be set to their inactive levels during **any** kernel reset of CCU60 as well as of CCU61 (see also [Figure 27-46](#)).*

Depending on the application, it may make sense to always reset both kernels in this case.

27.11.2 Module Output Select

For CCU6061 module, there are 3 trigger signals which are selectable from the output signals of the CCU60 and CCU61 kernels.

Each of the trigger signals (TRIG0, TRIG1, TRIG2) of the module is routed to each converter of the VADC module, as follows:

- TRIG0 of CCU6061 is routed to inputs BGREQGTC and GxREQGTC, $x = 0..3$
- TRIG1 of CCU6061 is routed to inputs BGREQGTD and GxREQGTD, $x = 0..3$
- TRIG2 of CCU6061 is routed to inputs BGREQGTE and GxREQGTE, $x = 0..3$

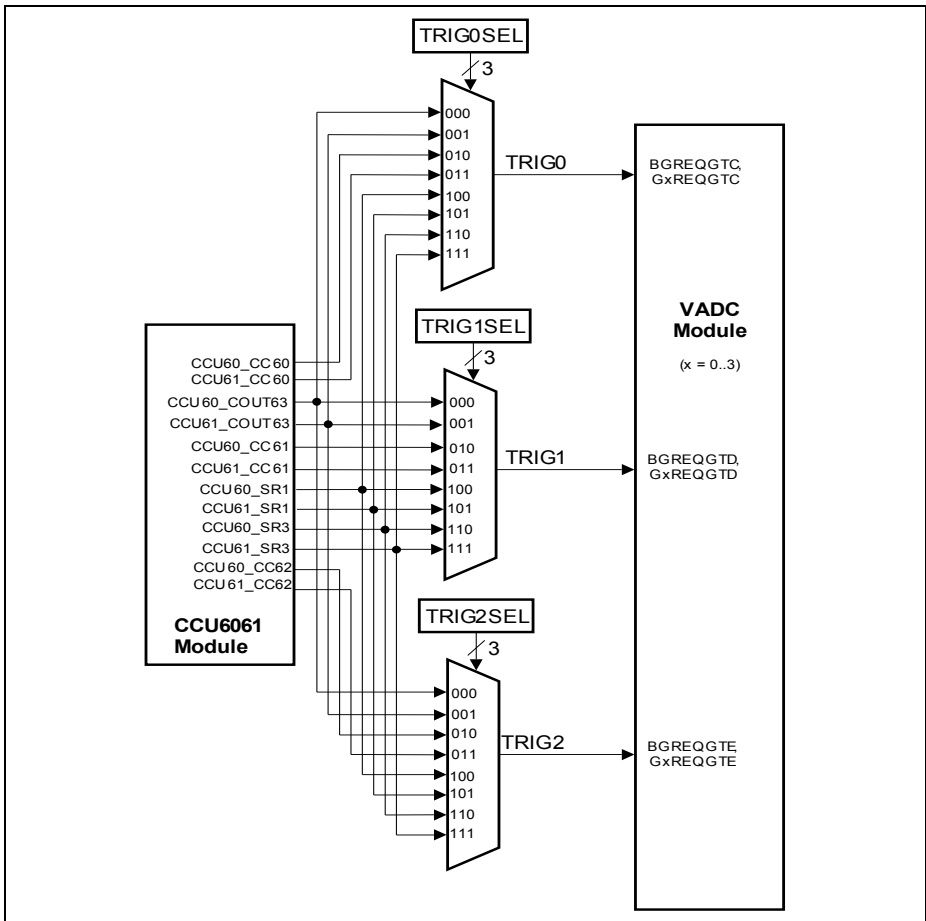


Figure 27-46 Output select trigger TC26x B-Step

27.11.3 Synchronous Start

Synchronous start of the capture/compare timers is supported by control bit `SYSCON.CCTRIG0` in the SCU module. Bit `SYSCON.CCTRIG0` is connected to the `T12HR` and `T13HR` inputs of the `CCU60` and `CCU61` kernels.

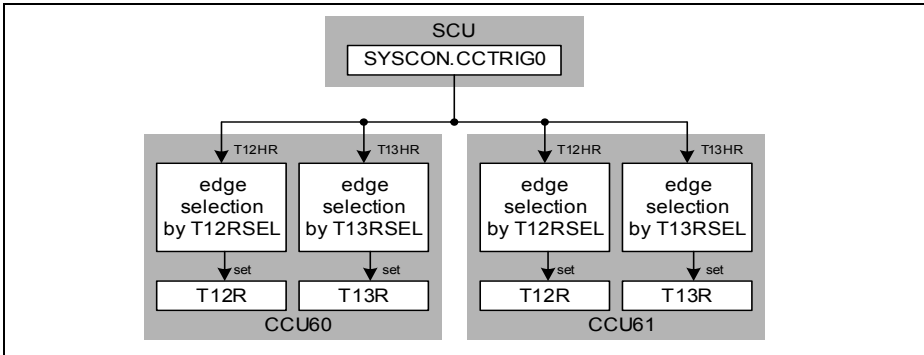


Figure 27-47 Synchronization Concept in TC26x B-Step

27.11.4 Digital Connections

The following tables show the digital connections of the CCU6x module with other modules or pins in the TC26x B-Step device.

Each input signal can be selected among 4 or 8 possible input lines, e.g. the input vector for input signal CC60IN is composed of CC60IN[D:A], whereas the input vectors for T12HR and T13HR are composed of T12HR[H:A] and T13HR[H:A]. The following sections refer to the interface signals.

The CCU6x module is clocked with the SPB_Bus clock, so $f_{CC6} = f_{SPB}$.

Note: All functional inputs of the CCU6 are synchronized to f_{CC6} before they can affect the module internal logic. The resulting delay of $2/f_{CC6}$ and an uncertainty of $1/f_{CC6}$ have to be taken into account for precise timing calculation.

An edge of an input signal can only be correctly detected if both, the high phase and the low phase of the input signal are each longer than $1/f_{CC6}$.

27.11.4.1 Connections of CCU60

This table describes the module interconnections of CCU60.

Table 27-17 CCU60 Digital Connections in TC26x B-Step

Signal	from/to Module	I/O to CCU60	Can be used to/as
CC60INA	P02.0	I	input signals for capture event on channel CC60
CC60INB	P00.1	I	
CC60INC	P02.6	I	
CC60IND	0	I	
CC61INA	P02.2	I	input signals for capture event on channel CC61
CC61INB	P00.3	I	
CC61INC	P02.7	I	
CC61IND	0	I	
CC62INA	P02.4	I	input signals for capture event on channel CC62
CC62INB	P00.5	I	
CC62INC	P02.8	I	
CC62IND	ERU_PDOUT4	I	

Capture/Compare Unit 6 (CCU6)
Table 27-17 CCU60 Digital Connections in TC26x B-Step (cont'd)

Signal	from/to Module	I/O to CCU60	Can be used to/as
CTRAPA	P00.11 ²⁾	I	input signals for CTRAP
CTRAPB	CCU60_whe_n ¹⁾	I	
CTRAPC	VADCG0BFL3	I	
CTRAPD	ERU_PDOUT0	I	
CCPOS0A	P02.6	I	input signals for CCPOS0
CCPOS0B	CCU61_SR2	I	
CCPOS0C	P10.4 ²⁾	I	
CCPOS0D	P40.0 (SENT0A)	I	
CCPOS1A	P02.7	I	input signals for CCPOS1
CCPOS1B	P40.1 (SENT1A)	I	
CCPOS1C	P10.7 ²⁾	I	
CCPOS1D	P40.2 (SENT2A)	I	
CCPOS2A	P02.8	I	input signals for CCPOS2
CCPOS2B	P40.3 (SENT3A)	I	
CCPOS2C	P10.8 ²⁾	I	
CCPOS2D	0	I	
T12HRA	SCU_CCTRIG0	I	input signals for T12HR
T12HRB	P00.7	I	
T12HRC	P00.9	I	
T12HRD	GTM_TOM0_8	I	
T12HRE	P00.0	I	
T12HRF	GPT120_T6OFL	I	
T12HRG	CCU61_SR2	I	
T12HRH	ERU_PDOUT0	I	

Capture/Compare Unit 6 (CCU6)
Table 27-17 CCU60 Digital Connections in TC26x B-Step (cont'd)

Signal	from/to Module	I/O to CCU60	Can be used to/as
T13HRA	SCU_CCTRIG0	I	input signals for T13HR
T13HRB	P00.8	I	
T13HRC	P00.9	I	
T13HRD	GTM_TOM1_8	I	
T13HRE	0	I	
T13HRF	GPT120_T6OFL	I	
T13HRG	CCU61_SR2	I	
T13HRH	CCU60_SR1	I	
CC60	P02.0 P02.6 P11.12 P15.6 (see port chapter)	O	compare outputs of channel CC60
	VADC		VADC gating input, see CCU60_MOSEL for the respective trigger select
COUT60	P02.1 P11.9 P15.7 (see port chapter)	O	
	ERU_IN01		
CC61	P02.2 P02.7 P11.11 P15.5 (see port chapter)	O	compare outputs of channel CC61
	VADC		VADC gating input, see CCU60_MOSEL for the respective trigger select
COUT61	P02.3 P11.6 P15.8 (see port chapter)	O	

Capture/Compare Unit 6 (CCU6)
Table 27-17 CCU60 Digital Connections in TC26x B-Step (cont'd)

Signal	from/to Module	I/O to CCU60	Can be used to/as
CC62	P02.4 P02.8 P11.10 P15.4 (see port chapter)	O	compare outputs of channel CC62
	VADC		VADC gating input, see CCU60_MOSEL for the respective trigger select
COUT62	P02.5 P11.3 P14.0 (see port chapter)	O	
COUT63	P00.0 P11.2 P14.1 P32.4 (see port chapter)	O	compare output of channel CC63
	VADC		VADC gating input, see CCU60_MOSEL for the respective trigger select
LBE	n.c.	O	lost bit event output
whe_n ¹⁾	CCU60_CTRAPB	O	wrong Hall event output (inverted)
OTGB0[0:15]	OCDS	O	OCDS Trigger Bus 0
OTGB1[0:15]	OCDS	O	OCDS Trigger Bus 1
SR0	Interrupt Router: SRC_CCU60SR0	O	CCU60 Service Request 0
SR1	CCU60_T13HRH	O	T13 count trigger
	VADC		VADC gating input, see CCU60_MOSEL for the respective trigger select
	Interrupt Router: SRC_CCU60SR1		CCU60 Service Request 1

Capture/Compare Unit 6 (CCU6)
Table 27-17 CCU60 Digital Connections in TC26x B-Step (cont'd)

Signal	from/to Module	I/O to CCU60	Can be used to/as
SR2	CCU61_CCPOS0B, CCU61_T12HRG, CCU61_T13HRG	O	CCU61 triggers
	Interrupt Router: SRC_CCU60SR2		CCU60 Service Request 2
SR3	VADC_GxREQTRA (x = 0..3), VADC_BGREQTRA	O	VADC trigger capability, for additional options see also CCU60_MOSEL
	Interrupt Router: SRC_CCU60SR3		CCU60 Service Request 3

1) CCU6x_whe_n is the inverted version of the internal Wrong Hall Event signal whe (to match the polarity requirements of the CTRAP input)

2) not available for devices in 144-pin package

27.11.4.2 Connections of CCU61

This table describes the module interconnections of CCU61.

Table 27-18 CCU61 Digital Connections in TC26x B-Step

Signal	from/to Module	I/O to CCU61	Can be used to/as
CC60INA	P00.1		input signals for capture event on channel CC60
CC60INB	P02.0		
CC60INC	P00.7		
CC60IND	0		
CC61INA	P00.3		input signals for capture event on channel CC61
CC61INB	P02.2		
CC61INC	P00.8		
CC61IND	0		
CC62INA	P00.5		input signals for capture event on channel CC62
CC62INB	P02.4		
CC62INC	P00.9		
CC62IND	ERU_PDOOUT5		
CTRAPA	P00.0		input signals for CTRAP
CTRAPB	CCU61_whe_n ¹⁾		
CTRAPC	P33.4		
CTRAPD	ERU_PDOOUT1		
CCPOS0A	P00.7		input signals for CCPOS0
CCPOS0B	CCU60_SR2		
CCPOS0C	P33.7		
CCPOS0D	0		
CCPOS1A	P00.8		input signals for CCPOS1
CCPOS1B	P40.6 (SENT2D)		
CCPOS1C	P33.6		
CCPOS1D	P40.7 (SENT3D)		

Capture/Compare Unit 6 (CCU6)
Table 27-18 CCU61 Digital Connections in TC26x B-Step (cont'd)

Signal	from/to Module	I/O to CCU61	Can be used to/as	
CCPOS2A	P00.9	I	input signals for CCPOS2	
CCPOS2B	P40.8 (SENT4A)	I		
CCPOS2C	P33.5	I		
CCPOS2D	P40.9 (SENT5A)	I		
T12HRA	SCU_CCTRIG0	I	input signals for T12HR	
T12HRB	P02.6	I		
T12HRC	P02.8	I		
T12HRD	GTM_ATOM0_1	I		
T12HRE	P00.11 ²⁾	I		
T12HRF	GPT120_T6OFL	I		
T12HRG	CCU60_SR2	I		
T12HRH	ERU_PDOUT1	I		
T13HRA	SCU_CCTRIG0	I		input signals for T13HR
T13HRB	P02.7	I		
T13HRC	P02.8	I		
T13HRD	GTM_TOM0_9	I		
T13HRE	0	I		
T13HRF	GPT120_T6OFL	I		
T13HRG	CCU60_SR2	I		
T13HRH	CCU61_SR1	I		
CC60	P00.1 P00.7 P20.8 P33.13 (see port chapter)	O	compare outputs of channel CC60	
	VADC		VADC gating input, see CCU60_MOSEL for the respective trigger select	
COUT60	P00.2 P20.11 P33.12 (see port chapter)	O		
	ERU_IN11			

Capture/Compare Unit 6 (CCU6)
Table 27-18 CCU61 Digital Connections in TC26x B-Step (cont'd)

Signal	from/to Module	I/O to CCU61	Can be used to/as
CC61	P00.3 P00.8 P20.9 P33.11 (see port chapter)	O	compare outputs of channel CC61
	VADC		VADC gating input, see CCU60_MOSEL for the respective trigger select
COUT61	P00.4 P20.12 P33.10 (see port chapter)	O	
CC62	P00.5 P00.9 P20.10 P33.9 (see port chapter)	O	compare outputs of channel CC62
	VADC		VADC gating input, see CCU60_MOSEL for the respective trigger select
COUT62	P00.6 P20.13 P33.8 (see port chapter)	O	
COUT63	P00.10 P00.12 ²⁾ P20.7 (see port chapter)	O	compare output of channel CC63
	VADC		VADC gating input, see CCU60_MOSEL for the respective trigger select
LBE	n.c.	O	lost bit event output
whe_n ¹⁾	CCU61_CTRAPB	O	wrong Hall event output (inverted)
OTGB0[0:15]	OCDS	O	OCDS Trigger Bus 0
OTGB1[0:15]	OCDS	O	OCDS Trigger Bus 1

Capture/Compare Unit 6 (CCU6)
Table 27-18 CCU61 Digital Connections in TC26x B-Step (cont'd)

Signal	from/to Module	I/O to CCU61	Can be used to/as
SR0	Interrupt Router: SRC_CCU61SR0	O	CCU61 Service Request 0
SR1	CCU61_T13HRH	O	T13 count trigger
	VADC		VADC gating input, see CCU60_MOSEL for the respective trigger select
	Interrupt Router: SRC_CCU61SR1		CCU61 Service Request 1
SR2	CCU60_CCPOS0B, CCU60_T12HRG, CCU60_T13HRG	O	CCU60 triggers
	Interrupt Router: SRC_CCU61SR2		CCU61 Service Request 2
SR3	VADC_GxREQTRB (x = 0..3), VADC_BGREQTRB	O	VADC trigger capability, for additional options see also CCU60_MOSEL
	Interrupt Router: SRC_CCU61SR3		CCU61 Service Request 3

1) CCU6x_who_n is the inverted version of the internal Wrong Hall Event signal whe (to match the polarity requirements of the CTRAP input)

2) not available for devices in 144-pin package

28 General Purpose Timer Unit (GPT12)

The General Purpose Timer Unit blocks GPT1 and GPT2 have very flexible multifunctional timer structures which may be used for timing, event counting, pulse width measurement, pulse generation, frequency multiplication, and other purposes.

They incorporate five 16-bit timers that are grouped into the two timer blocks GPT1 and GPT2. Each timer in each block may operate independently in a number of different modes such as Gated Timer or Counter Mode, or may be concatenated with another timer of the same block.

Each block has alternate input/output functions and specific interrupts (service requests) associated with it. Input signals can be selected from several sources by register PISEL.

The GPT12 module is clocked with clock f_{GPT} .

Block GPT1 contains three timers/counters: The core timer T3 and the two auxiliary timers T2 and T4. The maximum resolution is $f_{\text{GPT}}/4$. The auxiliary timers of GPT1 may optionally be configured as reload or capture registers for the core timer. These registers are listed in [Section 28.1.6.1](#).

The following list summarizes the supported features:

- $f_{\text{GPT}}/4$ maximum resolution
- 3 independent timers/counters
- Timers/counters can be concatenated
- 4 operating modes:
 - Timer Mode
 - Gated Timer Mode
 - Counter Mode
 - Incremental Interface Mode
- Reload and Capture functionality
- Separate interrupts

Block GPT2 contains two timers/counters: The core timer T6 and the auxiliary timer T5. The maximum resolution is $f_{\text{GPT}}/2$. An additional Capture/Reload register (CAPREL) supports capture and reload operation with extended functionality. These registers are listed in [Section 28.2.7.1](#).

The following list summarizes the supported features:

- $f_{\text{GPT}}/2$ maximum resolution
- 2 independent timers/counters
- Timers/counters can be concatenated
- 3 operating modes:
 - Timer Mode
 - Gated Timer Mode
 - Counter Mode
- Extended capture/reload functions via 16-bit capture/reload register CAPREL
- Separate interrupts

General Purpose Timer Unit (GPT12)

28.1 Timer Block GPT1

All three timers of block GPT1 (T2, T3, T4) can run in one of 4 basic modes: Timer Mode, Gated Timer Mode, Counter Mode, or Incremental Interface Mode. All timers can count up or down. Each timer of GPT1 is controlled by a separate control register TxCON.

Each timer has an input pin TxIN (alternate pin function) associated with it, which serves as the gate control in Gated Timer Mode, or as the count input in Counter Mode. The count direction (up/down) may be programmed via software or may be dynamically altered by a signal at the External Up/Down control input TxEUD (alternate pin function). An overflow/underflow of core timer T3 is indicated by the Output Toggle Latch T3OTL, whose state may be output on the associated pin T3OUT (alternate pin function). The auxiliary timers T2 and T4 may additionally be concatenated with the core timer T3 (through T3OTL) or may be used as capture or reload registers for the core timer T3.

The current contents of each timer can be read or modified by the CPU by accessing the corresponding timer count registers T2, T3, or T4. When any of the timer registers is written to by the CPU in the state immediately preceding a timer increment, decrement, reload, or capture operation, the CPU write operation has priority in order to guarantee correct results.

The interrupt requests of GPT1 are signalled on service request lines SR0, SR1, and SR2.

The input and output lines of GPT1 are connected to pins. The control registers for the port functions are located in the respective port modules.

Note: The timing requirements for external input signals can be found in [Section 28.1.5](#), [Section 28.5.2](#) summarizes the module interface signals, including pins and interrupt request signals.

General Purpose Timer Unit (GPT12)

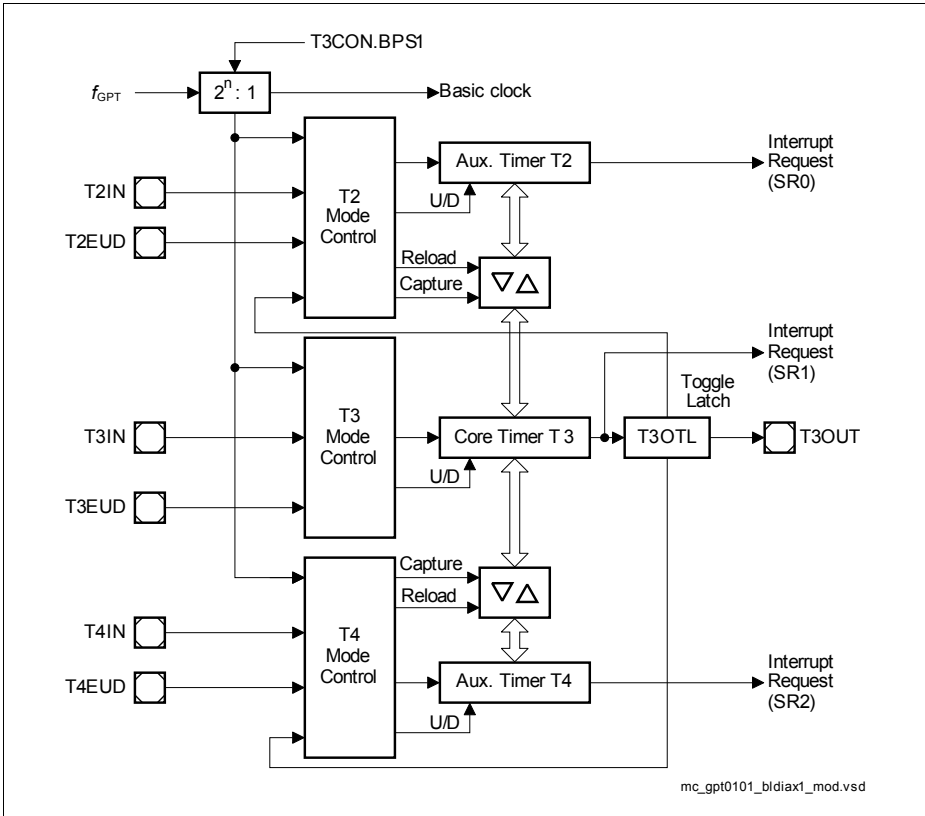


Figure 28-1 GPT1 Block Diagram

28.1.1 GPT1 Core Timer T3 Control

The current contents of the core timer T3 are reflected by its count register T3. This register can also be written to by the CPU, for example, to set the initial start value.

The core timer T3 is configured and controlled via its control register T3CON.

Timer T3 Run Control

The core timer T3 can be started or stopped by software through bit T3R (Timer T3 Run Bit). This bit is relevant in all operating modes of T3. Setting bit T3R will start the timer, clearing bit T3R stops the timer.

In Gated Timer Mode, the timer will only run if T3R = 1 and the gate is active (high or low, as programmed).

Note: When bit T2RC or T4RC in timer control register T2CON or T4CON is set, bit T3R will also control (start and stop) the auxiliary timer(s) T2 and/or T4.

Count Direction Control

The count direction of the GPT1 timers (core timer and auxiliary timers) can be controlled either by software or by the external input pin TxEUD (Timer Tx External Up/Down Control Input). These options are selected by bits TxUD and TxUDE in the respective control register TxCON. When the up/down control is provided by software (bit TxUDE = 0), the count direction can be altered by setting or clearing bit TxUD. When bit TxUDE = 1, pin TxEUD is selected to be the controlling source of the count direction. However, bit TxUD can still be used to reverse the actual count direction, as shown in [Table 28-6](#). The count direction can be changed regardless of whether or not the timer is running.

Note: When pin TxEUD is used as external count direction control input, it must be configured as input.

General Purpose Timer Unit (GPT12)

Timer T3 Output Toggle Latch

The overflow/underflow signal of timer T3 is connected to a block named ‘Toggle Latch’, shown in the Timer Mode diagrams. **Figure 28-2** illustrates the details of this block. An overflow or underflow of T3 will clock two latches: The first latch represents bit T3OTL in control register T3CON. The second latch is an internal latch toggled by T3OTL’s output. Both latch outputs are connected to the input control blocks of the auxiliary timers T2 and T4. The output level of the shadow latch will match the output level of T3OTL, but is delayed by one clock cycle. When the T3OTL value changes, this will result in a temporarily different output level from T3OTL and the shadow latch, which can trigger the selected count event in T2 and/or T4.

When software writes to T3OTL, both latches are set or cleared simultaneously. In this case, both signals to the auxiliary timers carry the same level and no edge will be detected. Bit T3OE (overflow/underflow output enable) in register T3CON enables the state of T3OTL to be monitored via an external pin T3OUT. When T3OTL is linked to an external port pin (must be configured as output), T3OUT can be used to control external HW. If T3OE = 1, pin T3OUT outputs the state of T3OTL. If T3OE = 0, pin T3OUT outputs a high level (as long as the T3OUT alternate function is selected for the port pin).

The trigger signals can serve as an input for the counter function or as a trigger source for the reload function of the auxiliary timers T2 and T4.

As can be seen from **Figure 28-2**, when latch T3OTL is modified by software to determine the state of the output line, also the internal shadow latch is set or cleared accordingly. Therefore, no trigger condition is detected by T2/T4 in this case.

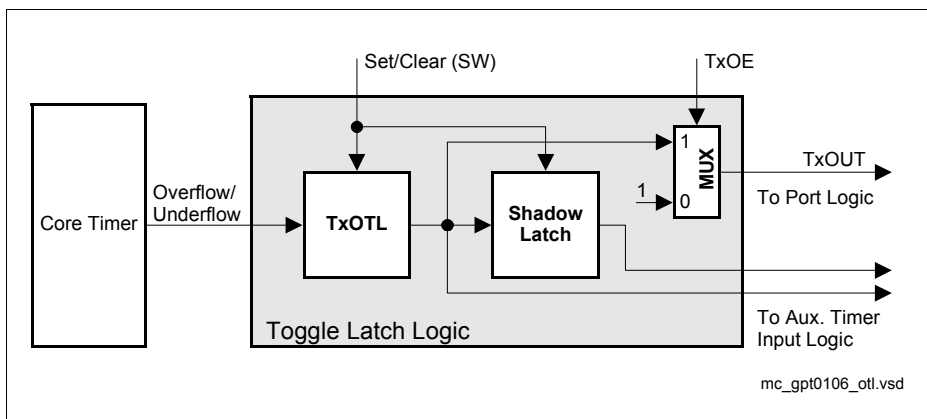


Figure 28-2 Block Diagram of the Toggle Latch Logic of Core Timer T3 (x = 3)

General Purpose Timer Unit (GPT12)

28.1.2 GPT1 Core Timer T3 Operating Modes

Timer T3 can operate in one of several modes.

Timer T3 in Timer Mode

Timer Mode for the core timer T3 is selected by setting bitfield T3M in register T3CON to 000_B. In Timer Mode, T3 is clocked with the module's input clock f_{GPT} divided by two programmable prescalers controlled by bitfields BPS1 and T3I in register T3CON. Please see [Section 28.1.5](#) for details on the input clock options.

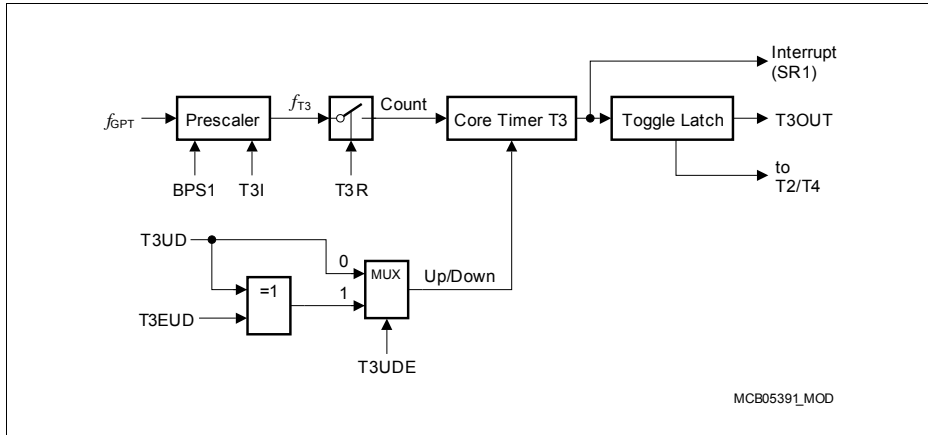


Figure 28-3 Block Diagram of Core Timer T3 in Timer Mode

General Purpose Timer Unit (GPT12)

Timer T3 in Gated Timer Mode

Gated Timer Mode for the core timer T3 is selected by setting bitfield T3M in register T3CON to 010_B or 011_B. Bit T3M.0 (T3CON.3) selects the active level of the gate input. The same options for the input frequency are available in Gated Timer Mode as in Timer Mode (see Section 28.1.5). However, the input clock to the timer in this mode is gated by the external input pin T3IN (Timer T3 External Input).

To enable this operation, the associated pin T3IN must be configured as input.

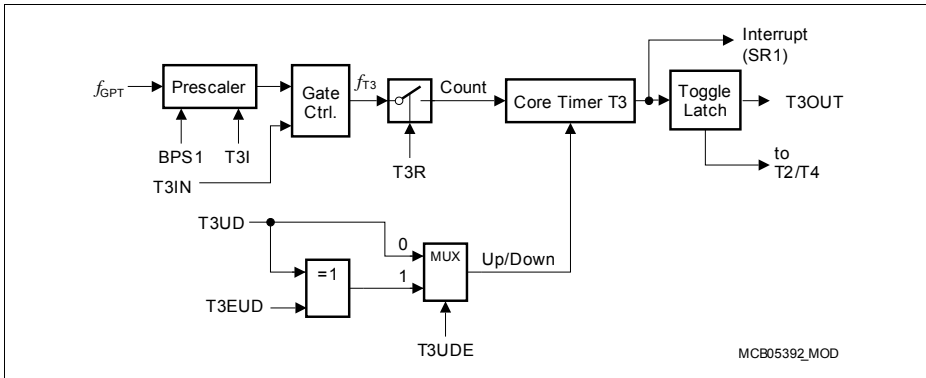


Figure 28-4 Block Diagram of Core Timer T3 in Gated Timer Mode

If T3M = 010_B, the timer is enabled when T3IN shows a low level. A high level at this line stops the timer. If T3M = 011_B, line T3IN must have a high level in order to enable the timer. Additionally, the timer can be turned on or off by software using bit T3R. The timer will only run if T3R is 1 and the gate is active. It will stop if either T3R is 0 or the gate is inactive.

Note: A transition of the gate signal at pin T3IN does not cause a service request via SR1.

General Purpose Timer Unit (GPT12)

Timer T3 in Counter Mode

Counter Mode for the core timer T3 is selected by setting bitfield T3M in register T3CON to 001_B. In Counter Mode, timer T3 is clocked by a transition at the external input pin T3IN. The event causing an increment or decrement of the timer can be a positive, a negative, or both a positive and a negative transition at this line. Bitfield T3I in control register T3CON selects the triggering transition (see [Table 28-8](#)).

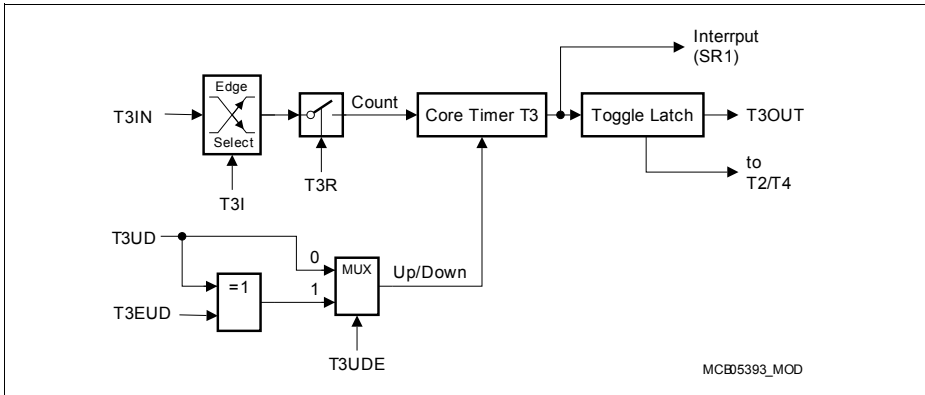


Figure 28-5 Block Diagram of Core Timer T3 in Counter Mode

For Counter Mode operation, pin T3IN must be configured as input. The maximum input frequency allowed in Counter Mode depends on the selected prescaler value. To ensure that a transition of the count input signal applied to T3IN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles before it changes. This information can be found in [Section 28.1.5](#).

General Purpose Timer Unit (GPT12)
Timer T3 in Incremental Interface Mode

Incremental Interface Mode for the core timer T3 is selected by setting bitfield T3M in register T3CON to 110_B or 111_B. In Incremental Interface Mode, the two inputs associated with core timer T3 (T3IN, T3EUD) are used to interface to an incremental encoder. T3 is clocked by each transition on one or both of the external input pins to provide 2-fold or 4-fold resolution of the encoder input.

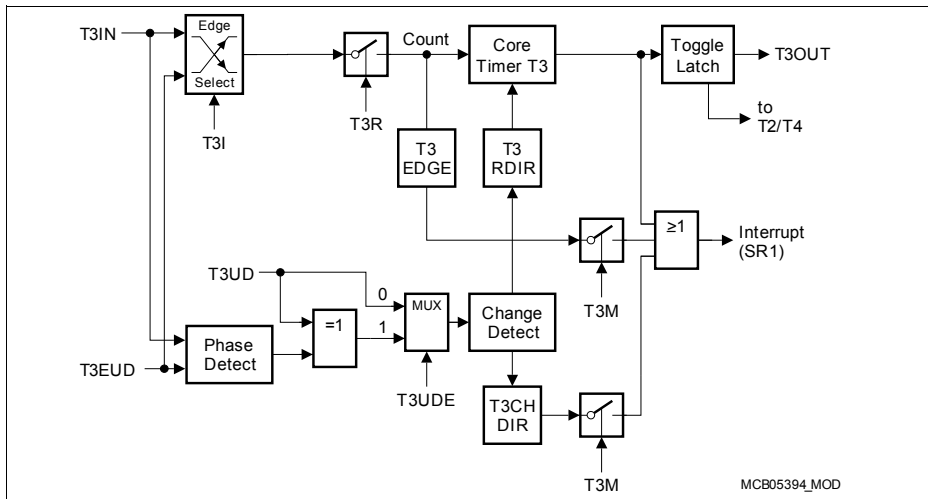


Figure 28-6 Block Diagram of Core Timer T3 in Incremental Interface Mode

Bitfield T3I in control register T3CON selects the triggering transitions (see [Table 28-10](#)). The sequence of the transitions of the two input signals is evaluated and generates count pulses as well as the direction signal. So T3 is modified automatically according to the speed and the direction of the incremental encoder and, therefore, its contents always represent the encoder's current position.

The service request generation can be selected: In Rotation Detection Mode (T3M = 110_B), a service request is generated each time the count direction of T3 changes. In Edge Detection Mode (T3M = 111_B), a service request is generated each time a count edge for T3 is detected. Count direction, changes in the count direction, and count requests are monitored by status bits T3RDIR, T3CHDIR, and T3EDGE in register T3CON.

The incremental encoder can be connected directly to the TC26x B-Step without external interface logic. In a standard system, however, comparators will be employed to convert the encoder's differential outputs (such as A, \bar{A}) to digital signals (such as A). This greatly increases noise immunity.

General Purpose Timer Unit (GPT12)

Note: The third encoder output T0, that indicates the mechanical zero position, may be connected to an external interrupt input and trigger a reset of timer T3. If input T4IN is available, T0 can be connected there and clear T3 automatically without requiring an interrupt (see bit CLRT3EN in register T4CON).

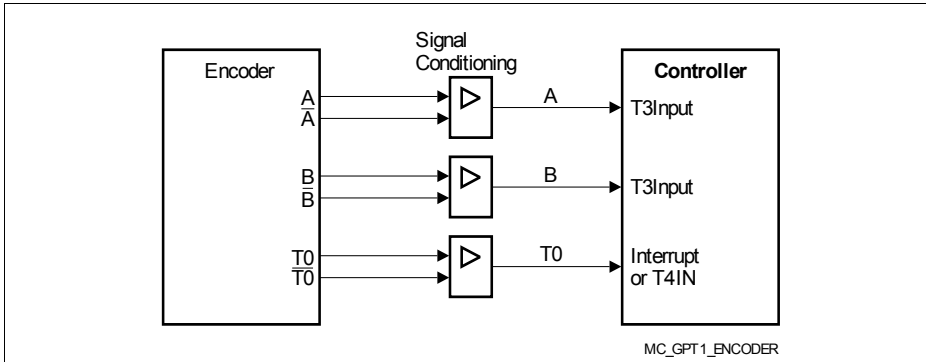


Figure 28-7 Connection of the Encoder to the TC26x B-Step

For Incremental Interface Mode operation, the following conditions must be met:

- Bitfield T3M must be 110_B or 111_B.
- Both pins T3IN and T3EUD must be configured as input.
- Pin T4IN must be configured as input, if used for T0.
- Bit T3UDE must be 1 to enable automatic external direction control.

The maximum count frequency allowed in Incremental Interface Mode depends on the selected prescaler value. To ensure that a transition of any input signal is recognized correctly, its level must be held high or low for a minimum number of module clock cycles before it changes. This information can be found in [Section 28.1.5](#).

As in Incremental Interface Mode two input signals with a 90° phase shift are evaluated, their maximum input frequency can be half the maximum count frequency.

In Incremental Interface Mode, the count direction is automatically derived from the sequence in which the input signals change, which corresponds to the rotation direction of the connected sensor. [Table 28-1](#) summarizes the possible combinations.

Table 28-1 GPT1 Core Timer T3 (Incremental Interface Mode) Count Direction

Level on Respective other Input	T3IN Input		T3EUD Input	
	Rising ↑	Falling ↓	Rising ↑	Falling ↓
High	Down	Up	Up	Down
Low	Up	Down	Down	Up

General Purpose Timer Unit (GPT12)

Figure 28-8 and Figure 28-9 give examples of T3's operation, visualizing count signal generation and direction control. They also show how input jitter is compensated, which might occur if the sensor rests near to one of its switching points.

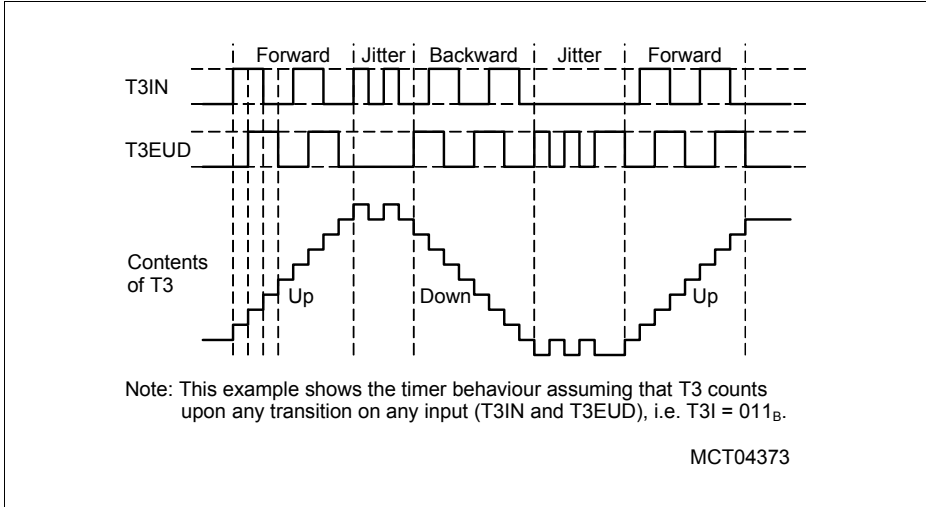


Figure 28-8 Evaluation of Incremental Encoder Signals, 2 Count Inputs

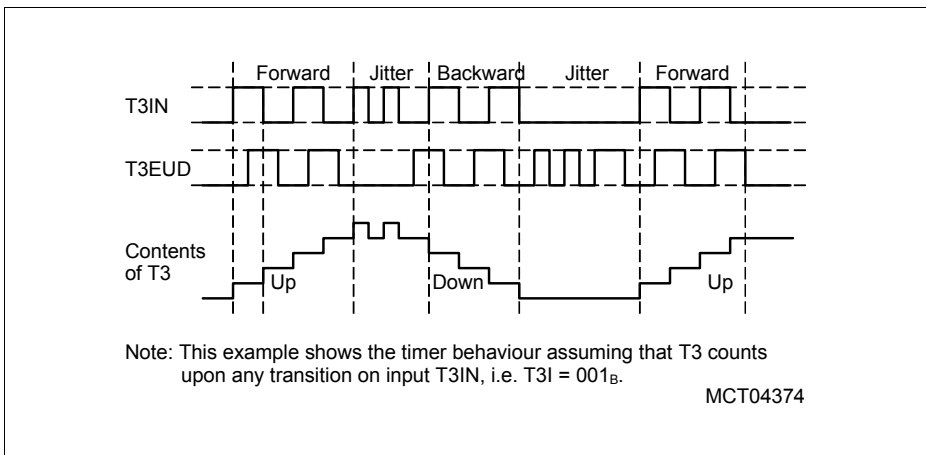


Figure 28-9 Evaluation of Incremental Encoder Signals, 1 Count Input

Note: Timer T3 operating in Incremental Interface Mode automatically provides information on the sensor's current position. Dynamic information (speed,

General Purpose Timer Unit (GPT12)

*acceleration, deceleration) may be obtained by measuring the incoming signal periods (see **“Combined Capture Modes”** on **Page 28-58**).*

General Purpose Timer Unit (GPT12)

28.1.3 GPT1 Auxiliary Timers T2/T4 Control

Auxiliary timers T2 and T4 have exactly the same functionality. They can be configured for Timer Mode, Gated Timer Mode, Counter Mode, or Incremental Interface Mode with the same options for the timer frequencies and the count signal as the core timer T3. In addition to these 4 counting modes, the auxiliary timers can be concatenated with the core timer, or they may be used as reload or capture registers in conjunction with the core timer. The start/stop function of the auxiliary timers can be remotely controlled by the T3 run control bit. Several timers may thus be controlled synchronously.

The current contents of an auxiliary timer are reflected by its count register T2 or T4, respectively. These registers can also be written to by the CPU, for example, to set the initial start value.

The individual configurations for timers T2 and T4 are determined by their control registers T2CON and T4CON, that are organized identically. Note that functions which are present in all 3 timers of block GPT1 are controlled in the same bit positions and in the same manner in each of the specific control registers.

Note: The auxiliary timers have no output toggle latch and no alternate output function.

Timer T2/T4 Run Control

Each of the auxiliary timers T2 and T4 can be started or stopped by software in two different ways:

- Through the associated timer run bit (T2R or T4R). In this case it is required that the respective control bit TxRC = 0.
- Through the core timer's run bit (T3R). In this case the respective remote control bit must be set (TxRC = 1).

The selected run bit is relevant in all operating modes of T2/T4. Setting the bit will start the timer, clearing the bit stops the timer.

In Gated Timer Mode, the timer will only run if the selected run bit is set and the gate is active (high or low, as programmed).

Note: If remote control is selected T3R will start/stop timer T3 and the selected auxiliary timer(s) synchronously.

Count Direction Control

The count direction of the GPT1 timers (core timer and auxiliary timers) is controlled in the same way, either by software or by the external input pin TxEUD. Please refer to the description in [Table 28-6](#).

Note: When pin TxEUD is used as external count direction control input, it must be configured as input.

General Purpose Timer Unit (GPT12)

28.1.4 GPT1 Auxiliary Timers T2/T4 Operating Modes

The operation of the auxiliary timers in the basic operating modes is almost identical with the core timer's operation, with very few exceptions. Additionally, some combined operating modes can be selected.

Timers T2 and T4 in Timer Mode

Timer Mode for an auxiliary timer Tx is selected by setting its bitfield TxM in register TxCON to 000_B.

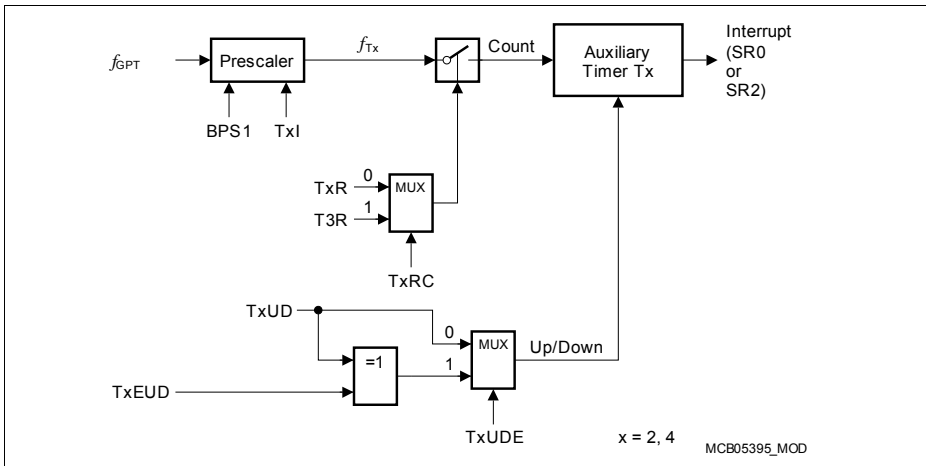


Figure 28-10 Block Diagram of an Auxiliary Timer in Timer Mode

General Purpose Timer Unit (GPT12)
Timers T2 and T4 in Gated Timer Mode

Gated Timer Mode for an auxiliary timer Tx is selected by setting bitfield TxM in register TxCON to 010_B or 011_B. Bit TxM.0 (TxCON.3) selects the active level of the gate input.

Note: A transition of the gate signal at TxIN does not cause a service request. Service requests of timer T2 are handled via SR0, and service requests of timer T4 are handled via SR2.

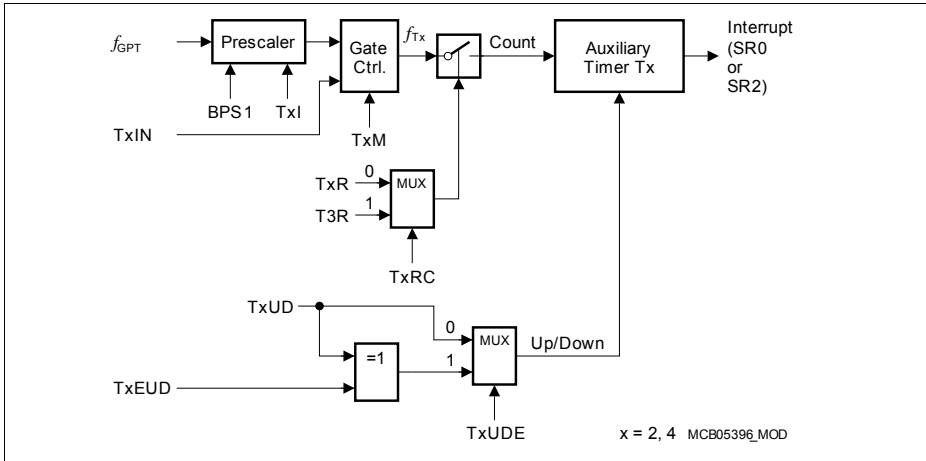


Figure 28-11 Block Diagram of an Auxiliary Timer in Gated Timer Mode

Note: There is no output toggle latch for T2 and T4.

Start/stop of an auxiliary timer can be controlled locally or remotely.

General Purpose Timer Unit (GPT12)

Timers T2 and T4 in Counter Mode

Counter Mode for an auxiliary timer Tx is selected by setting bitfield TxM in register TxCON to 001_B. In Counter Mode, an auxiliary timer can be clocked either by a transition at its external input line TxIN, or by a transition of timer T3's toggle latch T3OTL. The event causing an increment or decrement of a timer can be a positive, a negative, or both a positive and a negative transition at either the respective input pin or at the toggle latch. Bitfield TxI in control register TxCON selects the triggering transition (see [Table 28-9](#)).

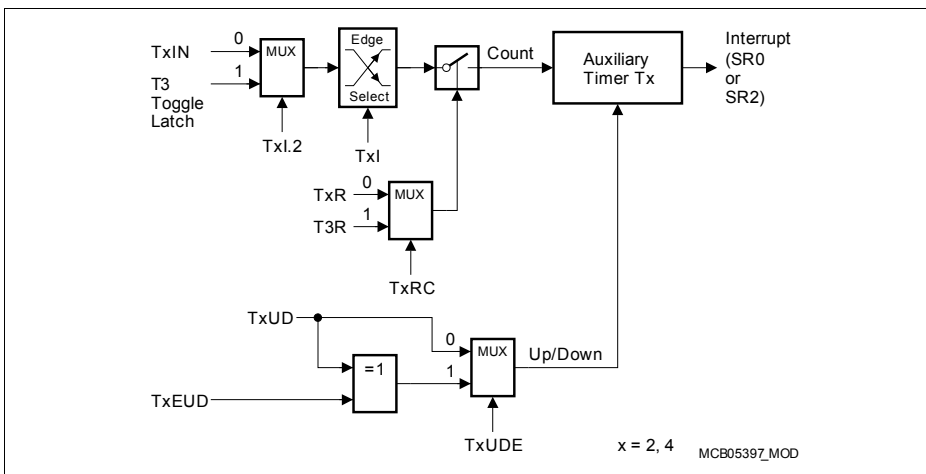


Figure 28-12 Block Diagram of an Auxiliary Timer in Counter Mode

Note: Only state transitions of T3OTL which are caused by the overflows/underflows of T3 will trigger the counter function of T2/T4. Modifications of T3OTL via software will NOT trigger the counter function of T2/T4.

For counter operation, pin TxIN must be configured as input. The maximum input frequency allowed in Counter Mode depends on the selected prescaler value. To ensure that a transition of the count input signal applied to TxIN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles before it changes. This information can be found in [Section 28.1.5](#).

General Purpose Timer Unit (GPT12)

Timer Concatenation

Using the toggle bit T3OTL as a clock source for an auxiliary timer in Counter Mode concatenates the core timer T3 with the respective auxiliary timer. This concatenation forms either a 32-bit or a 33-bit timer/counter, depending on which transition of T3OTL is selected to clock the auxiliary timer.

- **32-bit Timer/Counter:** If both a positive and a negative transition of T3OTL are used to clock the auxiliary timer, this timer is clocked on every overflow/underflow of the core timer T3. Thus, the two timers form a 32-bit timer.
- **33-bit Timer/Counter:** If either a positive or a negative transition of T3OTL is selected to clock the auxiliary timer, this timer is clocked on every second overflow/underflow of the core timer T3. This configuration forms a 33-bit timer (16-bit core timer + T3OTL + 16-bit auxiliary timer).

As long as bit T3OTL is not modified by software, it represents the state of the internal toggle latch, and can be regarded as part of the 33-bit timer.

The count directions of the two concatenated timers are not required to be the same. This offers a wide variety of different configurations.

T3, which represents the low-order part of the concatenated timer, can operate in Timer Mode, Gated Timer Mode or Counter Mode in this case.

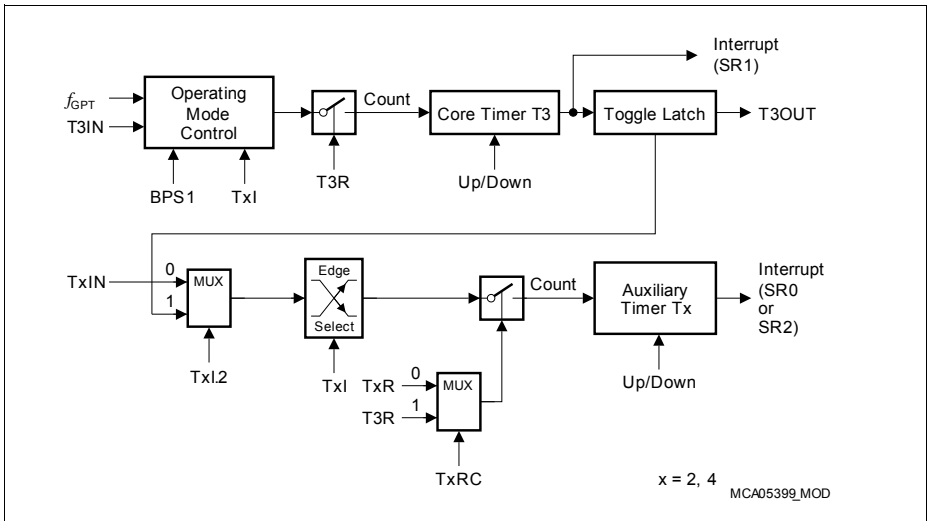


Figure 28-13 Concatenation of Core Timer T3 and an Auxiliary Timer

When reading the low and high parts of the concatenated timer, care must be taken to obtain consistent values in particular after a timer overflow/underflow (e.g. one part may already have considered an overflow, while the other has not).

General Purpose Timer Unit (GPT12)

Note: This is a general issue when reading multi-word results with consecutive instructions, and not necessarily unique to the GPT12 module architecture.

The following algorithm may be used to read concatenated GPT1 timers, represented by TIMER_HIGH (for auxiliary timer, here T2) and TIMER_LOW (for core timer T3). The high part is read twice, and reading of the low part is repeated if two different values were read for the high part:

- TIMER_HIGH_TMP = T2
- TIMER_LOW = T3
- Wait two basic clock cycles (to allow increment/decrement of auxiliary timer in case of core timer overflow/underflow) - see [Table 28-2](#)
- TIMER_HIGH = T2
- If TIMER_HIGH is not equal to TIMER_HIGH_TMP then TIMER_LOW = T3

After execution of this algorithm, TIMER_HIGH and TIMER_LOW represent a consistent time stamp of the concatenated timers.

The equivalent number of module clock cycles corresponding to two basic clock cycles is shown in [Table 28-2](#).

Table 28-2 Number of Module Clock Cycles to Wait for Two Basic Clock Cycles

Block Prescaler	BPS1 = 01 _B	BPS1 = 00 _B	BPS1 = 11 _B	BPS1 = 10 _B
Number of Module clocks	8	16	32	64

In case the required timer resolution can be achieved with different combinations of the Block Prescaler BPS1 and the Individual Prescalers TXI (see [Table 28-7](#)), the variant with the smallest value for the Block Prescaler may be chosen to minimize the waiting time.

General Purpose Timer Unit (GPT12)

Timers T2 and T4 in Incremental Interface Mode

Incremental Interface Mode for an auxiliary timer Tx is selected by setting bitfield TxM in the respective register TxCON to 110_B or 111_B. In Incremental Interface Mode, the two inputs associated with an auxiliary timer Tx (TxIN, TxEUD) are used to interface to an incremental encoder. Tx is clocked by each transition on one or both of the external input pins to provide 2-fold or 4-fold resolution of the encoder input.

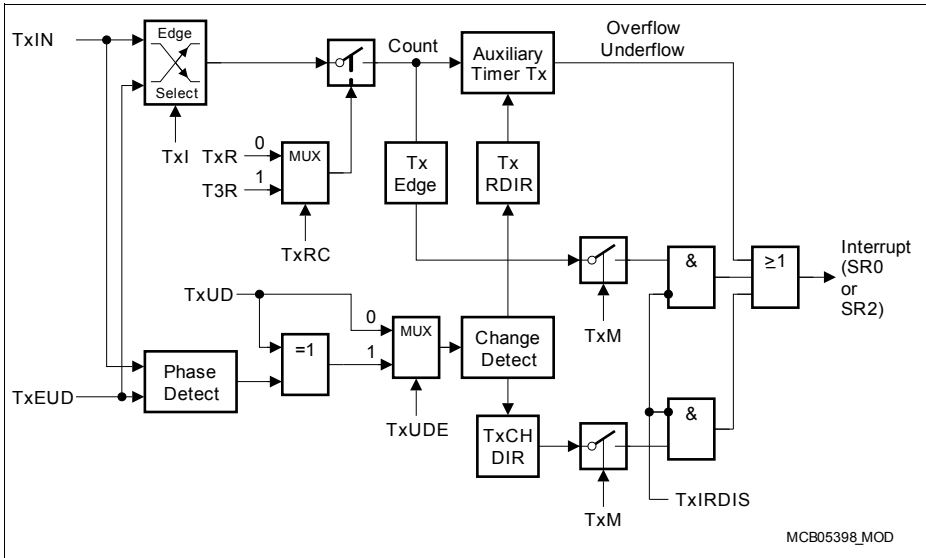


Figure 28-14 Block Diagram of an Auxiliary Timer in Incremental Interface Mode

The operation of the auxiliary timers T2 and T4 in Incremental Interface Mode and the interrupt generation are the same as described for the core timer T3. The descriptions, figures and tables apply accordingly.

Note: Timers T2 and T4 operating in Incremental Interface Mode automatically provide information on the sensor's current position. For dynamic information (speed, acceleration, deceleration) see "Combined Capture Modes" on Page 28-58).

General Purpose Timer Unit (GPT12)

Timers T2 and T4 in Reload Mode

Reload Mode for an auxiliary timer Tx is selected by setting bitfield TxM in the respective register TxCON to 100_B. In Reload Mode, the core timer T3 is reloaded with the contents of an auxiliary timer register, triggered by one of two different signals. The trigger signal is selected the same way as the clock source for Counter Mode (see [Table 28-9](#)), i.e. a transition of the auxiliary timer's input TxIN or the toggle latch T3OTL may trigger the reload.

Note: When programmed for Reload Mode, the respective auxiliary timer (T2 or T4) stops independently of its run flag T2R or T4R.

The timer input pin TxIN must be configured as input if it shall trigger a reload operation.

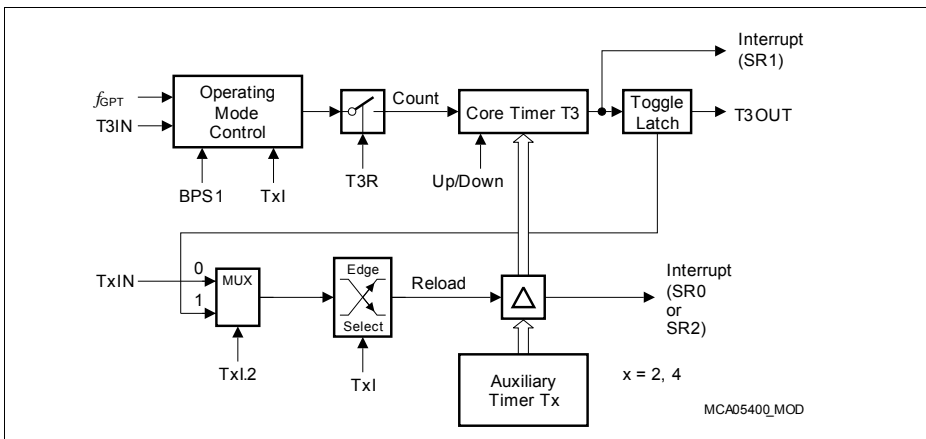


Figure 28-15 GPT1 Auxiliary Timer in Reload Mode

Upon a trigger signal, T3 is loaded with the contents of the respective timer register (T2 or T4) and the respective service request SR0 or SR2 is activated.

Note: When a T3OTL transition is selected for the trigger signal, service request SR1 will also become active, indicating T3's overflow or underflow. Modifications of T3OTL via software will NOT trigger the counter function of T2/T4.

To ensure that a transition of the reload input signal applied to TxIN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles, detailed in [Section 28.1.5](#).

The Reload Mode triggered by the T3 toggle latch can be used in a number of different configurations. The following functions can be performed, depending on the selected active transition:

General Purpose Timer Unit (GPT12)

- If both a positive and a negative transition of T3OTL are selected to trigger a reload, the core timer will be reloaded with the contents of the auxiliary timer each time it overflows or underflows. This is the standard Reload Mode (reload on overflow/underflow).
- If either a positive or a negative transition of T3OTL is selected to trigger a reload, the core timer will be reloaded with the contents of the auxiliary timer on every second overflow or underflow.
- Using this “single-transition” mode for both auxiliary timers allows to perform very flexible Pulse Width Modulation (PWM). One of the auxiliary timers is programmed to reload the core timer on a positive transition of T3OTL, the other is programmed for a reload on a negative transition of T3OTL. With this combination the core timer is alternately reloaded from the two auxiliary timers.

Figure 28-16 shows an example for the generation of a PWM signal using the “single-transition” reload mechanism. T2 defines the high time of the PWM signal (reloaded on positive transitions) and T4 defines the low time of the PWM signal (reloaded on negative transitions). The PWM signal can be output on pin T3OUT if T3OE = 1. With this method, the high and low time of the PWM signal can be varied in a wide range.

Note: The output toggle latch T3OTL is accessible via software and may be changed, if required, to modify the PWM signal.

However, this will NOT trigger the reloading of T3.

General Purpose Timer Unit (GPT12)

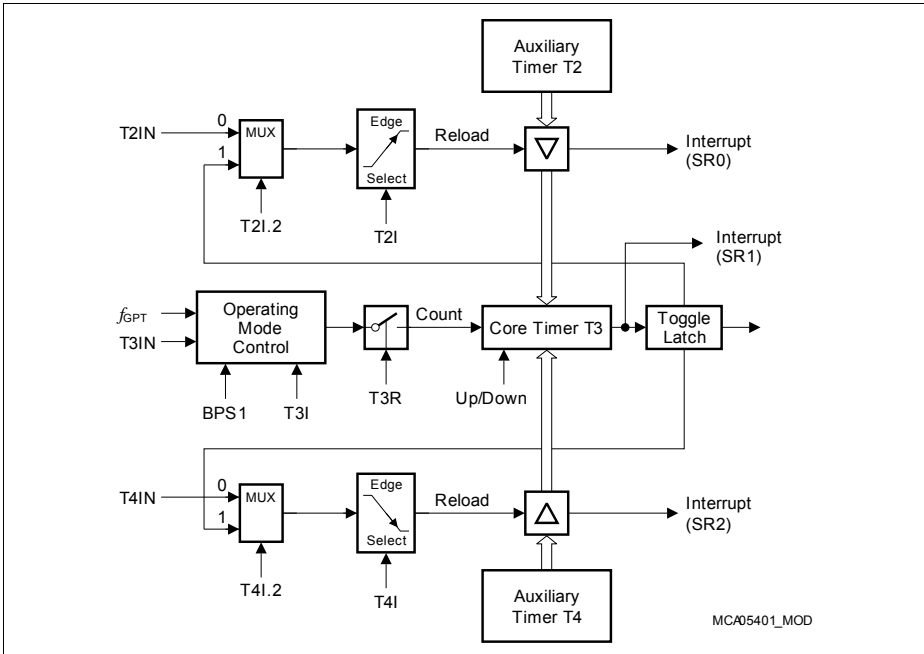


Figure 28-16 GPT1 Timer Reload Configuration for PWM Generation

Note: Although possible, selecting the same reload trigger event for both auxiliary timers should be avoided. In such a case, both reload registers would try to load the core timer at the same time. If this combination is selected, T2 is disregarded and the contents of T4 is reloaded.

The implementation of the GPT1 finite state machine may require special consideration in following cases.

In Case 1, when T2 or T4 are used to reload T3 on overflow/underflow, and T3 is read by software on the fly, the following unexpected values may be read from T3:

- When T3 is counting up, 0000_H or 0001_H may be read from T3 directly after an overflow, although the reload value in T2/T4 is higher (0001_H may be read in particular if BPS1 = 01_B and T3I = 000_B).
- When T3 is counting down, FFFF_H or FFFE_H may be read from T3 directly after an underflow, although the reload value in T2/T4 is lower (FFFE_H may be read in particular if BPS1 = 01_B and T3I = 000_B).

Note: All timings derived from T3 in this configuration (e.g. distance between service requests, PWM waveform on T3OUT, etc.) are accurate except for the specific case described below.

General Purpose Timer Unit (GPT12)

Recommendation:

- When T3 counts up, and T3_value < reload value is read from T3, T3_value should be replaced with the reload value for further calculations.
- When T3 counts down, and T3_value > reload value is read from T3, T3_value should be replaced with the reload value for further calculations.

Alternatively, if the intention is to identify the overflow/underflow of T3, the T3 service request (SR1) may be used.

In Case 2, when T2 is used to reload T3 in the configuration with BPS1 = 01_B and T3I = 000_B (i.e. fastest configuration/highest resolution of T3), the reload of T3 is performed with a delay of one basic clock cycle.

Recommendation:

To compensate the delay and achieve correct timing,

- increment the reload value in T2 by 1 when T3 is configured to count up,
- decrement the reload value in T2 by 1 when T3 is configured to count down.

Alternatively, use T4 instead of T2 as reload register for T3. In this configuration the reload of T3 is not delayed, i.e. the effect described above does not occur with T4.

General Purpose Timer Unit (GPT12)

Timers T2 and T4 in Capture Mode

Capture Mode for an auxiliary timer Tx is selected by setting bitfield TxM in the respective register TxCON to 101_B. In Capture Mode, the contents of the core timer T3 are latched into an auxiliary timer register in response to a signal transition at the respective auxiliary timer's external input pin TxIN. The capture trigger signal can be a positive, a negative, or both a positive and a negative transition.

The two least significant bits of bitfield TxI select the active transition (see [Table 28-9](#)). Bit 2 of TxI is irrelevant for Capture Mode and must be cleared (TxI.2 = 0).

Note: When programmed for Capture Mode, the respective auxiliary timer (T2 or T4) stops independently of its run flag T2R or T4R.

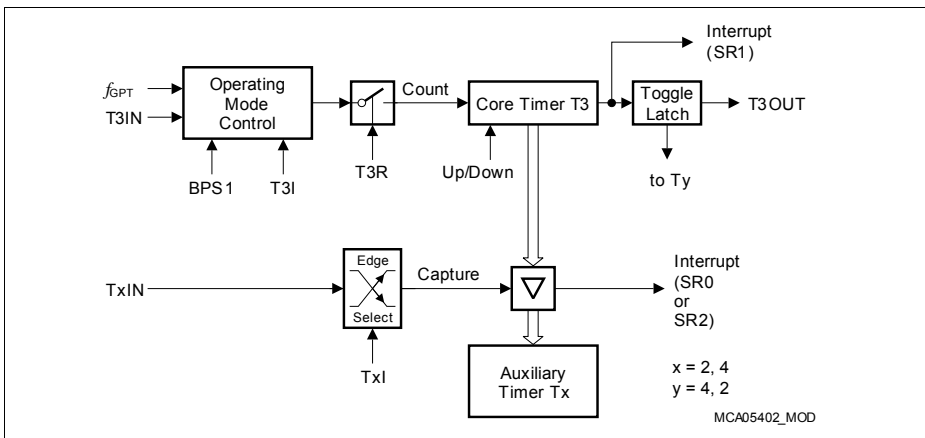


Figure 28-17 GPT1 Auxiliary Timer in Capture Mode

Upon a trigger (selected transition) at the corresponding input pin TxIN the contents of the core timer T3 are loaded into the auxiliary timer register Tx and the associated service request (SR0 or SR2) will be activated.

For Capture Mode operation, the respective timer input pin TxIN must be configured as input. To ensure that a transition of the capture input signal applied to TxIN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles, detailed in [Section 28.1.5](#).

General Purpose Timer Unit (GPT12)
28.1.5 GPT1 Clock Signal Control

All actions within the timer block GPT1 are triggered by transitions of its basic clock. This basic clock is derived from the module clock f_{GPT} by a basic block prescaler, controlled by bitfield BPS1 in register T3CON (see [Figure 28-1](#)). The count clock can be generated in two different ways:

- **Internal count clock**, derived from GPT1's basic clock via a programmable prescaler, is used for (Gated) Timer Mode.
- **External count clock**, derived from the timer's input pin(s), is used for Counter Mode or Incremental Interface Mode.

For both ways, the basic clock determines the maximum count frequency and the timer's resolution:

Table 28-3 Basic Clock Selection for Block GPT1

Block Prescaler ¹⁾	BPS1 = 01 _B	BPS1 = 00 _B ²⁾	BPS1 = 11 _B	BPS1 = 10 _B
Prescaling Factor for GPT1: F(BPS1)	F(BPS1) = 4	F(BPS1) = 8	F(BPS1) = 16	F(BPS1) = 32
Maximum External Count Frequency	$f_{GPT}/8$	$f_{GPT}/16$	$f_{GPT}/32$	$f_{GPT}/64$
Input Signal Stable Time	$4 \times t_{GPT}$	$8 \times t_{GPT}$	$16 \times t_{GPT}$	$32 \times t_{GPT}$

1) Please note the non-linear encoding of bitfield BPS1.

2) Default after reset.

Note: The GPT1 block uses a finite state machine to control the actions. Since multiple interactions are possible between the timers (T2, T3, T4), these elements are processed sequentially. However, all actions are normally completed within one basic clock cycle. The GPT1 state machine has 8 states (4 states when BPS1 = 01_B) and processes the timers in the order T3 - T2 (all actions except capture) - T4 - T2 (capture).

Note: When initializing the GPT1 block after reset, and the block prescaler BPS1 in register T3CON needs to be set to a value different from its default value (00_B), it must be initialized first before any mode involving external trigger signals is configured. These modes include counter, incremental interface, capture, and reload mode. Otherwise, unintended count/capture/reload events may occur during the first basic clock cycle.

In this case, or when changing BPS1 during operation of the GPT1 block, disable related interrupts before modification of BPS1, and afterwards clear the corresponding service request flags and re-initialize those registers (T2, T3, T4) that might be affected by a count/capture/reload event.

General Purpose Timer Unit (GPT12)
Internal Count Clock Generation

In Timer Mode and Gated Timer Mode, the count clock for each GPT1 timer is derived from the GPT1 basic clock by a programmable prescaler, controlled by bitfield TxI in the respective timer's control register TxCON.

The count frequency f_{Tx} for a timer Tx and its resolution r_{Tx} are scaled linearly with lower clock frequencies, as can be seen from the following formula:

$$f_{Tx} = \frac{f_{GPT}}{F(BPS1) \times 2^{<TxI>}} \quad r_{Tx}[\mu s] = \frac{F(BPS1) \times 2^{<TxI>}}{f_{GPT}[\text{MHz}]} \quad (28.1)$$

The effective count frequency depends on the common module clock prescaler factor F(BPS1) as well as on the individual input prescaler factor $2^{<TxI>}$. **Table 28-7** summarizes the resulting overall divider factors for a GPT1 timer that result from these cascaded prescalers.

Table 28-4 lists GPT1 timer's parameters (such as count frequency, resolution, and period) resulting from the selected overall prescaler factor $F(BPS1) \times 2^{<TxI>}$ and the module clock f_{GPT} . Note that some numbers may be rounded.

Table 28-4 GPT1 Timer Parameters

Overall Prescaler Factor	Example 1: Module Clock $f_{GPT} = 100 \text{ MHz}$			Example 2: Module Clock $f_{GPT} = 66.5 \text{ MHz}$		
	Frequency	Resolution	Period	Frequency	Resolution	Period
4	25 MHz	40 ns	2.62 ms	16.625 MHz	60.2 ns	3.94 ms
8	12.5 MHz	80 ns	5.24 ms	8.313 MHz	120.3 ns	7.88 ms
16	6.25 MHz	160 ns	10.49 ms	4.156 MHz	240.6 ns	15.77 ms
32	3.125 MHz	320 ns	20.97 ms	2.078 MHz	481.2 ns	31.54 ms
64	1.563 MHz	640 ns	41.94 ms	1.039 MHz	962.4 ns	63.07 ms
128	781.25 kHz	1.28 μs	83.89 ms	519.53 kHz	1.924 μs	126.1 ms
256	390.6 kHz	2.56 μs	167.8 ms	259.77 kHz	3.850 μs	252.3 ms
512	195.3 kHz	5.12 μs	335.5 ms	129.88 kHz	7.699 μs	504.6 ms
1024	97.7 kHz	10.24 μs	671.1 ms	64.94 kHz	15.398 μs	1.009 s
2048	48.8 kHz	20.48 μs	1.342 s	32.47 kHz	30.797 μs	2.018 s
4096	24.4 kHz	40.96 μs	2.684 s	16.24 kHz	61.594 μs	4.037 s

External Count Clock Input

The external input signals of the GPT1 block are sampled with the GPT1 basic clock (see **Figure 28-1**). To ensure that a signal is recognized correctly, its current level (high or

General Purpose Timer Unit (GPT12)

low) must be held active for at least one complete sampling period, before changing. A signal transition is recognized if two subsequent samples of the input signal represent different levels. Therefore, a minimum of two basic clock periods is required for the sampling of an external input signal. Thus, the maximum frequency of an input signal must not be higher than half the basic clock.

Table 28-5 summarizes the resulting requirements for external GPT1 input signals.

Table 28-5 GPT1 External Input Signal Limits

GPT1 Divider BPS1	Input Freq. Factor	Input Phase Duration	Example 1: Module Clock $f_{GPT} =$ 100 MHz		Example 2: Module Clock $f_{GPT} =$ 66.5 MHz	
			Max. Input Frequency	Min. Level Hold Time	Max. Input Frequency	Min. Level Hold Time
01 _B	$f_{GPT}/8$	$4 \times t_{GPT}$	12.5 MHz	40 ns	8.313 MHz	60.2 ns
00 _B	$f_{GPT}/16$	$8 \times t_{GPT}$	6.25 MHz	80 ns	4.156 MHz	120.3 ns
11 _B	$f_{GPT}/32$	$16 \times t_{GPT}$	3.125 MHz	160 ns	2.078 MHz	240.6 ns
10 _B	$f_{GPT}/64$	$32 \times t_{GPT}$	1.563 MHz	320 ns	1.039 MHz	481.2 ns

These limitations are valid for all external input signals to GPT1, including the external count signals in Counter Mode and Incremental Interface Mode, the gate input signals in Gated Timer Mode, and the external direction signals.

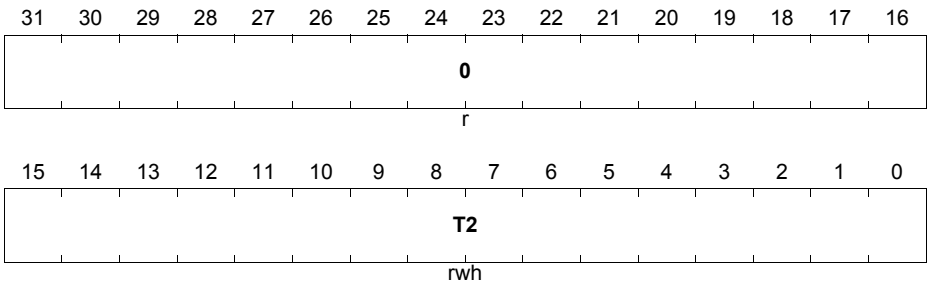
General Purpose Timer Unit (GPT12)

28.1.6 GPT1 Registers

28.1.6.1 GPT1 Timer Registers

T2

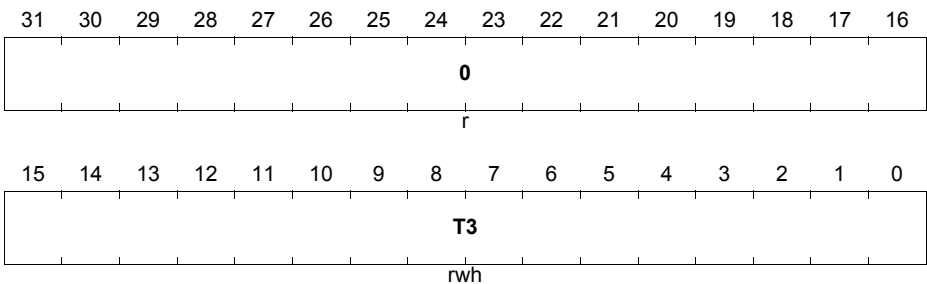
Timer T2 Register (34_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
T2	[15:0]	rwh	Timer T2 Contains the current value of Timer T2.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

T3

Timer T3 Register (38_H) Reset Value: 0000 0000_H

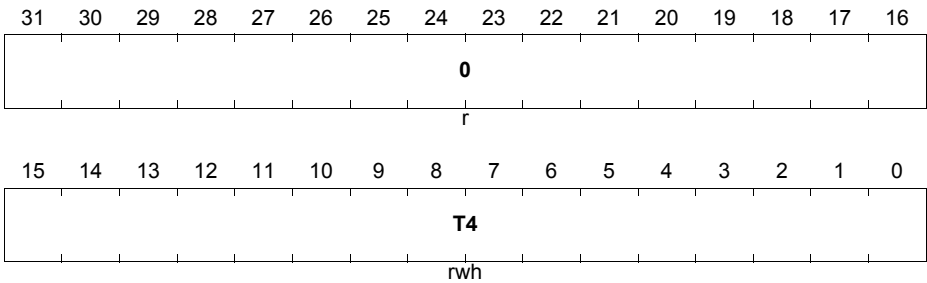


General Purpose Timer Unit (GPT12)

Field	Bits	Type	Description
T3	[15:0]	rwh	Timer T3 Contains the current value of Timer T3.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

T4

Timer T4 Register (3C_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
T4	[15:0]	rwh	Timer T4 Contains the current value of Timer T4.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

General Purpose Timer Unit (GPT12)

28.1.6.2 GPT1 Timer Control Registers

GPT1 Core Timer T3 Control Register

T3CON

Timer T3 Control Register

 (14_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T3 R DIR	T3 CH DIR	T3E DGE	BPS1	T3 OTL	T3 OE	T3 UDE	T3 UD	T3 R	T3M			T3I			
rh	rwh	rwh	rw	rwh	rw	rw	rw	rw	rw			rw			

Field	Bits	Type	Description
T3I	[2:0]	rw	Timer T3 Input Parameter Selection Depends on the operating mode, see respective sections for encoding: Table 28-7 for Timer Mode and Gated Timer Mode Table 28-8 for Counter Mode Table 28-10 for Incremental Interface Mode
T3M	[5:3]	rw	Timer T3 Mode Control 000 _B Timer Mode 001 _B Counter Mode 010 _B Gated Timer Mode with gate active low 011 _B Gated Timer Mode with gate active high 100 _B Reserved. Do not use this combination 101 _B Reserved. Do not use this combination 110 _B Incremental Interface Mode (Rotation Detection Mode) 111 _B Incremental Interface Mode (Edge Detection Mode)
T3R	6	rw	Timer T3 Run Bit 0 _B Timer T3 stops 1 _B Timer T3 runs

General Purpose Timer Unit (GPT12)

Field	Bits	Type	Description
T3UD	7	rw	Timer T3 Up/Down Control¹⁾ 0 _B Timer T3 counts up 1 _B Timer T3 counts down <i>Note: This bit only controls count direction of T3 if bit T3UDE = 0.</i>
T3UDE	8	rw	Timer T3 External Up/Down Enable¹⁾ 0 _B Count direction is controlled by bit T3UD; input T3EUD is disconnected 1 _B Count direction is controlled by input T3EUD
T3OE	9	rw	Overflow/Underflow Output Enable 0 _B Alternate Output Function Disabled 1 _B State of T3 toggle latch is output on pin T3OUT
T3OTL	10	rwh	Timer T3 Overflow Toggle Latch Toggles on each overflow/underflow of T3. Can be set or cleared by software (see separate description)
BPS1	[12:11]	rw	GPT1 Block Prescaler Control Selects the basic clock for block GPT1 (see also Section 28.1.5) 00 _B $f_{GPT}/8$ 01 _B $f_{GPT}/4$ 10 _B $f_{GPT}/32$ 11 _B $f_{GPT}/16$
T3EDGE	13	rwh	Timer T3 Edge Detection Flag The bit is set each time a count edge is detected. T3EDGE must be cleared by software. 0 _B No count edge was detected 1 _B A count edge was detected
T3CHDIR	14	rwh	Timer T3 Count Direction Change Flag This bit is set each time the count direction of timer T3 changes. T3CHDIR must be cleared by software. 0 _B No change of count direction was detected 1 _B A change of count direction was detected
T3RDIR	15	rh	Timer T3 Rotation Direction Flag 0 _B Timer T3 counts up 1 _B Timer T3 counts down
0	[31:16]	r	Reserved Read as 0; should be written with 0.

1) See [Table 28-6](#) for encoding of bits T3UD and T3UDE.

General Purpose Timer Unit (GPT12)

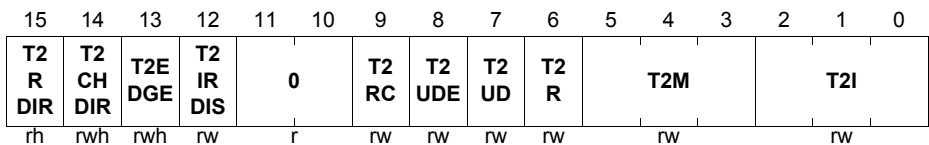
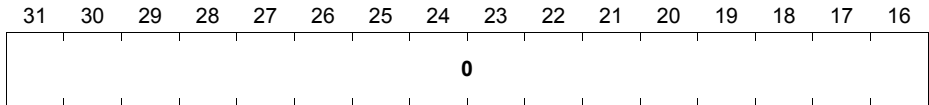
GPT1 Auxiliary Timers T2/T4 Control Registers

T2CON

Timer T2 Control Register

(10_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
T2I	[2:0]	rw	Timer T2 Input Parameter Selection Depends on the operating mode, see respective sections for encoding: Table 28-7 for Timer Mode and Gated Timer Mode Table 28-9 for Counter Mode Table 28-10 for Incremental Interface Mode
T2M	[5:3]	rw	Timer T2 Mode Control (Basic Operating Mode) 000 _B Timer Mode 001 _B Counter Mode 010 _B Gated Timer Mode with gate active low 011 _B Gated Timer Mode with gate active high 100 _B Reload Mode 101 _B Capture Mode 110 _B Incremental Interface Mode (Rotation Detection Mode) 111 _B Incremental Interface Mode (Edge Detection Mode)
T2R	6	rw	Timer T2 Run Bit 0 _B Timer T2 stops 1 _B Timer T2 runs <i>Note: This bit only controls timer T2 if bit T2RC = 0.</i>

General Purpose Timer Unit (GPT12)

Field	Bits	Type	Description
T2UD	7	rw	Timer T2 Up/Down Control¹⁾ 0 _B Timer T2 counts up 1 _B Timer T2 counts down <i>Note: This bit only controls count direction of T2 if bit T2UDE = 0.</i>
T2UDE	8	rw	Timer T2 External Up/Down Enable¹⁾ 0 _B Count direction is controlled by bit T2UD; input T2EUD is disconnected 1 _B Count direction is controlled by input T2EUD
T2RC	9	rw	Timer T2 Remote Control 0 _B Timer T2 is controlled by its own run bit T2R 1 _B Timer T2 is controlled by the run bit T3R of core timer T3, not by bit T2R
T2IRDIS	12	rw	Timer T2 Interrupt Disable 0 _B Interrupt generation for T2CHDIR and T2EDGE interrupts in Incremental Interface Mode is enabled 1 _B Interrupt generation for T2CHDIR and T2EDGE interrupts in Incremental Interface Mode is disabled
T2EDGE	13	rwh	Timer T2 Edge Detection The bit is set each time a count edge is detected. T2EDGE must be cleared by software. 0 _B No count edge was detected 1 _B A count edge was detected
T2CHDIR	14	rwh	Timer T2 Count Direction Change The bit is set each time the count direction of timer T2 changes. T2CHDIR must be cleared by software. 0 _B No change in count direction was detected 1 _B A change in count direction was detected
T2RDIR	15	rh	Timer T2 Rotation Direction 0 _B Timer T2 counts up 1 _B Timer T2 counts down
0	[11:10], [31:16]	r	Reserved Read as 0; should be written with 0.

1) See [Table 28-6](#) for encoding of bits T2UD and T2UDE.

General Purpose Timer Unit (GPT12)

T4CON

Timer T4 Control Register

 (18_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T4 R DIR	T4 CH DIR	T4E DGE	T4 IR DIS	CLR T3 EN	CLR T2 EN	T4 RC	T4 UDE	T4 UD	T4 R	T4M			T4I		
rh	rwh	rwh	rw	rw	rw	rw	rw	rw	rw	rw			rw		

Field	Bits	Type	Description
T4I	[2:0]	rw	Timer T4 Input Parameter Selection Depends on the operating mode, see respective sections for encoding: Table 28-7 for Timer Mode and Gated Timer Mode Table 28-9 for Counter Mode Table 28-10 for Incremental Interface Mode
T4M	[5:3]	rw	Timer T4 Mode Control (Basic Operating Mode) 000 _B Timer Mode 001 _B Counter Mode 010 _B Gated Timer Mode with gate active low 011 _B Gated Timer Mode with gate active high 100 _B Reload Mode 101 _B Capture Mode 110 _B Incremental Interface Mode (Rotation Detection Mode) 111 _B Incremental Interface Mode (Edge Detection Mode)
T4R	6	rw	Timer T4 Run Bit 0 _B Timer T4 stops 1 _B Timer T4 runs <i>Note: This bit only controls timer T4 if bit T4RC = 0.</i>

General Purpose Timer Unit (GPT12)

Field	Bits	Type	Description
T4UD	7	rw	Timer T4 Up/Down Control¹⁾ 0 _B Timer T4 counts up 1 _B Timer T4 counts down <i>Note: This bit only controls count direction of T4 if bit T4UDE = 0.</i>
T4UDE	8	rw	Timer T4 External Up/Down Enable¹⁾ 0 _B Count direction is controlled by bit T4UD; input T4EUD is disconnected 1 _B Count direction is controlled by input T4EUD
T4RC	9	rw	Timer T4 Remote Control 0 _B Timer T4 is controlled by its own run bit T4R 1 _B Timer T4 is controlled by the run bit T3R of core timer T3, but not by bit T4R
CLRT2EN	10	rw	Clear Timer T2 Enable Enables the automatic clearing of timer T2 upon a falling edge of the selected T4EUD input. 0 _B No effect of T4EUD on timer T2 1 _B A falling edge on T4EUD clears timer T2
CLRT3EN	11	rw	Clear Timer T3 Enable Enables the automatic clearing of timer T3 upon a falling edge of the selected T4IN input. 0 _B No effect of T4IN on timer T3 1 _B A falling edge on T4IN clears timer T3
T4IRDIS	12	rw	Timer T4 Interrupt Disable 0 _B Interrupt generation for T4CHDIR and T4EDGE interrupts in Incremental Interface Mode is enabled 1 _B Interrupt generation for T4CHDIR and T4EDGE interrupts in Incremental Interface Mode is disabled
T4EDGE	13	rwh	Timer T4 Edge Detection The bit is set each time a count edge is detected. T4EDGE has to be cleared by software. 0 _B No count edge was detected 1 _B A count edge was detected

General Purpose Timer Unit (GPT12)

Field	Bits	Type	Description
T4CHDIR	14	rwh	Timer T4 Count Direction Change The bit is set each time the count direction of timer T4 changes. T4CHDIR must be cleared by software. 0 _B No change in count direction was detected 1 _B A change in count direction was detected
T4RDIR	15	rh	Timer T4 Rotation Direction 0 _B Timer T4 counts up 1 _B Timer T4 counts down
0	[31:16]	r	Reserved Read as 0; should be written with 0.

1) See [Table 28-6](#) for encoding of bits T4UD and T4UDE.

General Purpose Timer Unit (GPT12)
Encoding of GPT1 Timer Count Direction Control
Table 28-6 GPT1 Timer Count Direction Control

Pin TxEUD	Bit TxUDE	Bit TxUD	Count Direction	Bit TxRDIR
X	0	0	Count Up	0
X	0	1	Count Down	1
0	1	0	Count Up	0
1	1	0	Count Down	1
0	1	1	Count Down	1
1	1	1	Count Up	0

Encoding of GPT1 Overall Prescaler Factor in Timer Mode and Gated Timer Mode
Table 28-7 GPT1 Overall Prescaler Factors for Internal Count Clock (Timer Mode and Gated Timer Mode)

Individual Prescaler for Tx	Common Prescaler for Module Clock ¹⁾			
	BPS1 = 01 _B	BPS1 = 00 _B	BPS1 = 11 _B	BPS1 = 10 _B
Txl = 000 _B	4	8	16	32
Txl = 001 _B	8	16	32	64
Txl = 010 _B	16	32	64	128
Txl = 011 _B	32	64	128	256
Txl = 100 _B	64	128	256	512
Txl = 101 _B	128	256	512	1024
Txl = 110 _B	256	512	1024	2048
Txl = 111 _B	512	1024	2048	4096

1) Please note the non-linear encoding of bitfield BPS1.

General Purpose Timer Unit (GPT12)
Encoding of GPT1 Input Edge Selection in Counter Mode
Table 28-8 GPT1 Core Timer T3 Input Edge Selection (Counter Mode)

Txl	Triggering Edge for Counter Increment/Decrement
000 _B	None. Counter T3 is disabled
001 _B	Positive transition (rising edge) on T3IN
010 _B	Negative transition (falling edge) on T3IN
011 _B	Any transition (rising or falling edge) on T3IN
1XX _B	Reserved. Do not use this combination

Table 28-9 GPT1 Auxiliary Timers T2/T4 Input Edge Selection

T2I/T4I	Triggering Edge for Counter Increment/Decrement, Capture¹⁾ or Reload
X00 _B	None. Counter Tx is disabled
001 _B	Positive transition (rising edge) on TxIN
010 _B	Negative transition (falling edge) on TxIN
011 _B	Any transition (rising or falling edge) on TxIN
101 _B	Positive transition (rising edge) of T3 toggle latch T3OTL ²⁾
110 _B	Negative transition (falling edge) of T3 toggle latch T3OTL ²⁾
111 _B	Any transition (rising or falling edge) of T3 toggle latch T3OTL ²⁾

1) For Capture Mode, only settings TxI = 0XX may be used.

2) Not for Capture Mode.

General Purpose Timer Unit (GPT12)**Encoding of Input Edge Selection in Incremental Interface Mode****Table 28-10 GPT1 Timer Tx Input Edge Selection
(Incremental Interface Mode)**

Txl	Triggering Edge for Counter Increment/Decrement
000 _B	None. Counter Tx stops.
001 _B	Any transition (rising or falling edge) on TxIN.
010 _B	Any transition (rising or falling edge) on TxEUD.
011 _B	Any transition (rising or falling edge) on any Tx input (TxIN or TxEUD).
1XX _B	Reserved. Do not use this combination.

General Purpose Timer Unit (GPT12)**28.2 Timer Block GPT2**

Both timers of block GPT2 (T5, T6) can run in one of 3 basic modes: Timer Mode, Gated Timer Mode, or Counter Mode. All timers can count up or down. Each timer of GPT2 is controlled by a separate control register TxCON.

Each timer has an input pin TxIN (alternate pin function) associated with it, which serves as the gate control in Gated Timer Mode, or as the count input in Counter Mode. The count direction (up/down) may be programmed via software or may be dynamically altered by a signal at the External Up/Down control input TxEUD (alternate pin function). An overflow/underflow of core timer T6 is indicated by the Output Toggle Latch T6OTL, whose state may be output on the associated pin T6OUT (alternate pin function). The auxiliary timer T5 may additionally be concatenated with core timer T6 (through T6OTL).

The Capture/Reload register CAPREL can be used to capture the contents of timer T5, or to reload timer T6. A special mode facilitates the use of register CAPREL for both functions at the same time. This mode allows frequency multiplication. The capture function is triggered by the input pin CAPIN, or by GPT1 timer's T3 input lines T3IN and T3EUD. The reload function is triggered by an overflow or underflow of timer T6.

The current contents of each timer can be read or modified by the CPU by accessing the corresponding timer count registers T5 or T6. When any of the timer registers is written by the CPU in the state immediately preceding a timer increment, decrement, reload, or capture operation, the CPU write operation has priority in order to guarantee correct results.

The interrupt requests of GPT2 are signalled on service request lines SR3, SR4, and SR5.

The input and output lines of GPT2 are connected to pins. The control registers for the port functions are located in the respective port modules.

Note: The timing requirements for external input signals can be found in [Section 28.2.6](#), [Section 28.5.2](#) summarizes the module interface signals, including pins and interrupt request signals.

General Purpose Timer Unit (GPT12)

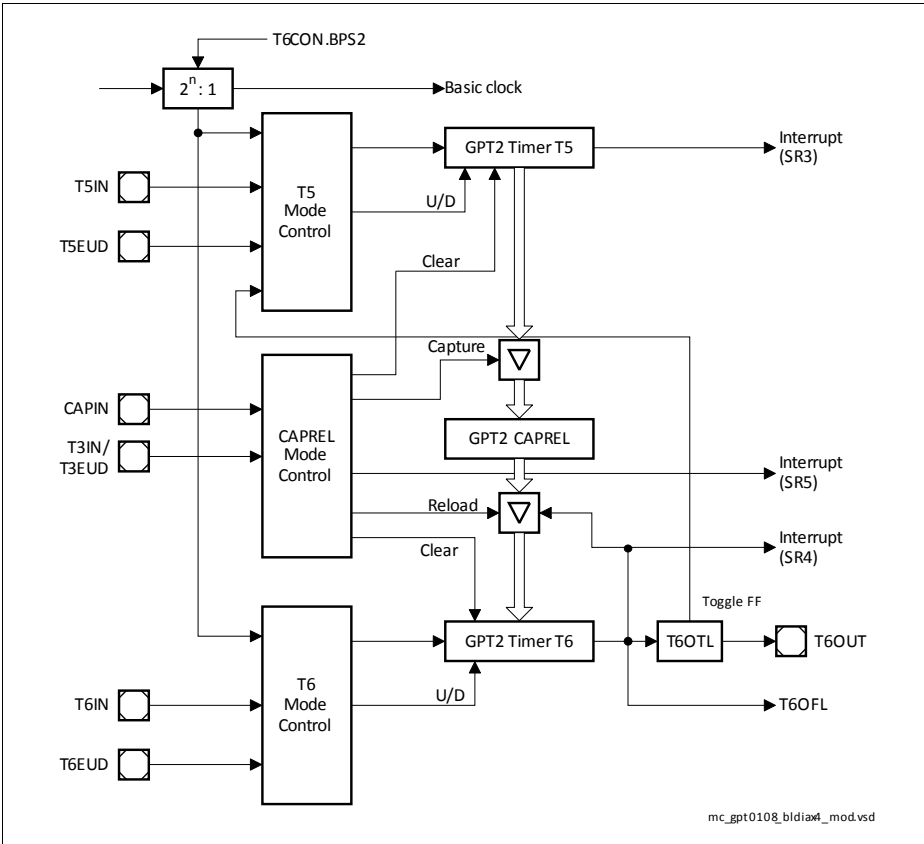


Figure 28-18 GPT2 Block Diagram

28.2.1 GPT2 Core Timer T6 Control

The current contents of the core timer T6 are reflected by its count register T6. This register can also be written to by the CPU, for example, to set the initial start value.

The core timer T6 is configured and controlled via its control register T6CON.

Timer T6 Run Control

The core timer T6 can be started or stopped by software through bit T6R (timer T6 run bit). This bit is relevant in all operating modes of T6. Setting bit T6R will start the timer, clearing bit T6R stops the timer.

In Gated Timer Mode, the timer will only run if T6R = 1 and the gate is active (high or low, as programmed).

Note: When bit T5RC in timer control register T5CON is set, bit T6R will also control (start and stop) the Auxiliary Timer T5.

Count Direction Control

The count direction of the GPT2 timers (core timer and auxiliary timer) can be controlled either by software or by the external input pin TxEUD (Timer Tx External Up/Down Control Input). These options are selected by bits TxUD and TxUDE in the respective control register TxCON. When the up/down control is provided by software (bit TxUDE = 0), the count direction can be altered by setting or clearing bit TxUD. When bit TxUDE = 1, pin TxEUD is selected to be the controlling source of the count direction. However, bit TxUD can still be used to reverse the actual count direction, as shown in [Table 28-15](#). The count direction can be changed regardless of whether or not the timer is running.

General Purpose Timer Unit (GPT12)

Timer T6 Output Toggle Latch

The overflow/underflow signal of timer T6 is connected to a block named ‘Toggle Latch’, shown in the Timer Mode diagrams. **Figure 28-19** illustrates the details of this block. An overflow or underflow of T6 will clock two latches: The first latch represents bit T6OTL in control register T6CON. The second latch is an internal latch toggled by T6OTL’s output. Both latch outputs are connected to the input control block of the auxiliary timer T5. The output level of the shadow latch will match the output level of T6OTL, but is delayed by one clock cycle. When the T6OTL value changes, this will result in a temporarily different output level from T6OTL and the shadow latch, which can trigger the selected count event in T5.

When software writes to T6OTL, both latches are set or cleared simultaneously. In this case, both signals to the auxiliary timer carry the same level and no edge will be detected. Bit T6OE (overflow/underflow output enable) in register T6CON enables the state of T6OTL to be monitored via an external pin T6OUT. When T6OTL is linked to an external port pin (must be configured as output), T6OUT can be used to control external HW. If T6OE = 1, pin T6OUT outputs the state of T6OTL. If T6OE = 0, pin T6OUT outputs a high level (while it selects the timer output signal).

As can be seen from **Figure 28-19**, when latch T6OTL is modified by software to determine the state of the output line, also the internal shadow latch is set or cleared accordingly. Therefore, no trigger condition is detected by T5 in this case.

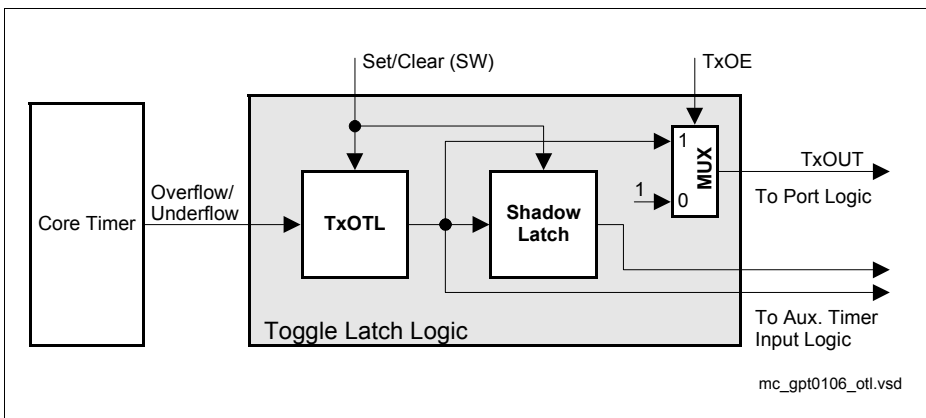


Figure 28-19 Block Diagram of the Toggle Latch Logic of Core Timer T6 (x = 6)

General Purpose Timer Unit (GPT12)

28.2.2 GPT2 Core Timer T6 Operating Modes

Timer T6 can operate in one of several modes.

Timer T6 in Timer Mode

Timer Mode for the core timer T6 is selected by setting bitfield T6M in register T6CON to 000_B. In this mode, T6 is clocked with the module's input clock f_{GPT} divided by two programmable prescalers controlled by bitfields BPS2 and T6I in register T6CON. Please see Section 28.2.6 for details on the input clock options.

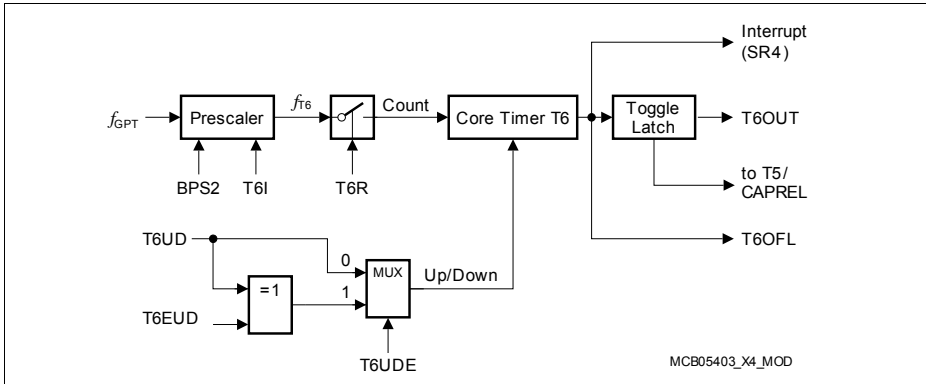


Figure 28-20 Block Diagram of Core Timer T6 in Timer Mode

General Purpose Timer Unit (GPT12)

Timer 6 in Gated Timer Mode

Gated Timer Mode for the core timer T6 is selected by setting bitfield T6M in register T6CON to 010_B or 011_B. Bit T6M.0 (T6CON.3) selects the active level of the gate input. The same options for the input frequency are available in Gated Timer Mode as in Timer Mode (see Section 28.2.6). However, the input clock to the timer in this mode is gated by the external input pin T6IN (Timer T6 External Input).

To enable this operation, the associated pin T6IN must be configured as input.

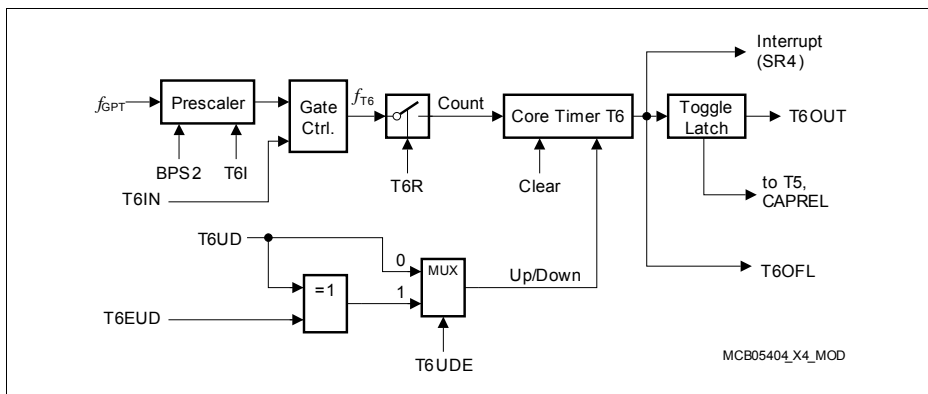


Figure 28-21 Block Diagram of Core Timer T6 in Gated Timer Mode

If T6M = 010_B, the timer is enabled when T6IN shows a low level. A high level at this line stops the timer. If T6M = 011_B, line T6IN must have a high level in order to enable the timer. Additionally, the timer can be turned on or off by software using bit T6R. The timer will only run if T6R is 1 and the gate is active. It will stop if either T6R is 0 or the gate is inactive.

Note: A transition of the gate signal at pin T6IN does not cause a service request.

General Purpose Timer Unit (GPT12)

Timer 6 in Counter Mode

Counter Mode for the core timer T6 is selected by setting bitfield T6M in register T6CON to 001_B. In Counter Mode, timer T6 is clocked by a transition at the external input pin T6IN. The event causing an increment or decrement of the timer can be a positive, a negative, or both a positive and a negative transition at this line. Bitfield T6I in control register T6CON selects the triggering transition (see [Table 28-17](#)).

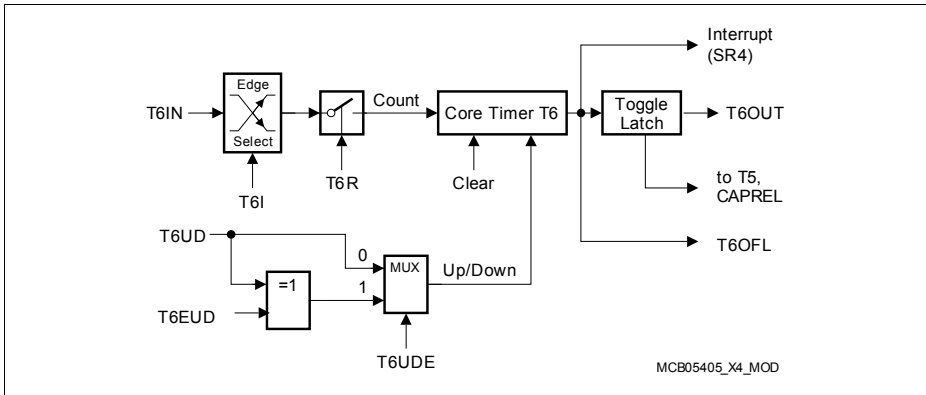


Figure 28-22 Block Diagram of Core Timer T6 in Counter Mode

For Counter Mode operation, pin T6IN must be configured as input. The maximum input frequency allowed in Counter Mode depends on the selected prescaler value. To ensure that a transition of the count input signal applied to T6IN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles before it changes. This information can be found in [Section 28.2.6](#).

General Purpose Timer Unit (GPT12)

28.2.3 GPT2 Auxiliary Timer T5 Control

Auxiliary timer T5 can be configured for Timer Mode, Gated Timer Mode, or Counter Mode with the same options for the timer frequencies and the count signal as the core timer T6. In addition to these 3 counting modes, the auxiliary timer can be concatenated with the core timer. The contents of T5 may be captured to register CAPREL upon an external or an internal trigger. The start/stop function of the auxiliary timer can be remotely controlled by the T6 run control bit. Both timers may thus be controlled synchronously.

The current contents of the auxiliary timer are reflected by its count register T5. This register can also be written to by the CPU, for example, to set the initial start value.

The individual configurations for timer T5 are determined by its control register T5CON. Some bits in this register also control the function of the CAPREL register. Note that functions which are present in all timers of block GPT2 are controlled in the same bit positions and in the same manner in each of the specific control registers.

Note: The auxiliary timer has no output toggle latch and no alternate output function.

Timer T5 Run Control

The auxiliary timer T5 can be started or stopped by software in two different ways:

- Through the associated timer run bit (T5R). In this case it is required that the respective control bit T5RC = 0.
- Through the core timer's run bit (T6R). In this case the respective remote control bit must be set (T5RC = 1).

The selected run bit is relevant in all operating modes of T5. Setting the bit will start the timer, clearing the bit stops the timer.

In Gated Timer Mode, the timer will only run if the selected run bit is set and the gate is active (high or low, as programmed).

Note: If remote control is selected T6R will start/stop timer T6 and the auxiliary timer T5 synchronously.

General Purpose Timer Unit (GPT12)

28.2.4 GPT2 Auxiliary Timer T5 Operating Modes

The operation of the auxiliary timer in the basic operating modes is almost identical with the core timer's operation, with very few exceptions. Additionally, some combined operating modes can be selected.

Timer T5 in Timer Mode

Timer Mode for the auxiliary timer T5 is selected by setting its bitfield T5M in register T5CON to 000_B.

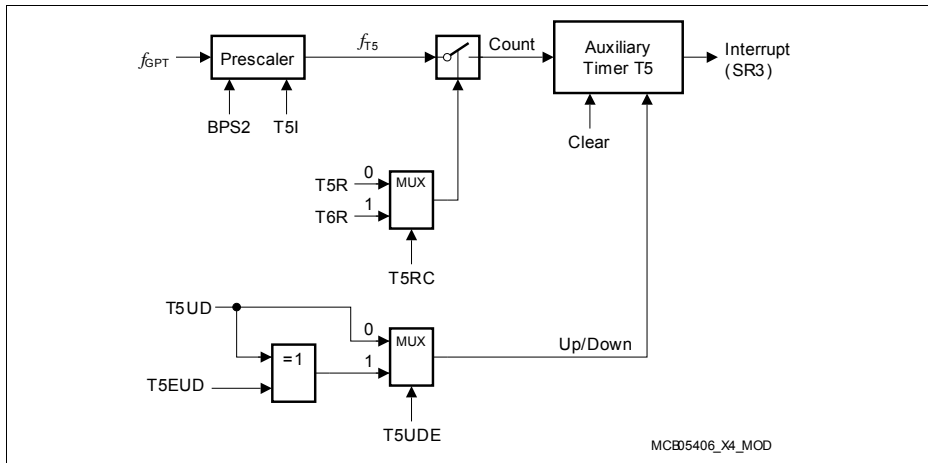


Figure 28-23 Block Diagram of Auxiliary Timer T5 in Timer Mode

General Purpose Timer Unit (GPT12)

Timer T5 in Gated Timer Mode

Gated Timer Mode for the auxiliary timer T5 is selected by setting bitfield T5M in register T5CON to 010_B or 011_B. Bit T5M.0 (T5CON.3) selects the active level of the gate input.

Note: A transition of the gate signal at line T5IN does not cause a service request.

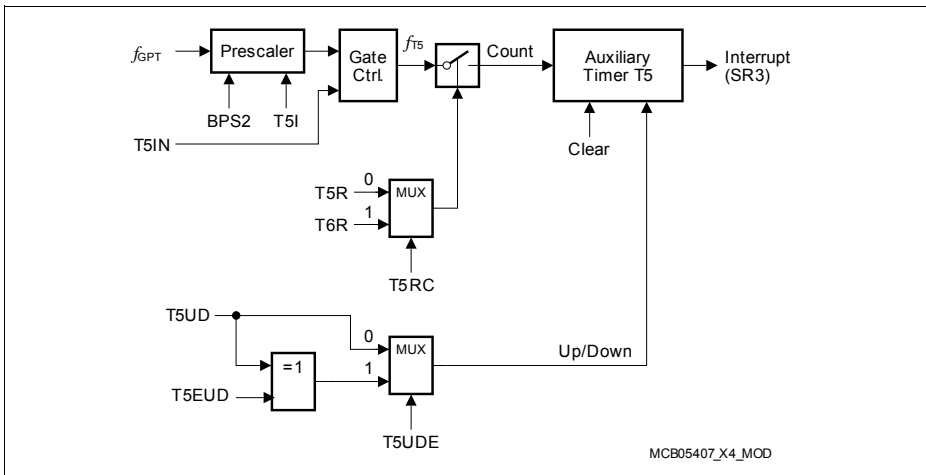


Figure 28-24 Block Diagram of Auxiliary Timer T5 in Gated Timer Mode

Note: There is no output toggle latch for T5.

Start/stop of the auxiliary timer can be controlled locally or remotely.

General Purpose Timer Unit (GPT12)

Timer T5 in Counter Mode

Counter Mode for auxiliary timer T5 is selected by setting bitfield T5M in register T5CON to 001_B. In Counter Mode, the auxiliary timer can be clocked either by a transition at its external input line T5IN, or by a transition of timer T6's toggle latch T6OTL. The event causing an increment or decrement of a timer can be a positive, a negative, or both a positive and a negative transition at either the respective input pin or at the toggle latch. Bitfield T5I in control register T5CON selects the triggering transition (see [Table 28-18](#)).

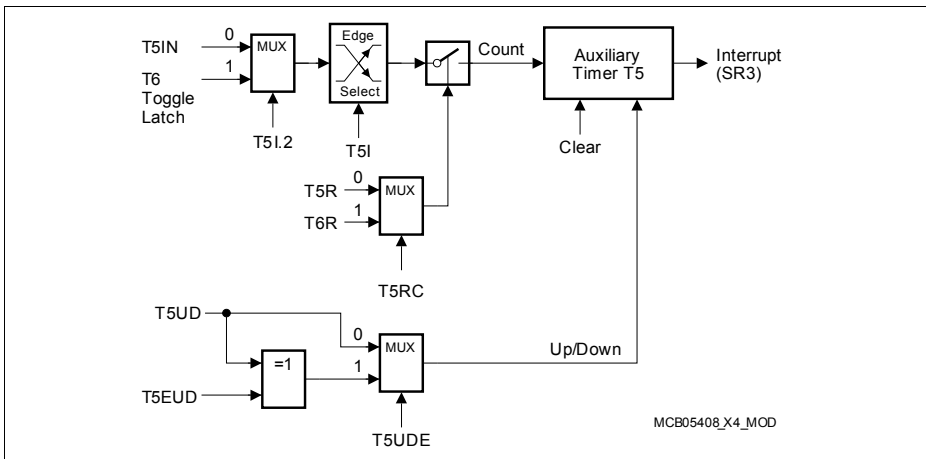


Figure 28-25 Block Diagram of Auxiliary Timer T5 in Counter Mode

Note: Only state transitions of T6OTL which are caused by the overflows/underflows of T6 will trigger the counter function of T5. Modifications of T6OTL via software will NOT trigger the counter function of T5.

For counter operation, pin T5IN must be configured as input. The maximum input frequency allowed in Counter Mode depends on the selected prescaler value. To ensure that a transition of the count input signal applied to T5IN is recognized correctly, its level must be held high or low for a minimum number of module clock cycles before it changes. This information can be found in [Section 28.2.6](#).

General Purpose Timer Unit (GPT12)

Timer Concatenation

Using the toggle bit T6OTL as a clock source for the auxiliary timer in Counter Mode concatenates the core timer T6 with the auxiliary timer T5. This concatenation forms either a 32-bit or a 33-bit timer/counter, depending on which transition of T6OTL is selected to clock the auxiliary timer.

- **32-bit Timer/Counter:** If both a positive and a negative transition of T6OTL are used to clock the auxiliary timer, this timer is clocked on every overflow/underflow of the core timer T6. Thus, the two timers form a 32-bit timer.
- **33-bit Timer/Counter:** If either a positive or a negative transition of T6OTL is selected to clock the auxiliary timer, this timer is clocked on every second overflow/underflow of the core timer T6. This configuration forms a 33-bit timer (16-bit core timer + T6OTL + 16-bit auxiliary timer).

As long as bit T6OTL is not modified by software, it represents the state of the internal toggle latch, and can be regarded as part of the 33-bit timer.

The count directions of the two concatenated timers are not required to be the same. This offers a wide variety of different configurations.

T6, which represents the low-order part of the concatenated timer, can operate in Timer Mode, Gated Timer Mode or Counter Mode in this case.

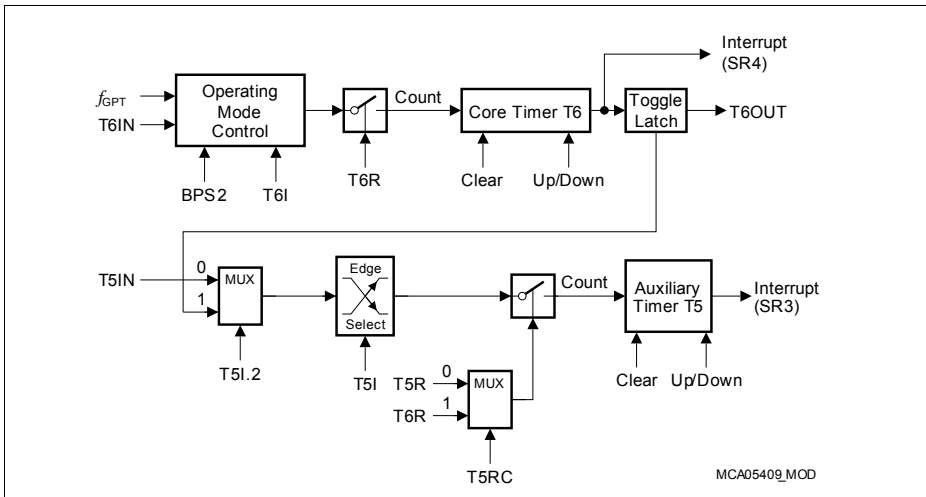


Figure 28-26 Concatenation of Core Timer T6 and Auxiliary Timer T5

When reading the low and high parts of the concatenated timer, care must be taken to obtain consistent values in particular after a timer overflow/underflow (e.g. one part may already have considered an overflow, while the other has not).

General Purpose Timer Unit (GPT12)

Note: This is a general issue when reading multi-word results with consecutive instructions, and not necessarily unique to the GPT12 module architecture.

The following algorithm may be used to read concatenated GPT2 timers, represented by TIMER_HIGH (for auxiliary timer T5) and TIMER_LOW (for core timer T6). The high part is read twice, and reading of the low part is repeated if two different values were read for the high part:

- TIMER_HIGH_TMP = T5
- TIMER_LOW = T6
- Wait two basic clock cycles (to allow increment/decrement of auxiliary timer in case of core timer overflow/underflow) - see [Table 28-11](#)
- TIMER_HIGH = T5
- If TIMER_HIGH is not equal to TIMER_HIGH_TMP then TIMER_LOW = T6

After execution of this algorithm, TIMER_HIGH and TIMER_LOW represent a consistent time stamp of the concatenated timers.

The equivalent number of module clock cycles corresponding to two basic clock cycles is shown in [Table 28-11](#).

Table 28-11 Number of Module Clock Cycles to Wait for Two Basic Clock Cycles

Block Prescaler	BPS2 = 01 _B	BPS2 = 00 _B	BPS2 = 11 _B	BPS2 = 10 _B
Number of module clocks	4	8	16	32

In case the required timer resolution can be achieved with different combinations of the Block Prescaler BPS2 and the Individual Prescalers TxI (see [Table 28-16](#)), the variant with the smallest value for the Block Prescaler may be chosen to minimize the waiting time. E.g. in order to run T6 at $f_{SYS}/512$, select BPS2 = 00_B, T6I = 111_B, and insert 8 NOPs (or other instructions) to ensure the required waiting time before reading TIMER_HIGH the second time.

General Purpose Timer Unit (GPT12)

28.2.5 GPT2 Register CAPREL Operating Modes

The Capture/Reload register CAPREL can be used to capture the contents of timer T5, or to reload timer T6. A special mode facilitates the use of register CAPREL for both functions at the same time. This mode allows frequency multiplication. The capture function is triggered by the input pin CAPIN, by GPT1 timer's T3 input lines T3IN and T3EUD, or by read accesses to GPT1 timers. The reload function is triggered by an overflow or underflow of timer T6.

The capture trigger signal can also be used to clear the contents of timers T5 and T6 individually.

The functions of register CAPREL are controlled via several bit(field)s in the timer control registers T5CON and T6CON.

Capture/Reload Register CAPREL in Capture Mode

Capture Mode for register CAPREL is selected by setting bit T5SC in control register T5CON (set bitfield CI in register T5CON to a non-zero value to select a trigger signal). In Capture Mode, the contents of the auxiliary timer T5 are latched into register CAPREL in response to a signal transition at the selected external input pin(s). Bit CT3 selects the external input line CAPIN or the input lines T3IN and/or T3EUD of GPT1 timer T3 as the source for a capture trigger. Either a positive, a negative, or both a positive and a negative transition at line CAPIN can be selected to trigger the capture function, or transitions on input T3IN or input T3EUD or both inputs, T3IN and T3EUD. The active edge is controlled by bitfield CI in register T5CON. [Table 28-19](#) summarizes these options.

General Purpose Timer Unit (GPT12)

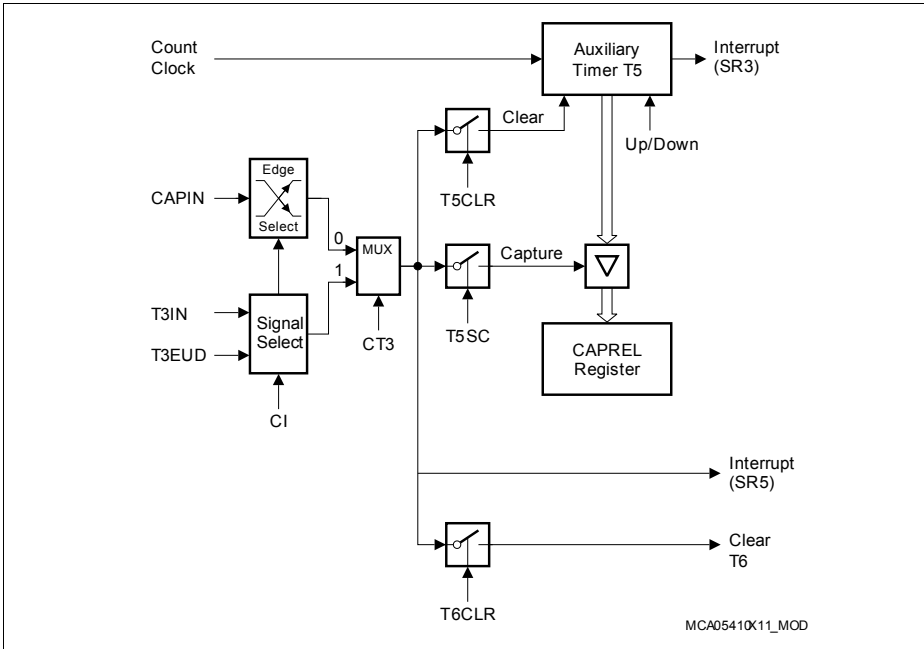


Figure 28-27 Capture/Reload Register CAPREL in Capture Mode

When a selected trigger is detected, the contents of the auxiliary timer T5 are latched into register CAPREL and the service request is activated. The same event can optionally clear timer T5 and/or timer T6. This option is enabled by bit T5CLR in register T5CON and bit T6CLR in register T6CON, respectively. If TxCLR = 0 the contents of timer Tx is not affected by a capture. If TxCLR = 1 timer Tx is cleared after the current timer T5 value has been latched into register CAPREL.

Note: Bit T5SC only controls whether or not a capture is performed. If T5SC is cleared the external input pin(s) can still be used to clear timer T5 and/or T6, or as external interrupt input(s). This interrupt is signalled by the CAPREL interrupt request SR5.

When capture triggers T3IN or T3EUD are enabled (CT3 = 1), register CAPREL captures the contents of T5 upon transitions of the selected input(s). These values can be used to measure T3's input signals. This is useful, for example, when T3 operates in Incremental Interface Mode, in order to derive dynamic information (speed, acceleration) from the input signals.

For Capture Mode operation, the selected pins CAPIN, T3IN, or T3EUD must be configured as input. To ensure that a transition of a trigger input signal applied to one of

General Purpose Timer Unit (GPT12)

these inputs is recognized correctly, its level must be held high or low for a minimum number of module clock cycles, detailed in [Section 28.2.6](#).

Capture/Reload Register CAPREL in Reload Mode

Reload Mode for register CAPREL is selected by setting bit T6SR in control register T6CON. In Reload Mode, the core timer T6 is reloaded with the contents of register CAPREL, triggered by an overflow or underflow of T6. This will not activate the service request SR5 associated with the CAPREL register. However, service request SR4 will be activated, indicating the overflow/underflow of T6.

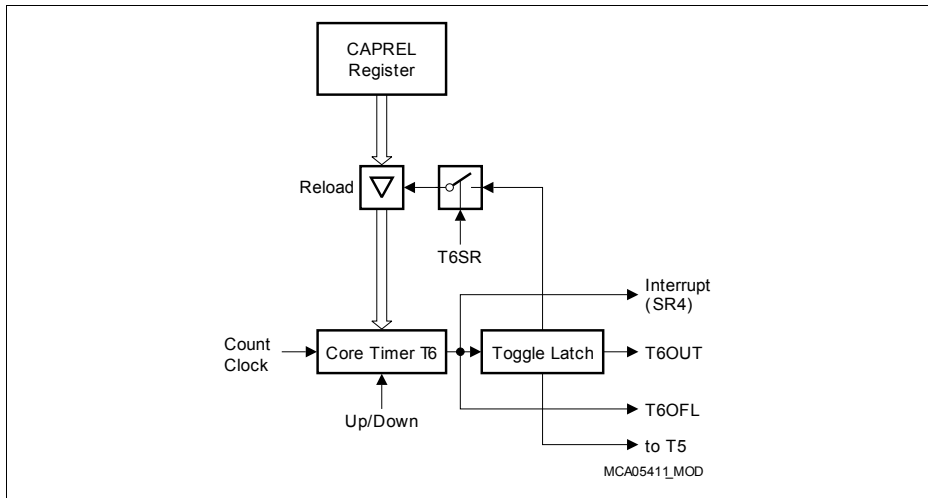


Figure 28-28 Capture/Reload Register CAPREL in Reload Mode

General Purpose Timer Unit (GPT12)

Capture/Reload Register CAPREL in Capture-And-Reload Mode

Since the reload function and the capture function of register CAPREL can be enabled individually by bits T5SC and T6SR, the two functions can be enabled simultaneously by setting both bits. This feature can be used to generate an output frequency that is a multiple of the input frequency.

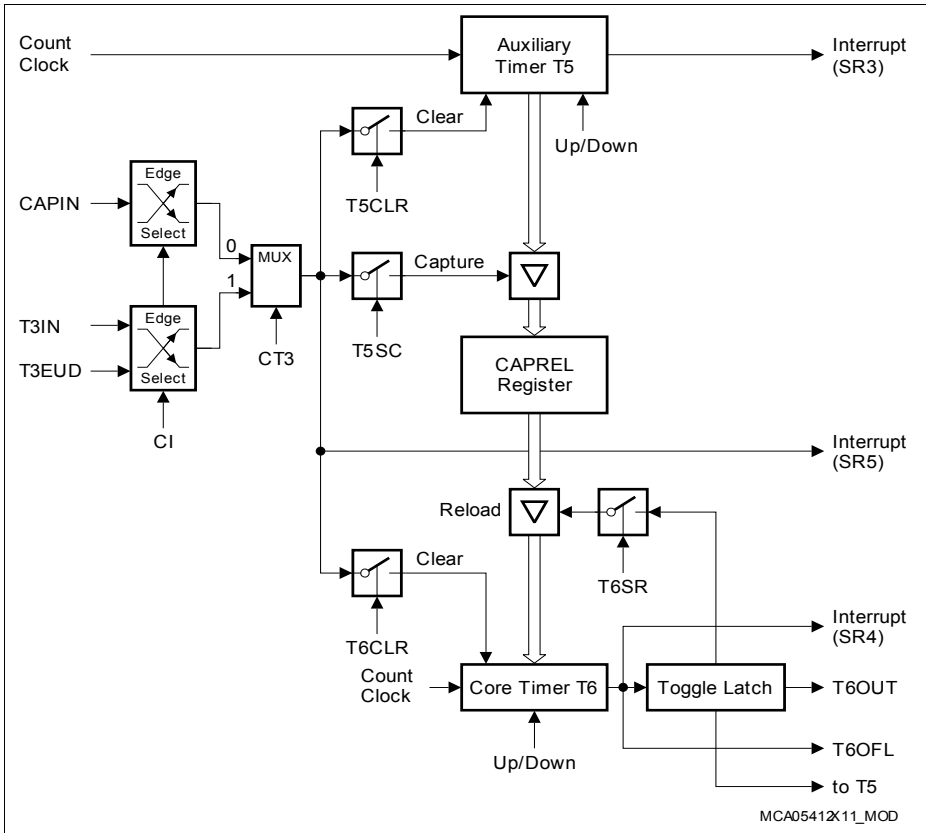


Figure 28-29 Capture/Reload Register CAPREL in Capture-And-Reload Mode

This combined mode can be used to detect consecutive external events which may occur aperiodically, but where a finer resolution, that means, more 'ticks' within the time between two external events is required.

For this purpose, the time between the external events is measured using timer T5 and the CAPREL register. Timer T5 runs in Timer Mode counting up with a frequency of e.g. $f_{GPT}/32$. The external events are applied e.g. to pin CAPIN (or other pins supporting

General Purpose Timer Unit (GPT12)

capture mode, see [Table 28-19](#)). When an external event occurs, the contents of timer T5 are latched into register CAPREL and timer T5 is cleared ($T5CLR = 1$). Thus, register CAPREL always contains the correct time between two events, measured in timer T5 increments. Timer T6, which runs in Timer Mode counting down with a frequency of e.g. $f_{GPT}/4$, uses the value in register CAPREL to perform a reload on underflow. This means, the value in register CAPREL represents the time between two underflows of timer T6, now measured in timer T6 increments. Since (in this example) timer T6 runs 8 times faster than timer T5, it will underflow 8 times within the time between two external events. Thus, the underflow signal of timer T6 generates 8 ‘ticks’. Upon each underflow, the interrupt request SR4 will be activated and bit T6OTL will be toggled. The state of T6OTL may be output on pin T6OUT. This signal on T6OUT has 8 times more transitions than the signal which is applied to the selected input (CAPIN in this example).

Note: The underflow signal of Timer T6 can furthermore be used to clock functional units in other modules, e.g. one or more of the timers of the capture/compare units CCU6 (connections see [Section 28.5.2](#)). This gives the user the possibility to set compare events based on a finer resolution than that of the external events. This connection is accomplished via signal T6OFL.

Capture Correction

A certain deviation of the output frequency is generated by the fact that timer T5 will count actual time units (e.g. T5 running at 1 MHz will count up to the value $64_H/100_D$ for a 10 kHz input signal), while T6OTL will only toggle upon an underflow of T6 (i.e. the transition from 0000_H to $FFFF_H$). In the above mentioned example, T6 would count down from 64_H , so the underflow would occur after 101 timing ticks of T6. The actual output frequency then is 79.2 kHz, instead of the expected 80 kHz.

This deviation can be compensated for by using T6 overflows. In this case, T5 counts down and T6 counts up. Upon a signal transition at the selected input pin(s), e.g. on CAPIN, the count value in T5 is captured into CAPREL and T5 is cleared to 0000_H . In its next clock cycle, T5 underflows to $FFFF_H$, and continues to count down with the following clocks. T6 is reloaded from CAPREL upon an overflow, and continues to count up with its following clock cycles (8 times faster in the above example). In this case, T5 and T6 count the same number of steps with their respective internal count frequency.

In the above example, T5 running at 1 MHz will count down to the value $FF9C_H/-100_D$ for a 10 kHz input signal applied e.g. at CAPIN, while T6 counts up from $FF9C_H$ through $FFFF_H$ to 0000_H . So the overflow occurs after 100 timing ticks of T6, and the actual output frequency at T6OUT then is the expected 80 kHz.

However, in this case CAPREL does not directly contain the time between two external events, but rather its 2’s complement. Software will have to convert this value, if it is required for the operation.

General Purpose Timer Unit (GPT12)

Combined Capture Modes

For incremental interface applications in particular, several timer features can be combined to obtain dynamic information such as speed, acceleration, or deceleration. The current position itself can be obtained directly from the timer register (T2, T3, T4).

The time information to determine the dynamic parameters is generated by capturing the contents of the free-running timer T5 into register CAPREL. Two trigger sources for this event can be selected:

- Capture trigger on sensor signal transitions
- Capture trigger on position read operations

Capturing on sensor signal transitions is available for timer T3 inputs. This mode is selected by setting bit CT3 and selecting the intended signal(s) via bitfield CI in register T5CON. CAPREL then indicates the time between two selected transitions (measured in T5 counts).

Capturing on position read operations is available for timers T2, T3, and T4. This mode is selected by clearing bit CT3 and selecting the rising edge via bitfield CI in register T5CON. Bitfield ISCAPIN in register PISEL then selects either a read access from T3 or a read access from any of T2 or T3 or T4. CAPREL then indicates the time between two read accesses.

In general, GPT12 has no destructive read mechanisms, except for this special operation mode 'capture on position read operations', where register CAPREL is intentionally updated after a read of T3 or T2/T4 (and T5 or T6 are optionally cleared). In this case, also the associated service request flag in register SRC_GPT120CIRQ is set.

Note: If mode 'capturing on position read operations' is selected, and the corresponding timer (T3, or any of T2, T3, T4) is read by a debugger, a capture event may be triggered (T5 is captured into CAPREL, and T5 or T6 may optionally be cleared (if T5CLR = 1 or T6CLR = 1)).

These operating modes directly support the measurement of position and rotational speed. Acceleration and deceleration can then be determined by evaluating subsequent speed measurements.

General Purpose Timer Unit (GPT12)
28.2.6 GPT2 Clock Signal Control

All actions within the timer block GPT2 are triggered by transitions of its basic clock. This basic clock is derived from the module clock f_{GPT} by a basic block prescaler, controlled by bitfield BPS2 in register T6CON (see [Figure 28-18](#)). The count clock can be generated in two different ways:

- **Internal count clock**, derived from GPT2's basic clock via a programmable prescaler, is used for (Gated) Timer Mode.
- **External count clock**, derived from the timer's input pin(s), is used for Counter Mode.

For both ways, the basic clock determines the maximum count frequency and the timer's resolution:

Table 28-12 Basic Clock Selection for Block GPT2

Block Prescaler ¹⁾	BPS2 = 01 _B	BPS2 = 00 _B ²⁾	BPS2 = 11 _B	BPS2 = 10 _B
Prescaling Factor for GPT2: F(BPS2)	F(BPS2) = 2	F(BPS2) = 4	F(BPS2) = 8	F(BPS2) = 16
Maximum External Count Frequency	$f_{GPT}/4$	$f_{GPT}/8$	$f_{GPT}/16$	$f_{GPT}/32$
Input Signal Stable Time	$2 \times t_{GPT}$	$4 \times t_{GPT}$	$8 \times t_{GPT}$	$16 \times t_{GPT}$

1) Please note the non-linear encoding of bitfield BPS2.

2) Default after reset.

Note: The GPT2 module uses a finite state machine to control the actions. Since multiple interactions are possible between the timers (T5, T6) and register CAPREL, these elements are processed sequentially. However, all actions are normally completed within one basic clock cycle. The GPT2 state machine has 4 states (2 states when BPS2 = 01_B) and processes T6 before T5.

Note: When initializing the GPT2 block after reset, and the block prescaler BPS2 in register T6CON needs to be set to a value different from its default value (00_B), it must be initialized first before any mode involving external trigger signals is configured. These modes include counter, capture, and reload mode. Otherwise, unintended count/capture/reload events may occur during the first basic clock cycle.

In this case, or when changing BPS2 during operation of the GPT2 block, disable related interrupts before modification of BPS2, and afterwards clear the corresponding service request flags and re-initialize those registers (T5, T6, CAPREL) that might be affected by a count/capture/reload event.

General Purpose Timer Unit (GPT12)
Internal Count Clock Generation

In Timer Mode and Gated Timer Mode, the count clock for each GPT2 timer is derived from the GPT2 basic clock by a programmable prescaler, controlled by bitfield TxI in the respective timer's control register TxCON.

The count frequency f_{Tx} for a timer Tx and its resolution r_{Tx} are scaled linearly with lower clock frequencies, as can be seen from the following formula:

$$f_{Tx} = \frac{f_{GPT}}{F(BPS2) \times 2^{<TxI>}} \quad r_{Tx}[\mu s] = \frac{F(BPS2) \times 2^{<TxI>}}{f_{GPT}[\text{MHz}]} \quad (28.2)$$

The effective count frequency depends on the common module clock prescaler factor F(BPS2) as well as on the individual input prescaler factor $2^{<TxI>}$. **Table 28-16** summarizes the resulting overall divider factors for a GPT2 timer that result from these cascaded prescalers.

Table 28-13 lists GPT2 timer's parameters (such as count frequency, resolution, and period) resulting from the selected overall prescaler factor $F(BPS2) \times 2^{<TxI>}$ and the module clock f_{GPT} . Note that some numbers may be rounded.

Table 28-13 GPT2 Timer Parameters

Overall Prescaler Factor	Example 1: Module Clock $f_{GPT} = 100 \text{ MHz}$			Example 2: Module Clock $f_{GPT} = 66.5 \text{ MHz}$		
	Frequency	Resolution	Period	Frequency	Resolution	Period
2	50 MHz	20 ns	1.311 ms	33.25 MHz	30.1 ns	1.97 ms
4	25 MHz	40 ns	2.621 ms	16.625 MHz	60.2 ns	3.94 ms
8	12.5 MHz	80 ns	5.243 ms	8.313 MHz	120.3 ns	7.88 ms
16	6.25 MHz	160 ns	10.49 ms	4.156 MHz	240.6 ns	15.77 ms
32	3.125 MHz	320 ns	20.97 ms	2.078 MHz	481.2 ns	31.54 ms
64	1.563 MHz	640 ns	41.94 ms	1.039 MHz	962.4 ns	63.07 ms
128	781.25 kHz	1.28 μs	83.89 ms	519.53 kHz	1.924 μs	126.1 ms
256	390.6 kHz	2.56 μs	167.8 ms	259.77 kHz	3.850 μs	252.3 ms
512	195.3 kHz	5.12 μs	335.5 ms	129.88 kHz	7.699 μs	504.6 ms
1024	97.7 kHz	10.24 μs	671.1 ms	64.94 kHz	15.398 μs	1.009 s
2048	48.8 kHz	20.48 μs	1.342 s	32.47 kHz	30.797 μs	2.018 s

External Count Clock Input

The external input signals of the GPT2 block are sampled with the GPT2 basic clock (see **Figure 28-18**). To ensure that a signal is recognized correctly, its current level (high or

General Purpose Timer Unit (GPT12)

low) must be held active for at least one complete sampling period, before changing. A signal transition is recognized if two subsequent samples of the input signal represent different levels. Therefore, a minimum of two basic clock periods are required for the sampling of an external input signal. Thus, the maximum frequency of an input signal must not be higher than half the basic clock.

Table 28-14 summarizes the resulting requirements for external GPT2 input signals.

Table 28-14 GPT2 External Input Signal Limits

GPT2 Divider BPS2	Input Freq. Factor	Input Phase Duration	Example 1: Module Clock $f_{GPT} =$ 100 MHz		Example 2: Module Clock $f_{GPT} =$ 66.5 MHz	
			Max. Input Frequency	Min. Level Hold Time	Max. Input Frequency	Min. Level Hold Time
01 _B	$f_{GPT}/4$	$2 \times t_{GPT}$	25 MHz	20 ns	16.625 MHz	30.1 ns
00 _B	$f_{GPT}/8$	$4 \times t_{GPT}$	12.5 MHz	40 ns	8.313 MHz	60.2 ns
11 _B	$f_{GPT}/16$	$8 \times t_{GPT}$	6.25 MHz	80 ns	4.156 MHz	120.3 ns
10 _B	$f_{GPT}/32$	$16 \times t_{GPT}$	3.125 MHz	160 ns	2.078 MHz	240.6 ns

These limitations are valid for all external input signals to GPT2, including the external count signals in Counter Mode and the gate input signals in Gated Timer Mode.

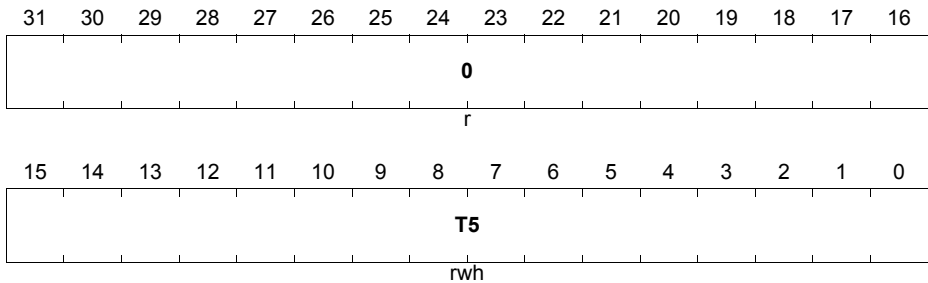
General Purpose Timer Unit (GPT12)

28.2.7 GPT2 Registers

28.2.7.1 GPT2 Timer Registers

T5
Timer T5 Register

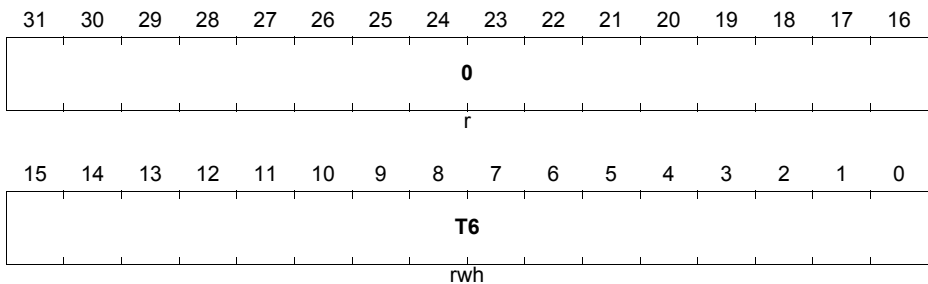
 (40_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
T5	[15:0]	rwh	Timer T5 Contains the current value of Timer T5.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

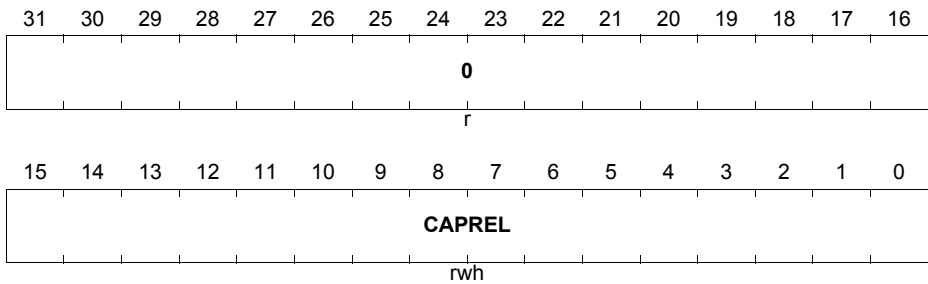
T6
Timer T6 Register

 (44_H)

 Reset Value: 0000 0000_H


General Purpose Timer Unit (GPT12)

Field	Bits	Type	Description
T6	[15:0]	rwh	Timer T6 Contains the current value of Timer T6.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

CAPREL
Capture and Reload Register
(30_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CAPREL	[15:0]	rwh	Current reload value or Captured value
0	[31:16]	r	Reserved Read as 0; should be written with 0.

General Purpose Timer Unit (GPT12)

28.2.7.2 GPT2 Timer Control Registers

GPT2 Core Timer T6 Control Register

T6CON
Timer T6 Control Register

 (20_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0																
r																
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
T6 SR	T6 CLR	0	BPS2	T6 OTL	T6 OE	T6 UDE	T6 UD	T6 R	T6M				T6I			
rw	rw	r	rw	rwh	rw	rw	rw	rw	rw	rw				rw		

Field	Bits	Type	Description
T6I	[2:0]	rw	Timer T6 Input Parameter Selection Depends on the operating mode, see respective sections for encoding: Table 28-16 for Timer Mode and Gated Timer Mode Table 28-17 for Counter Mode
T6M	[5:3]	rw	Timer T6 Mode Control (Basic Operating Mode) 000 _B Timer Mode 001 _B Counter Mode 010 _B Gated Timer Mode with gate active low 011 _B Gated Timer Mode with gate active high 100 _B Reserved. Do not use this combination. 101 _B Reserved. Do not use this combination. 110 _B Reserved. Do not use this combination. 111 _B Reserved. Do not use this combination.
T6R	6	rw	Timer T6 Run Bit 0 _B Timer T6 stops 1 _B Timer T6 runs

General Purpose Timer Unit (GPT12)

Field	Bits	Type	Description
T6UD	7	rw	Timer T6 Up/Down Control¹⁾ 0 _B Timer T6 counts up 1 _B Timer T6 counts down <i>Note: This bit only controls count direction of T6 if bit T6UDE = 0.</i>
T6UDE	8	rw	Timer T6 External Up/Down Enable¹⁾ 0 _B Count direction is controlled by bit T6UD; input T6EUD is disconnected 1 _B Count direction is controlled by input T6EUD
T6OE	9	rw	Overflow/Underflow Output Enable 0 _B Alternate Output Function Disabled 1 _B State of timer T6 toggle latch is output on pin T6OUT
T6OTL	10	rwh	Timer T6 Overflow Toggle Latch Toggles on each overflow/underflow of timer T6. Can be set or cleared by software (see separate description)
BPS2	[12:11]	rw	GPT2 Block Prescaler Control Selects the basic clock for block GPT2 (see also Section 28.2.6) 00 _B $f_{GPT}/4$ 01 _B $f_{GPT}/2$ 10 _B $f_{GPT}/16$ 11 _B $f_{GPT}/8$
T6CLR	14	rw	Timer T6 Clear Enable Bit 0 _B Timer T6 is not cleared on a capture event 1 _B Timer T6 is cleared on a capture event
T6SR	15	rw	Timer T6 Reload Mode Enable 0 _B Reload from register CAPREL Disabled 1 _B Reload from register CAPREL Enabled
0	13, [31:16]	r	Reserved Read as 0; should be written with 0.

 1) See [Table 28-15](#) for coding of bits T6UD and T6UDE

General Purpose Timer Unit (GPT12)

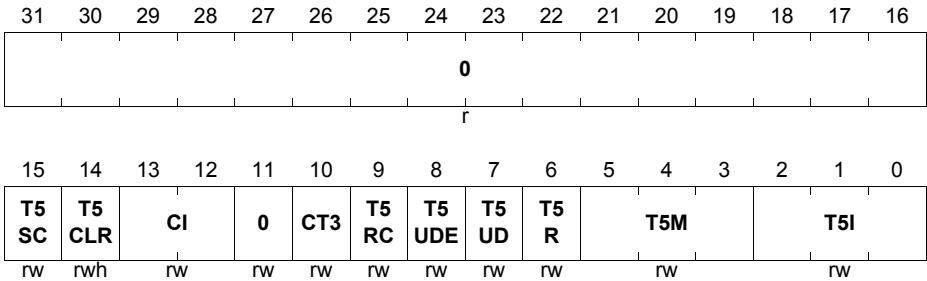
GPT2 Auxiliary Timer T5 Control Registers

T5CON

Timer T5 Control Register

(1C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
T5I	[2:0]	rw	Timer T5 Input Parameter Selection Depends on the operating mode, see respective sections for encoding: Table 28-16 for Timer Mode and Gated Timer Mode Table 28-18 for Counter Mode
T5M	[5:3]	rw	Timer T5 Mode Control (Basic Operating Mode) 000 _B Timer Mode 001 _B Counter Mode 010 _B Gated Timer Mode with gate active low 011 _B Gated Timer Mode with gate active high 100 _B Reserved. Do not use this combination 101 _B Reserved. Do not use this combination 110 _B Reserved. Do not use this combination 111 _B Reserved. Do not use this combination
T5R	6	rw	Timer T5 Run Bit 0 _B Timer T5 runs 1 _B Timer T5 stops <i>Note: This bit only controls timer T5 if bit T5RC = 0.</i>
T5UD	7	rw	Timer T5 Up/Down Control¹⁾ 0 _B Timer T5 counts up 1 _B Timer T5 counts down <i>Note: This bit only controls count direction of T5 if bit T5UDE = 0.</i>

General Purpose Timer Unit (GPT12)

Field	Bits	Type	Description
T5UDE	8	rw	Timer T5 External Up/Down Enable¹⁾ 0 _B Count direction is controlled by bit T5UD; input T5EUD is disconnected 1 _B Count direction is controlled by input T5EUD
T5RC	9	rw	Timer T5 Remote Control 0 _B Timer T5 is controlled by its own run bit T5R 1 _B Timer T5 is controlled by the run bit T6R of core timer T6, not by bit T5R
CT3	10	rw	Timer T3 Capture Trigger Enable 0 _B Capture trigger from input line CAPIN 1 _B Capture trigger from T3 input lines T3IN and/or T3EUD
0	11	rw	Reserved Has to be written with 0.
CI	[13:12]	rw	Register CAPREL Capture Trigger Selection²⁾ 00 _B Capture disabled 01 _B Positive transition (rising edge) on CAPIN ³⁾ or any transition on T3IN 10 _B Negative transition (falling edge) on CAPIN or any transition on T3EUD 11 _B Any transition (rising or falling edge) on CAPIN or any transition on T3IN or T3EUD
T5CLR	14	rw	Timer T5 Clear Enable Bit 0 _B Timer T5 is not cleared on a capture event 1 _B Timer T5 is cleared on a capture event
T5SC	15	rw	Timer T5 Capture Mode Enable 0 _B Capture into register CAPREL disabled 1 _B Capture into register CAPREL enabled
0	[31:16]	r	Reserved Read as 0; should be written with 0.

1) See [Table 28-15](#) for encoding of bits T5UD and T5UDE.

2) To define the respective trigger source signal, also bit CT3 must be regarded (see [Table 28-19](#)).

3) Rising edge must be selected for triggering the capture/clear operation by the internal GPT1 read signals (see bit field ISCAPIN in register [PISEL](#) and description in section [“Combined Capture Modes” on Page 28-58](#)).

General Purpose Timer Unit (GPT12)
Encoding of GPT2 Timer Count Direction Control
Table 28-15 GPT2 Timer Count Direction Control

Pin TxEUD	Bit TxUDE	Bit TxUD	Count Direction
X	0	0	Count Up
X	0	1	Count Down
0	1	0	Count Up
1	1	0	Count Down
0	1	1	Count Down
1	1	1	Count Up

Encoding of GPT2 Overall Prescaler Factor in Timer Mode and Gated Timer Mode
Table 28-16 GPT2 Overall Prescaler Factors for Internal Count Clock (Timer Mode and Gated Timer Mode)

Individual Prescaler for Tx	Common Prescaler for Module Clock ¹⁾			
	BPS2 = 01 _B	BPS2 = 00 _B	BPS2 = 11 _B	BPS2 = 10 _B
Txl = 000 _B	2	4	8	16
Txl = 001 _B	4	8	16	32
Txl = 010 _B	8	16	32	64
Txl = 011 _B	16	32	64	128
Txl = 100 _B	32	64	128	256
Txl = 101 _B	64	128	256	512
Txl = 110 _B	128	256	512	1024
Txl = 111 _B	256	512	1024	2048

1) Please note the non-linear encoding of bitfield BPS2.

General Purpose Timer Unit (GPT12)
Encoding of GPT2 Input Edge Selection in Counter Mode
Table 28-17 GPT2 Core Timer T6 (Counter Mode) Input Edge Selection

T6I	Triggering Edge for Counter Increment/Decrement
000 _B	None. Counter T6 is disabled
001 _B	Positive transition (rising edge) on T6IN
010 _B	Negative transition (falling edge) on T6IN
011 _B	Any transition (rising or falling edge) on T6IN
1XX _B	Reserved. Do not use this combination

Table 28-18 GPT2 Auxiliary Timer T5 (Counter Mode) Input Edge Selection

T5I	Triggering Edge for Counter Increment/Decrement
X00 _B	None. Counter T5 is disabled
001 _B	Positive transition (rising edge) on T5IN
010 _B	Negative transition (falling edge) on T5IN
011 _B	Any transition (rising or falling edge) on T5IN
101 _B	Positive transition (rising edge) of T6 toggle latch T6OTL
110 _B	Negative transition (falling edge) of T6 toggle latch T6OTL
111 _B	Any transition (rising or falling edge) of T6 toggle latch T6OTL

General Purpose Timer Unit (GPT12)

Encoding of GPT2 Input Selection for Capture and Timer Clear Function

Table 28-19 CAPREL Register Input Edge Selection

CT3	CI	Triggering Signal/Edge for Capture Mode and/or T5/T6 Clear
X	00 _B	None. Capture Mode is disabled.
0	01 _B	Positive transition (rising edge) on CAPIN, or read operation on selected GPT1 timers ¹⁾ .
0	10 _B	Negative transition (falling edge) on CAPIN.
0	11 _B	Any transition (rising or falling edge) on CAPIN.
1	01 _B	Any transition (rising or falling edge) on T3IN.
1	10 _B	Any transition (rising or falling edge) on T3EUD.
1	11 _B	Any transition (rising or falling edge) on T3IN or T3EUD.

1) Rising edge must be selected for triggering the capture/clear operation by the internal GPT1 read signals (see bit field ISCAPIN in register **PISEL** and description in section **“Combined Capture Modes” on Page 28-58**).

General Purpose Timer Unit (GPT12)
28.3 GPT12 Kernel Register Overview

Table 28-20 summarizes the GPT12 kernel registers and the module external registers and defines their addresses and reset values.

BPI registers are included in a separate **Table 28-21**.

Table 28-20 Register Overview of GPT12

Register Short Name	Register Long Name	Offset Addr. ¹⁾	Access Mode		Reset	Page Num.
			Read	Write		
PISEL	Port Input Select Register	04 _H	U, SV	U, SV, P	Application Reset	28-73
ID	Identification Register	08 _H	U, SV	BE	Application Reset	28-75
T2CON	Timer T2 Control Register	10 _H	U, SV	U, SV, P	Application Reset	28-32
T3CON	Timer T3 Control Register	14 _H	U, SV	U, SV, P	Application Reset	28-30
T4CON	Timer T4 Control Register	18 _H	U, SV	U, SV, P	Application Reset	28-34
T5CON	Timer T5 Control Register	1C	U, SV	U, SV, P	Application Reset	28-66
T6CON	Timer T6 Control Register	20 _H	U, SV	U, SV, P	Application Reset	28-64
CAPREL	Capture and Reload Register	30 _H	U, SV	U, SV, P	Application Reset	28-63
T2	Timer T2 Register	34 _H	U, SV	U, SV, P	Application Reset	28-28
T3	Timer T3 Register	38 _H	U, SV	U, SV, P	Application Reset	28-28
T4	Timer T4 Register	3C _H	U, SV	U, SV, P	Application Reset	28-29
T5	Timer T5 Register	40 _H	U, SV	U, SV, P	Application Reset	28-62
T6	Timer T6 Register	44 _H	U, SV	U, SV, P	Application Reset	28-62

1) The absolute register address is calculated as follows:
Module Base Address + Offset Address (shown in this column)

General Purpose Timer Unit (GPT12)

28.4 General Module Operation

This section provides information about the:

- Input Selection (see [Section 28.4.1](#))
- OCDS Suspend Functionality (see [Section 28.4.2](#))
- Miscellaneous GPT12 Register Description (see [Section 28.4.3](#))
- BPI Register Description (see [Section 28.4.4](#))

28.4.1 Input Selection

Each GPT12 input signal can be selected from a vector of two or four possible inputs by programming the port input select register **PISEL**. This permits to adapt the pin functionality of the device to the application requirements.

The output pins for the module output signals are chosen in the ports.

Naming convention (example):

The input vector T3IN[D:A] for input signal T3IN is composed of the signals T3INA to T3IND.

Note: All functional inputs of the GPT12 module are synchronized to the internal basic clock of the GPT1 and GPT2 block, respectively. An edge of an input signal can only be correctly recognized if the high phase and the low phase are longer than one basic clock period. See [Table 28-5](#) for GPT1 external input signal limits, and [Table 28-14](#) for GPT2 external input signal limits.

28.4.2 OCDS Suspend

The behavior of GPT12 upon an OCDS suspend request is controlled by the **OCS** register. GPT12 supports only Hard Suspend Mode.

Hard Suspend Mode

In Hard Suspend Mode the GPT12 kernel clock is switched off immediately. Reading and writing of registers is possible but will enable the kernel clock for a few cycles.

Attention: Register accesses with clocking in Hard Suspend Mode can have unintended side effects like signals becoming and staying active. This can affect also other modules, so a GPT12 kernel reset might not be sufficient to bring the system into a defined state.

General Purpose Timer Unit (GPT12)

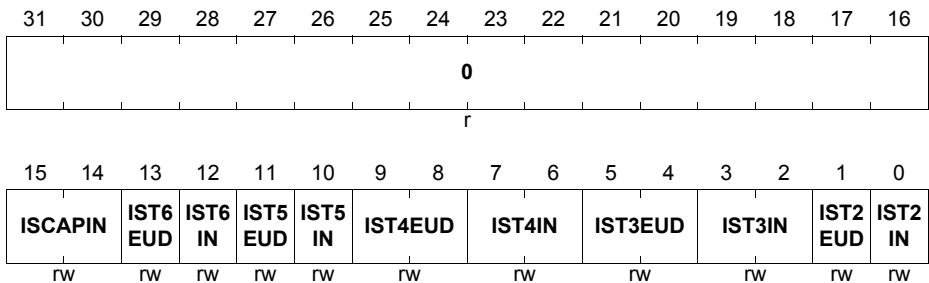
28.4.3 Miscellaneous GPT12 Registers

28.4.3.1 Port Input Select Register

Register PISEL contains bit fields selecting the actual input signal for the module inputs.

PISEL
Port Input Select Register

 (04_H)

 Reset Value: 0000 0000_H


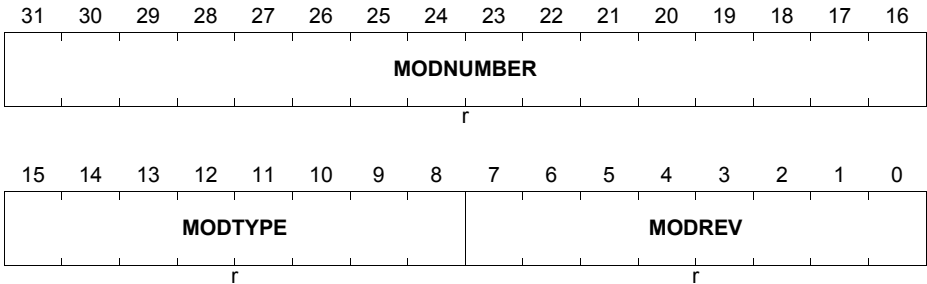
Field	Bits	Type	Description
IST2IN	0	rw	Input Select for T2IN 0 _B Signal T2INA is selected 1 _B Signal T2INB is selected
IST2EUD	1	rw	Input Select for T2EUD 0 _B Signal T2EUDA is selected 1 _B Signal T2EUDB is selected
IST3IN	[3:2]	rw	Input Select for T3IN 00 _B Signal T3INA is selected 01 _B Signal T3INB is selected 10 _B Signal T3INC is selected 11 _B Signal T3IND is selected
IST3EUD	[5:4]	rw	Input Select for T3EUD 00 _B Signal T3EUDA is selected 01 _B Signal T3EUDB is selected 10 _B Signal T3EUDC is selected 11 _B Signal T3EUDD is selected

General Purpose Timer Unit (GPT12)

Field	Bits	Type	Description
IST4IN	[7:6]	rw	Input Select for T4IN 00 _B Signal T4INA is selected 01 _B Signal T4INB is selected 10 _B Signal T4INC is selected 11 _B Signal T4IND is selected
IST4EUD	[9:8]	rw	Input Select for T4EUD 00 _B Signal T4EUDA is selected 01 _B Signal T4EUDB is selected 10 _B Signal T4EUDC is selected 11 _B Signal T4EUDD is selected
IST5IN	10	rw	Input Select for T5IN 0 _B Signal T5INA is selected 1 _B Signal T5INB is selected
IST5EUD	11	rw	Input Select for T5EUD 0 _B Signal T5EUDA is selected 1 _B Signal T5EUDB is selected
IST6IN	12	rw	Input Select for T6IN 0 _B Signal T6INA is selected 1 _B Signal T6INB is selected
IST6EUD	13	rw	Input Select for T6EUD 0 _B Signal T6EUDA is selected 1 _B Signal T6EUDB is selected
ISCAPIN	[15:14]	rw	Input Select for CAPIN 00 _B Signal CAPINA is selected 01 _B Signal CAPINB is selected 10 _B Signal CAPINC (Read trigger from T3) is selected 11 _B Signal CAPIND (Read trigger from T2 or T3 or T4) is selected
0	[31:16]	r	Reserved Read as 0; should be written with 0.

28.4.3.2 Identification Register

The GPT12 Module Identification Register ID contains read-only information about the module identification number and its revision.

General Purpose Timer Unit (GPT12)
ID
Identification Register
(08_H)
Reset Value: 0068 C0XX_H


Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number This bit field indicates the revision number of the TC26x B-Step module (01 _H = first revision).
MODTYPE	[15:8]	r	Module Type This bit field is C0 _H . It defines a 32-bit module
MODNUMBER	[31:16]	r	Module Number This bit field defines the module identification number. For the GPT12 module the module identification number is 68 _H .

General Purpose Timer Unit (GPT12)

28.4.4 BPI Registers

This section describes the registers of the BPI (Bus Peripheral Interface). [Figure 28-30](#) shows all registers associated with the BPI of a GPT12 module.

BPI Registers Overview

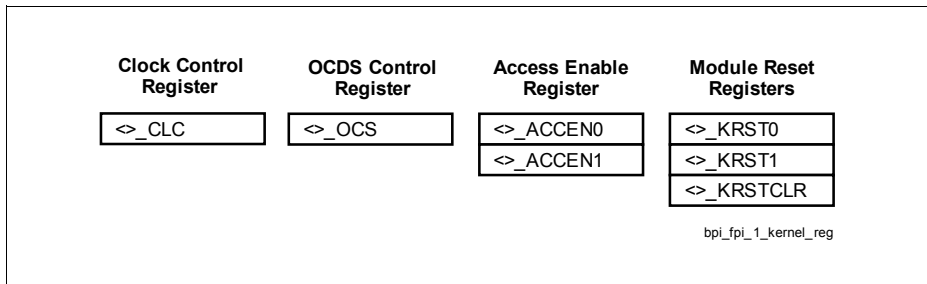


Figure 28-30 BPI Registers

[Table 28-21](#) gives an overview of the BPI registers, including their access modes and reset class.

Table 28-21 Registers Overview - BPI Registers

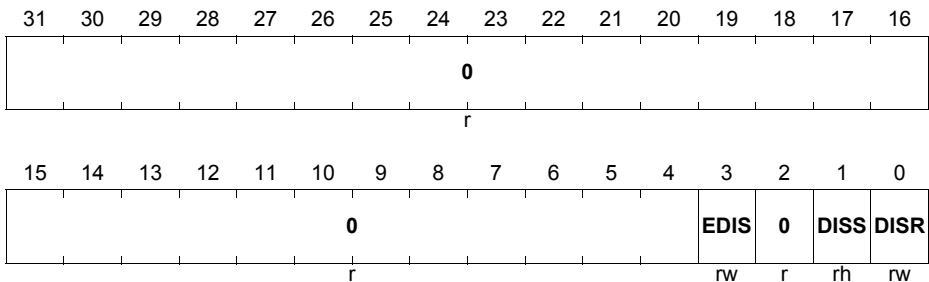
Register Short Name	Description	Offset Addr.	Access Mode		Reset	Page Num.
			Read	Write		
CLC	Clock Control Register	00 _H	U, SV	SV, E, P	Application Reset	28-78
-	Kernel Registers	-	-	-	-	-
OCS	OCDS Control and Status Register	E8 _H	U, SV	SV, P	Debug Reset	28-79
KRSTCLR	Reset Status Clear Register	EC _H	U, SV	SV, E, P	Application Reset	28-84
KRST1	Reset Control Register 1	F0 _H	U, SV	SV, E, P	Application Reset	28-83
KRST0	Reset Control Register 0	F4 _H	U, SV	SV, E, P	Application Reset	28-82
ACCEN1	Access Enable Register 1	F8 _H	U, SV	SV, SE	Application Reset	28-81
ACCEN0	Access Enable Register 0	FC _H	U, SV	SV, SE	Application Reset	28-80

General Purpose Timer Unit (GPT12)

Note: Register bits marked "r" in the following register description are virtual registers and do not contain flip-flops. They are always read as 0.

General Purpose Timer Unit (GPT12)
28.4.4.1 System Registers
Clock Control Register (CLC)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI for the GPT12 module. Where a module kernel is connected to the CLC clock control interface, CLC controls the f_{GPT} module clock signal and Sleep Mode for the module.

CLC
Clock Control Register
(00_H)
Reset Value: 0000 0003_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module. 0 _B Module disable is not requested. 1 _B Module disable is requested.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module. 0 _B Module is enabled. 1 _B Module is disabled.
EDIS	3	rw	Sleep Mode Enable Control Used to control module's sleep mode. 0 _B Sleep Mode request is regarded. Module is enabled to go into Sleep Mode. 1 _B Sleep Mode request is disregarded: Sleep Mode cannot be entered upon a request.

General Purpose Timer Unit (GPT12)

Field	Bits	Type	Description
0	[31:16], [15:4], 2	r	Reserved Read as 0; should be written with 0.

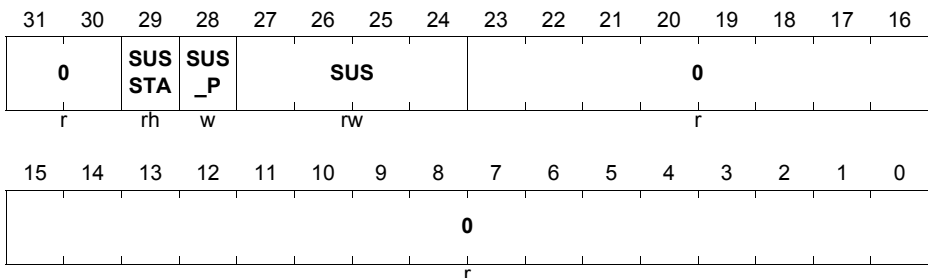
Note: Upon an accepted Sleep Mode request (with EDIS = '1'), or upon a disable request (DISR = '1'), the GPT12 kernel clock is switched off immediately. Therefore, software should ensure that the system controlled by the GPT12 kernel has reached a safe state before triggering a Sleep Mode or module disable request.

OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The OCS register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS register value will not change. When OCDS is disabled the OCS suspend control is ineffective.

Write access is 32 bit wide only and requires Supervisor Mode.

OCS
OCDS Control and Status Register (E8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately. others, reserved
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.

General Purpose Timer Unit (GPT12)

Field	Bits	Type	Description
SUSSTA	29	rh	Suspend State 0_B Module is not (yet) suspended 1_B Module is suspended
0	[23:0], [31:30]	r	Reserved Read as 0; must be written with 0.

Access Enable Register 0 (ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B , EN1 -> TAG ID 000001_B , ... , EN31 -> TAG ID 011111_B .

ACCEN0
Access Enable Register 0
(FC_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

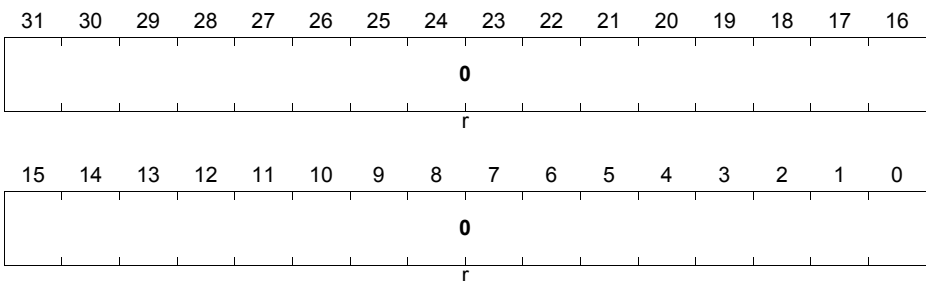
Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0_B Write access will not be executed 1_B Write access will be executed

1) The BPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers.

General Purpose Timer Unit (GPT12)
Access Enable Register 1 (ACCEN1)

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 10000_B to 11111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 10000_B, EN1 -> TAG ID 10001_B, ... , EN31 -> TAG ID 11111_B.

ACCEN1
Access Enable Register 1
(F8_H)
Reset Value: 0000 0000_H


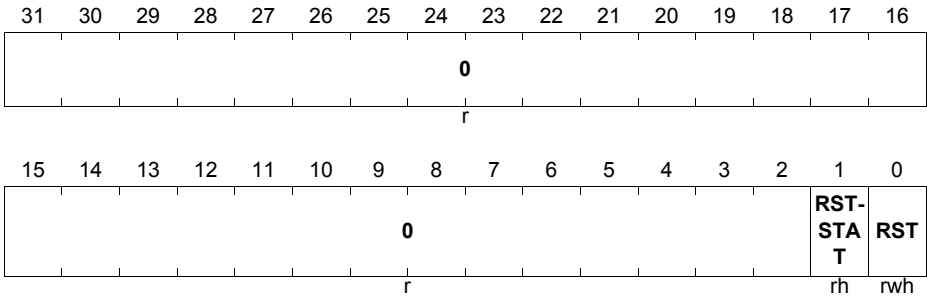
Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set (cleared to '0') by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI in the same clock cycle the RST bit is re-set by the BPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLR.CLR register bit.

Note: During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

General Purpose Timer Unit (GPT12)
KRST0
Kernel Reset Register 0
(F4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI after the kernel reset was executed.
RSTSTAT	1	rw	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI after the execution of a kernel reset in the same clock cycle both reset bits. 0 _B No kernel reset was executed 1 _B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set (cleared to '0') by the BPI with the end of the BPI kernel reset sequence.

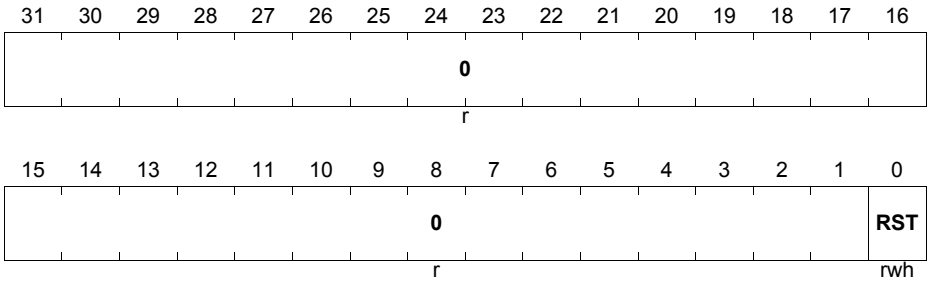
General Purpose Timer Unit (GPT12)

KRST1

Kernel Reset Register 1

(F0_H)

Reset Value: 0000 0000_H

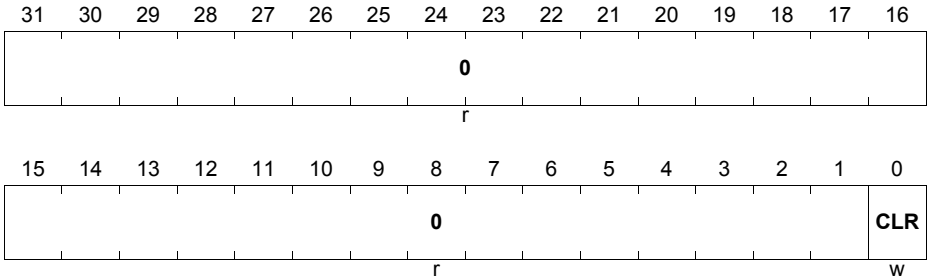


Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set.</p> <p>0_B No kernel reset was requested</p> <p>1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI after the kernel reset was executed.</p>
0	[31:1]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Kernel Reset Status Clear Register (KRSTCLR)

The Kernel Reset Status Clear Register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

General Purpose Timer Unit (GPT12)

KRSTCLR
Kernel Reset Status Clear Register (EC_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

General Purpose Timer Unit (GPT12)
28.5 Implementation of the GPT12 Module

This chapter describes the implementation of the GPT12 module in the TC26x B-Step device.

28.5.1 Address Map

There is one GPT12 kernel implemented in the TC26x B-Step, namely GPT120.

Table 28-22 Registers Address Space

Module	Base Address	End Address	Note
GPT120	F000 2E00 _H	F000 2EFF _H	

28.5.2 Module Connections

The following table shows the digital connections of the GPT12 module with other modules or pins in the TC26x B-Step device.

The GPT12 module is clocked with the SPB_Bus clock, so $f_{GPT} = f_{SPB}$.

Note: Signals that are not available in the 144-pin package variant are marked (see footnote ¹⁾).

Table 28-23 GPT120 Digital Connections in TC26x B-Step

Signal	from/to Module	I/O to GPT120	Can be used to/as
T2INA	P00.7	I	count input signals for timer T2
T2INB	P33.7	I	
T2EUDA	P00.8	I	direction input signals for timer T2
T2EUSB	P33.6	I	
T3INA	P02.6	I	count input signals for timer T3
T3INB	P10.4 ¹⁾	I	
T3INC	ERU_PDOUT4	I	
T3IND	0	I	
T3EUDA	P02.7	I	direction input signals for timer T3
T3EUSB	P10.7 ¹⁾	I	
T3EUDC	0	I	
T3EUDD	0	I	

General Purpose Timer Unit (GPT12)
Table 28-23 GPT120 Digital Connections in TC26x B-Step (cont'd)

Signal	from/to Module	I/O to GPT120	Can be used to/as
T3OUT	P10.6 P21.6 ERU_IN42	O	count output signal for timer T3
T4INA	P02.8	I	count input signals for timer T4
T4INB	P10.8 ¹⁾	I	
T4INC	0	I	
T4IND	0	I	
T4EUDA	P00.9	I	direction input signals for timer T4
T4EUSB	P33.5	I	
T4EUDC	0	I	
T4EUDD	0	I	
T5INA	P21.7	I	count input signals for timer T5
T5INB	P10.3	I	
T5EUDA	P21.6	I	direction input signals for timer T5
T5EUSB	P10.1	I	
T6INA	P20.3	I	count input signals for timer T6
T6INB	P10.2	I	
T6EUDA	P20.0	I	direction input signals for timer T6
T6EUSB	P10.0 ¹⁾	I	
T6OUT	P10.5 P21.7 ERU_IN52	O	count output signal for timer T6
T6OFL	CCU60_T12HRF CCU60_T13HRF CCU61_T12HRF CCU61_T13HRF	O	over/under-flow signal from timer T6
CAPINA	P13.2	I	capture/timer clear trigger input signals
CAPINB	ERU_PDOUT6	I	
SR0	SRC_GPT120T2	O	GPT120 Timer 2 Service Request
SR1	SRC_GPT120T3	O	GPT120 Timer 3 Service Request
SR2	SRC_GPT120T4	O	GPT120 Timer 4 Service Request

General Purpose Timer Unit (GPT12)

Table 28-23 GPT120 Digital Connections in TC26x B-Step (cont'd)

Signal	from/to Module	I/O to GPT120	Can be used to/as
SR3	SRC_GPT120T5	O	GPT120 Timer 5 Service Request
SR4	SRC_GPT120T6	O	GPT120 Timer 6 Service Request
SR5	SRC_GPT120CIRQ	O	GPT120 CAPREL Service Request

1) not available for devices in 144-pin package

Versatile Analog-to-Digital Converter (VADC)

29 Versatile Analog-to-Digital Converter (VADC)

The TC26x B-Step provides a series of analog input channels connected to a cluster of Analog/Digital Converters using the Successive Approximation Register (SAR) principle to convert analog input values (voltages) to discrete digital values.

The TC26x B-Step is based on individual SAR converters with dedicated Sample&Hold units.

The number of analog input channels and ADCs depends on the chosen product type (refer to **“Product-Specific Configuration”** on Page 29-147).

Each converter of the ADC cluster can operate independent of the others, controlled by a dedicated set of registers and triggered by a dedicated group request source. The results of each channel can be stored in a dedicated channel-specific result register or in a group-specific result register.

A background request source can access all analog input channels that are not assigned to any group request source. These conversions are executed with low priority. The background request source can, therefore, be regarded as an additional background converter.

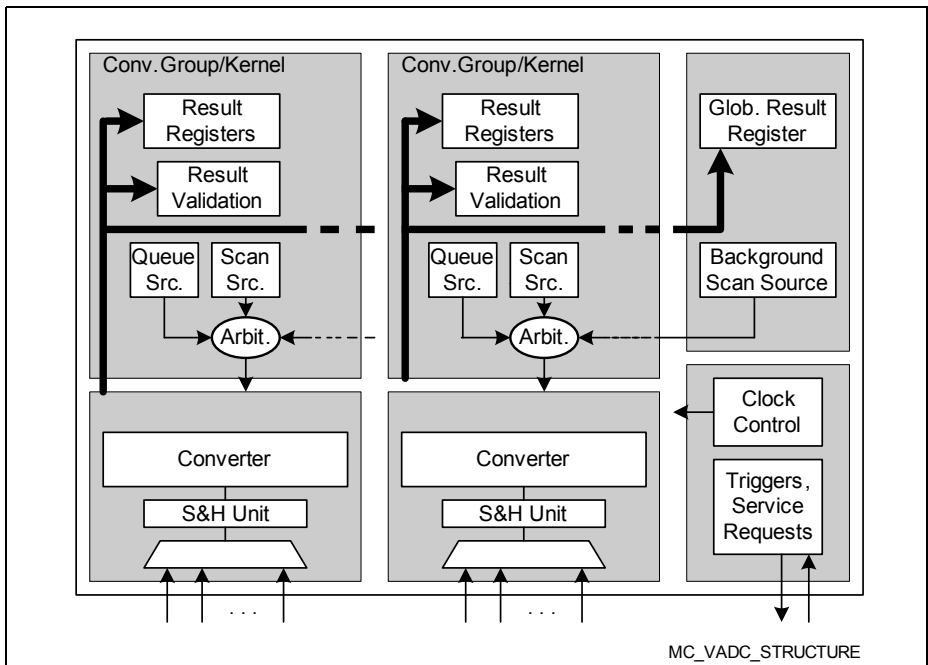


Figure 29-1 ADC Structure Overview

Versatile Analog-to-Digital Converter (VADC)

You will find the following major sections within this chapter:

- **“Introduction and Basic Structure” on Page 29-4**
- **“Configuration of General Functions” on Page 29-14**
- **“Analog Module Activation and Control” on Page 29-30**
- **“Conversion Request Generation” on Page 29-32**
 - **“Queued Request Source Handling” on Page 29-34**
 - **“Channel Scan Request Source Handling” on Page 29-50**
- **“Request Source Arbitration” on Page 29-66**
- **“Analog Input Channel Configuration” on Page 29-74**
- **“Conversion Result Handling” on Page 29-94**
- **“Synchronization of Conversions” on Page 29-117**
- **“Safety Features” on Page 29-123**
- **“External Multiplexer Control” on Page 29-128**
- **“Service Request Generation” on Page 29-133**
- **“Implementation into the TC26x B-Step” on Page 29-147**
including a **“Summary of Registers and Locations” on Page 29-149**
- **“Use Case Example for VADC” on Page 29-165**

Feature List

The following features describe the functionality of the ADC cluster:

- Nominal analog supply voltage 5.0 V, operation at 3.3 V with reduced performance
- Input voltage range from 0 V up to analog supply voltage
- Standard (V_{AREF}) and alternate (CH0) reference voltage source selectable for each channel to support ratiometric measurements and different signal scales
- Up to 4 independent converters with up to 16 analog input channels
- External analog multiplexer control, including adjusted sample time and scan support
- Conversion speed and sample time adjustable to adapt to sensors and reference
- Conversion time below 1 μ s (depending on result width and sample time)
- Flexible source selection and arbitration
 - Programmable arbitrary conversion sequence (single or repeated)
 - Configurable auto scan conversion (single or repeated) on each converter
 - Configurable auto scan conversion (single or repeated) in the background (all converters)
 - Conversions triggered by software, timer events, or external events
 - Cancel-inject-restart mode for reduced conversion delay on priority channels
- Powerful result handling
 - Selectable result width of 8/10/12 bits
 - Fast Compare Mode
 - Independent result registers
 - Configurable limit checking against programmable border values
 - Data rate reduction through adding a selectable number of conversion results
 - FIR/IIR filter with selectable coefficients

Versatile Analog-to-Digital Converter (VADC)

- Flexible service request generation based on selectable events
- Built-in safety features
 - Configurable register access protection supports safety applications
 - Broken wire detection with programmable default levels
 - Multiplexer test mode to verify signal path integrity
- Support of suspend and power saving modes

Table 29-1 Abbreviations used in ADC chapter

Abbreviation	Meaning
ADC	Analog to Digital Converter
DMA	Direct Memory Access (controller)
DNL	Differential Non-Linearity (error)
INL	Integral Non-Linearity (error)
LSB _n	Least Significant Bit: finest granularity of the analog value in digital format, represented by one least significant bit of the conversion result with n bits resolution (measurement range divided in 2 ⁿ equally distributed steps)
SCU	System Control Unit of the device
TUE	Total Unadjusted Error

Versatile Analog-to-Digital Converter (VADC)

29.1 Introduction and Basic Structure

The Versatile Analog to Digital Converter module (VADC) of the TC26x B-Step comprises a set of converter blocks that can be operated either independently or via a common request source that emulates a background converter. Each converter block is equipped with a dedicated input multiplexer and dedicated request sources, which together build separate groups, each assigned to a kernel.

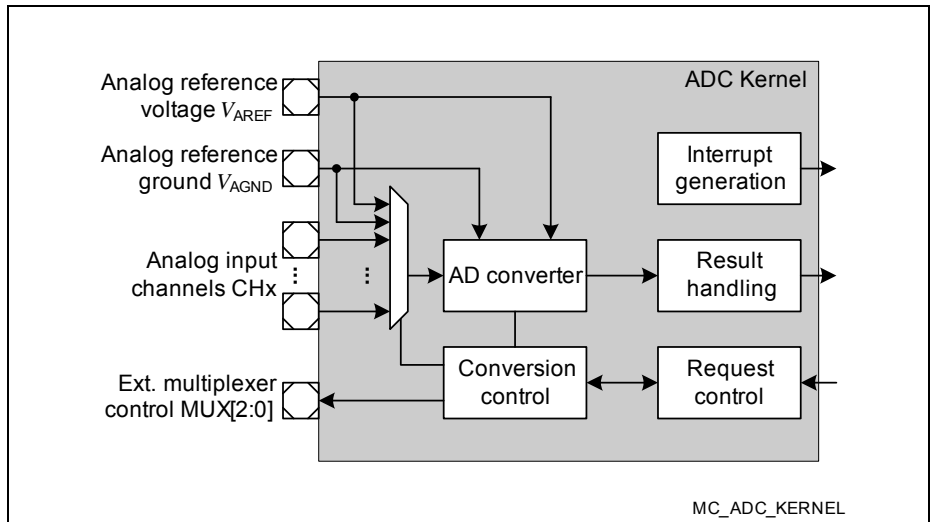


Figure 29-2 ADC Kernel Block Diagram

This basic structure supports application-oriented programming and operating while still providing general access to all resources. The almost identical converter groups allow a flexible assignment of functions to channels.

A set of functional units (described further down in this section) can be configured according to the requirements of a given application. These units build a path from the input signals to the digital results.

Each kernel provides a dedicated Sample&Hold unit connected to the input multiplexer.

The basic module clock f_{VADC} is connected to the system clock signal f_{SPB} .

Versatile Analog-to-Digital Converter (VADC)

Conversion Modes and Request Sources

Analog/Digital conversions can be requested by several request sources (2 group request sources and the background request source) and can be executed in several conversion modes. The request sources can be enabled concurrently with configurable priorities.

- **Fixed Channel Conversion (single or continuous)**

A specific channel source requests conversions of one selectable channel (once or repeatedly)

- **Auto Scan Conversion (single or continuous)**

A channel scan source (request source 1 or 2) requests auto scan conversions of a configurable linear sequence of all available channels (once or repeatedly)

- **Channel Sequence Conversion (single or continuous)**

A queued source (request source 0) requests a sequence of conversions of up to 8 arbitrarily selectable channels (once or repeatedly)

The conversion modes can be used concurrently by the available request sources, i.e. conversions in different modes can be enabled at the same time. Each source can be enabled separately and can be triggered by external events, such as edges of PWM or timer signals, or pin transitions.

Request Source Control

Because all request sources can be enabled at the same time, an arbiter resolves concurrent conversion requests from different sources. Each source can be triggered by external signals, by on-chip signals, or by software.

Requests with higher priority can either cancel a running lower-priority conversion (cancel-inject-repeat mode) or be converted immediately after the currently running conversion (wait-for-start mode). If the target result register has not been read, a conversion can be deferred (wait-for-read mode).

Certain channels can also be synchronized with other ADC kernels, so several signals can be converted in parallel.

Versatile Analog-to-Digital Converter (VADC)

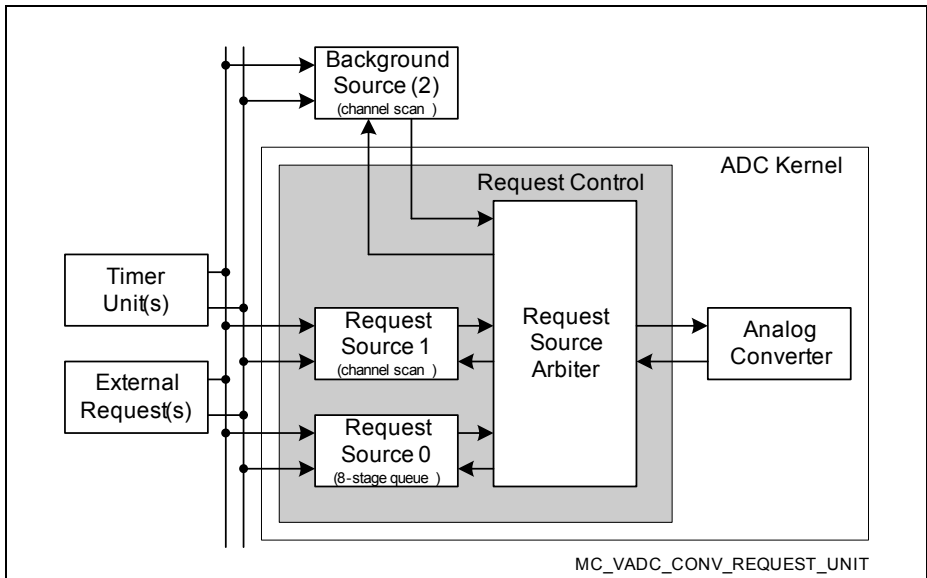


Figure 29-3 Conversion Request Unit

Input Channel Selection

The analog input multiplexer selects one of the available analog inputs (CH0 - CHx¹) to be converted. Three sources can select a linear sequence, an arbitrary sequence, or a specific channel. The priorities of these sources can be configured.

Additional external analog multiplexers can be controlled automatically, if more separate input channels are required than are built in (see [Section 29.10](#)).

Note: Not all analog input channels are necessarily available in all packages, due to pin limitations. Refer to the implementation description in [Section 29.12](#).

Conversion Control

Conversion parameters, such as sample phase duration or result resolution can be configured for 4 input classes (2 group-specific classes, 2 global classes). Each channel can be individually assigned to one of these input classes.

Each channel can select either the standard reference voltage or the alternate reference voltage (restrictions see [“Product-Specific Configuration” on Page 29-147](#)).

¹ The availability of input channels depends on the package of the used product type. A summary can be found in [Section 29.12.3](#).

Versatile Analog-to-Digital Converter (VADC)

The input channels can, thus, be adjusted to the type of sensor (or other analog sources) connected to the ADC.

This unit also controls the built-in multiplexer and external analog multiplexers, if selected.

Analog/Digital Converter

The selected input channel is converted to a digital value by first sampling the voltage on the selected input and then generating the selected number of result bits.

For 12-bit conversions, post-calibration is executed after converting the channel.

For broken wire detection (see [Section 29.9.1](#)), the converter network can be preloaded before sampling the selected input channel.

Result Handling

The conversion results of each analog input channel can be directed to one of 16 group-specific result registers and one global result register to be stored there. A result register can be used by a group of channels or by a single channel.

The wait-for-read mode avoids data loss due to result overwrite by blocking a conversion until the previous result has been read.

Data reduction (e.g. for digital anti-aliasing filtering) can automatically add up to 4 conversion results before issuing a service request.

Alternatively, an FIR or IIR filter can be enabled that preprocesses the conversion results before sending them to the result register.

Also, result registers can be concatenated to build FIFO structures that store a number of conversion results without overwriting previous data. This increases the allowed CPU latency for retrieving conversion data from the ADC.

Service Request Generation

Several ADC events can issue service requests to CPU or DMA:

- **Source events** indicate the completion of a conversion sequence in the corresponding request source. This event can be used to trigger the setup of a new sequence.
- **Channel events** indicate the completion of a conversion for a certain channel. This can be combined with limit checking, so interrupts are generated only if the result is within a defined range of values.
- **Result events** indicate the availability of new result data in the corresponding result register. If data reduction mode is active, events are generated only after a complete accumulation sequence.

Each event can be assigned to one of eight service request nodes. This allows grouping the requests according to the requirements of the application.

Versatile Analog-to-Digital Converter (VADC)

Safety Features

Safety-aware applications are supported with mechanisms that help to ensure the integrity of a signal path.

Broken-wire-detection (BWD) preloads the converter network with a selectable level before sampling the input channel. The result will then reflect the preload value if the input signal is no more connected. If buffer capacitors are used, a certain number of conversions may be required to reach the failure indication level.

Pull Down Diagnostics (PDD) connects an additional strong pull-down device to an input channel. A subsequent conversion can then confirm the expected modified signal level. This allows to check the proper connection of a signal source (sensor) to the multiplexer.

Multiplexer Diagnostics (MD) connects a weak pull-up or pull-down device to an input channel. A subsequent conversion can then confirm the expected modified signal level. This allows to check the proper operation of the multiplexer.

Converter Diagnostics (CD) connects an alternate signal to the converter. A subsequent conversion can then confirm the proper operation of the converter.

29.2 Electrical Models

Each conversion of an analog input voltage to a digital value consists of two consecutive phases:

- During the sample phase, the input voltage is sampled and stored.
The **Input Signal Path** is a simplified model for this.
- During the conversion phase the stored voltage is converted to a digital result.
The **Reference Voltage Path** is a simplified model for this.

Input Signal Path

The ADC of the TC26x B-Step uses a switched capacitor field represented by C_{AINSW} (small parasitic capacitances are present at each input pin). During the sample phase, the capacitor field C_{AINSW} is connected to the selected analog input CHx via the input multiplexer (modeled by ideal switches and series resistors R_{AIN}).

The switch to CHx is closed during the sample phase and connects the capacitor field to the input voltage V_{AINx} .

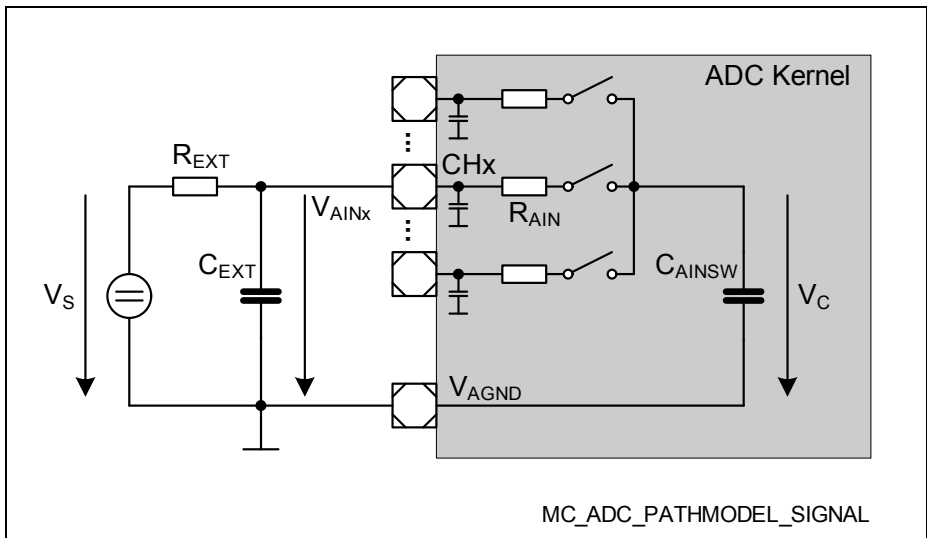


Figure 29-4 Signal Path Model

A simplified model for the analog input signal path is given in **Figure 29-4**. An analog voltage source (value V_S) with an internal impedance of R_{EXT} delivers the analog input that should be converted.

During the sample phase the corresponding switch is closed and the capacitor field C_{AINSW} is charged. Due to the low-pass behavior of the resulting RC combination, the

Versatile Analog-to-Digital Converter (VADC)

voltage V_C to be actually converted does not immediately follow V_S . The value R_{EXT} of the analog voltage source and the desired precision of the conversion strongly define the required length of the sample phase.

To reduce the influence of R_{EXT} and to filter input noise, it is recommended to introduce a fast external blocking capacitor C_{EXT} at the analog input pin of the ADC. Like this, mainly C_{EXT} delivers the charge during the sample phase. This structure allows a significantly shorter sample phase than without a blocking capacitor, because the low-pass time constant defining the sample time is mainly given by the values of R_{AIN} and C_{AINSW} .

The resulting low-pass filter of R_{EXT} (usually a parameter of the signal source) and C_{EXT} should be dimensioned according to the application's requirements:

- For quickly changing dynamic signals, a smaller capacitor allows V_{AINx} to follow V_S between two sample phases of the same analog input channel.
- For high-precision conversions, an external blocking capacitor C_{EXT} in the range of at least $2^n \times C_{AINSW}$ keeps the voltage change of V_{AINx} during the sample phase below 1 LSB_n. This voltage change is due to the charge redistribution between C_{EXT} and C_{AINSW} .

Leakage current through the analog input structure of the ADC can generate a voltage drop over R_{EXT} , introducing an error. The ADC input leakage current increases at high temperature and if the input voltage level is close to the analog supply ground V_{SS} or to the analog power supply V_{DDPA} . The input leakage current of an ADC channel can be reduced by avoiding input voltages close to the supplies.

An overload condition (input voltage exceeds the supply range) at adjacent analog inputs injects an additional leakage current (defined by a coupling factor).

The capacitor C_{AINSW} is automatically precharged to a voltage of approximately half the standard reference voltage V_{AREF} while the converter is idle. Due to varying parameters and parasitic effects, the precharge voltage of C_{AINSW} is typically smaller than $V_{AREF} / 2$.

Note: When conversions are executed in a sequence, or when a conversion cancels a running conversion, the sample phase starts immediately.

The converter does not become idle in this case and C_{AINSW} is not precharged!

Versatile Analog-to-Digital Converter (VADC)

Reference Voltage Path

During the conversion phase, parts of the capacitor field (represented by C_{AREFSW}) are switched to a reference input (V_{AREF} or CH0) or to V_{AGND} . Using CH0 as alternate reference source allows conversions of 5.0 V and 3.3 V based analog input signals with the same ADC kernel.

Stable and noise-free reference and analog supply voltages support accurate conversion results. Because noise can also be introduced from other modules (e.g. switching pins), it is strongly recommended to carefully decouple analog from digital signal domains.

The switching of parts of C_{AREFSW} requires a dynamic current at the selected reference input. The impedance R_{REFEXT} of the reference voltage source V_R has to be low enough to supply the reference current during the conversion phase. An external blocking capacitor C_{REFEXT} can supply the peak currents and minimize the current to be delivered by the reference source.

The reference current I_{AREF} introduces a voltage drop at R_{REFEXT} that should not be neglected for the calculation of the overall accuracy. The average reference current during a conversion depends on the reference voltage level and the time t_{ADC} between two conversion starts: $I_{AREF} = C_{AREFSW} \times V_{AREF} / t_{ADC}$.

The reference current can be reduced by preloading the capacitor field from the supply voltage V_{DDM} . This feature is enabled by setting bit REFPC in register GLOB_CFG. In this case the reference voltage and the supply voltage must be on the same level. The capacitor field is first preloaded from V_{DDM} in a first phase and then adjusted to V_{AREF} in a second phase.

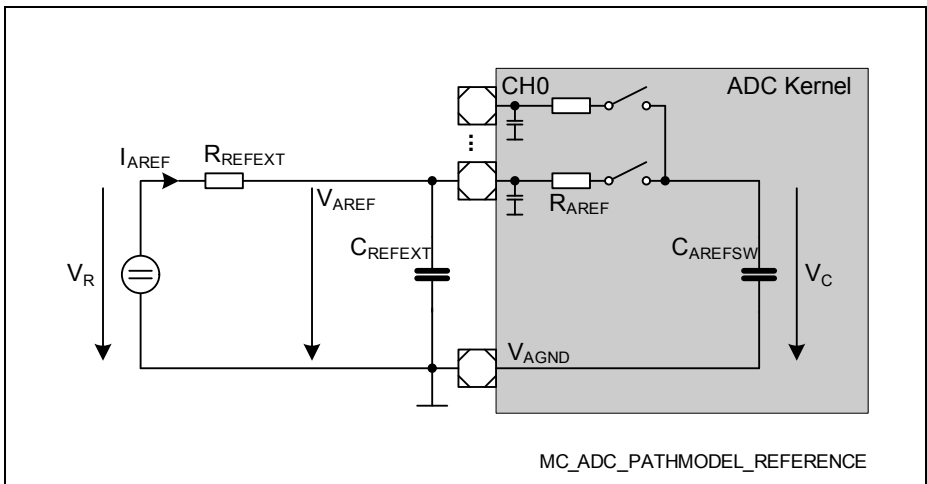


Figure 29-5 Reference Path Model

Versatile Analog-to-Digital Converter (VADC)

Reference Signal Connection

The reference voltages (V_{AREF} , V_{AGND}) have a strong influence on the digital result of a conversion. It is, therefore, recommended to carefully avoid noise of these inputs. The recommended filter structures (see figure) help to cancel high-frequency noise.

- Capacitors (C_{REFEXT}) between corresponding reference pins (V_{AREF} , V_{AGND}) provide peak currents for the conversion steps¹⁾
- Capacitors (C_N) between reference ground and signal ground (V_{AGND} , V_{SS}) and the inductance of the connections attenuate noise
- Use low-ESR capacitors connected closely to the pins
- Connect reference pins via separate lines to star points close to the sensor supply
- Additional resistors in the reference lines (R_F) must be able to carry the average reference current, to avoid inaccuracy due to voltage drop
- The reference ground lines are connected to the signal ground in many applications

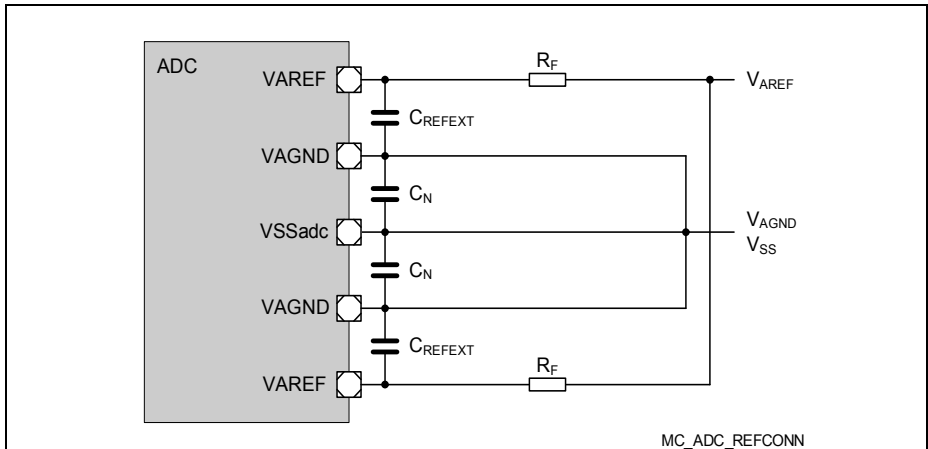


Figure 29-6 Connection of Reference Signals

Due to the charge redistribution between C_{REFEXT} and C_{AREFSW} , the voltage V_{AREF} decreases during the conversion phase. The error introduced by this effect is limited by the external blocking capacitor. $C_{REFEXT} \geq 2^n \times C_{AREFSW}$ limits this error to $1/2 \text{ LSB}_n$ (double C_{REFEXT} to limit the error to $1/4 \text{ LSB}_n$).

Assuming 12-bit conversions and $C_{AREFSW} = 30 \text{ pF}$ leads to:

$C_{REFEXT} > 2^{12} \times 30 \text{ pF} = 123 \text{ nF}$, 100 nF are a good approximation.

The noise filtering capacitor C_N depends on the board properties (e.g. 100 nF).

1) The required capacitor value here depends on the resolution and accuracy.

Versatile Analog-to-Digital Converter (VADC)

Transfer Characteristics and Error Definitions

The transfer characteristic of the ADC describes the association of analog input voltages to the 2^n discrete digital result values (n bits resolution). Each digital result value (in the range of 0 to 2^n-1) represents an input voltage range defined by the reference voltage range divided by 2^n . This range (called quantization step or code width) represents the granularity (called LSB_n) of the ADC. The discrete character of the digital result generates a system-inherent quantization uncertainty of $\pm 0.5 LSB_n$ for each conversion result.

The ideal transfer curve has the first digital transition (between 0 and 1) when the analog input reaches $0.5 LSB_n$. The quantization steps are equally distributed over the input voltage range.

Analog input voltages below or above the reference voltage limits lead to a saturation of the digital result at 0 or 2^n-1 .

The real transfer curve can exhibit certain deviations from the ideal transfer curve:

- The **offset error** is the deviation of the real transfer line from the ideal transfer line at the lowest code. This refers to best-fit lines through all possible codes, for both cases.
- The **gain error** is the deviation of the slope of the real transfer line from the slope of the ideal transfer line. This refers to best-fit lines through all possible codes, for both cases.
- The **differential non-linearity error (DNL)** is the deviation of the real code width (variation of the analog input voltage between two adjacent digital conversion results) from the ideal code width.
- The **integral non-linearity error (INL)** is the deviation of the real transfer curve from an adjusted ideal transfer curve (same offset and gain error as the real curve, but equal code widths).
- The **total unadjusted error (TUE)** describes the maximum deviation between a real conversion result and the ideal transfer characteristics over a given measurement range. Since some of these errors noted above can compensate each other, the TUE value generally is much less than the sum of the individual errors.
The TUE also covers production process variations and internal noise effects (if switching noise is generated by the system, this generally leads to an increased TUE value).

Versatile Analog-to-Digital Converter (VADC)

29.3 Configuration of General Functions

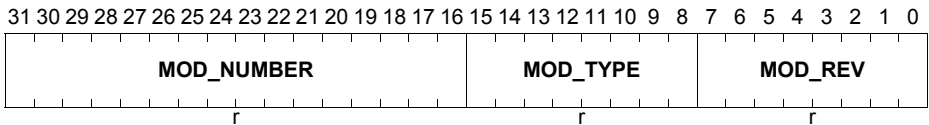
While many parameters can be selected individually for each channel, source, or group, some adjustments are valid for the whole ADC cluster.

29.3.1 Module Identification

The module identification register indicates the version of the ADC module that is used in the TC26x B-Step.

ID

Module Identification Register (0008_H) Reset Value: 00C5 C0XX_H



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Indicates the revision number of the implementation. This information depends on the design step.
MOD_TYPE	[15:8]	r	Module Type This internal marker is fixed to C0 _H .
MOD_NUMBER	[31:16]	r	Module Number Indicates the module identification number (00C5 _H = SARADC).

Versatile Analog-to-Digital Converter (VADC)

29.3.2 System Registers

A set of standardized registers provides general access to the module and controls basic system functions.

The Clock Control Register **CLC** allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. Register **CLC** controls the module clock signal and the reactivity to the sleep mode signal.

CLC
Clock Control Register

 (0000_H)

 Reset Value: 0000 0003_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	E DIS	0	DIS S	DIS R
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module. Also the analog section is disabled by clearing ANONS. 0 _B On request: enable the module clock 1 _B Off request: stop the module clock
DISS	1	r	Module Disable Status Bit 0 _B Module clock is enabled 1 _B Off: module is not clocked
0	2	r	Reserved, write 0, read as 0
EDIS	3	rw	Sleep Mode Enable Control Used to control module's reaction to sleep mode. 0 _B Sleep mode request is enabled and functional 1 _B Module disregards the sleep mode control signal
0	[31:4]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

The OCDS control and status register OCS controls the module's behavior in suspend mode (used for debugging) and includes the module-related control bits for the OCDS Trigger Bus (OTGB).

The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The OCS register can only be written when the OCDS is enabled.

If OCDS is being disabled, the OCS register value will not change.

When OCDS is disabled the OCS suspend control is ineffective.

Write access is 32 bit wide only and requires Supervisor Mode.

OCS
OCDS Control and Status Register (0028_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	SUS STA	SUS_P	SUS			0	0	0	0	0	0	0	0	0
r	r	rh	w	rw			r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	TG_P	TGB	TGS	
r	r	r	r	r	r	r	r	r	r	r	r	w	rw	rw	

Field	Bits	Type	Description
TGS	[1:0]	rw	Trigger Set for OTGB0/1 00 _B No Trigger Set output 01 _B Trigger Set 1: TS16_SSIG, input sample signals 10 _B Reserved 11 _B Reserved
TGB	2	rw	OTGB0/1 Bus Select 0 _B Trigger Set is output on OTGB0 1 _B Trigger Set is output on OTGB1
TG_P	3	w	TGS, TGB Write Protection TGS and TGB are only written when TG_P is 1, otherwise unchanged. Read as 0.
0	[23:4]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0000 _B Will not suspend 0001 _B Hard suspend: Clock is switched off immediately. 0010 _B Soft suspend mode 0: Stop conversions after the currently running one is completed and its result has been stored. No change for the arbiter. 0011 _B Soft suspend mode 1: Stop conversions after the currently running one is completed and its result has been stored. Stop arbiter after the current arbitration round. others: Reserved
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[31:30]	r	Reserved, write 0, read as 0

Table 29-2 TS16_SSIG Trigger Set VADC

Bits	Name	Description
[3:0]	GxSAMPLE	Input signal sample phase of converter group x (x = 3-0)
[15:4]	0	Reserved

Versatile Analog-to-Digital Converter (VADC)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on-chip bus master TAG IDs 00 0000_B to 01 1111_B (see On-Chip Bus chapter for the mapping of product TAG ID <-> master peripheral). Register provides one enable bit for each possible TAG ID encoding.

Register ACCEN0 itself is protected by the Safety Endinit feature.

Mapping of TAG IDs to .ENx: EN0 -> TAG ID 00 0000_B, EN1 -> TAG ID 00 0001_B, ..., EN31 -> TAG ID 01 1111_B (TAG IDs 1X XXXX_B are not used).

ACCEN0

Access Enable Register 0

(003C_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
ENx (x = 0-31)	x	rW	Access Enable for Master TAG ID x This bit enables write access to the module kernel addresses for transactions with the Master TAG ID x 0 _B No Write access 1 _B Write access will be executed

Individual Module Reset

The Kernel Reset Registers **KRST0/KRST1** are used to reset the related module kernel. To reset a module kernel it is necessary to set the RST bits in both Kernel Reset Registers. The RST bits will be cleared by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Registers 0 and 1 each include bit RST. To trigger a module reset, both RST bits must be set. They will be cleared automatically with the end of the BPI kernel reset sequence.

1) The Access Enable functionality controls only write transactions to registers CLC, OCS, KRSTx, and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

Versatile Analog-to-Digital Converter (VADC)

Kernel Reset Register 0 includes a kernel reset status bit that is set in the same clock cycle the RST bit is cleared. This bit indicates that a kernel reset was processed. Bit RSTSTAT can be cleared by setting bit CLR in register **KRSTCLR**.

Note: During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

KRST0
Kernel Reset Register 0
(0034_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	RST STA T	RST
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rh	rwh

Field	Bits	Type	Description
RST	0	rwh	Kernel Reset Request a kernel reset. The reset is executed if the reset bits of both kernel reset registers are set. 0 _B No action 1 _B A kernel reset was requested RST is cleared after the kernel reset was executed.
RSTSTAT	1	rh	Kernel Reset Status Indicates an executed kernel reset. RSTSTAT is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. 0 _B No kernel reset was executed 1 _B Kernel reset was executed Clear RSTSTAT by setting bit CLR in register KRSTCLR.
0	[31:2]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

KRST1

Kernel Reset Register 1

(0030_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	RST
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	rwh

Field	Bits	Type	Description
RST	0	rwh	Kernel Reset Request a kernel reset. The reset is executed if the reset bits of both kernel reset registers are set. 0 _B No action 1 _B A kernel reset was requested RST is cleared after the kernel reset was executed.
0	[31:1]	r	Reserved, write 0, read as 0

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (RSTSTAT).

KRSTCLR

Kernel Reset Status Clear Register (002C_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	CLR
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	w

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

29.3.3 General Clocking Scheme and Control

The A/D Converters of the TC26x B-Step are supplied with a global clock signal from the system f_{VADC} . Two clock signals are derived from this input and are distributed to all converters. The global configuration register defines common clock bases for all converters of the cluster. This ensures deterministic behavior of converters that shall operate in parallel.

The analog converter clock f_{ADCI} determines the performance of the converters and must be selected to comply with the specification given in the Data Sheet.

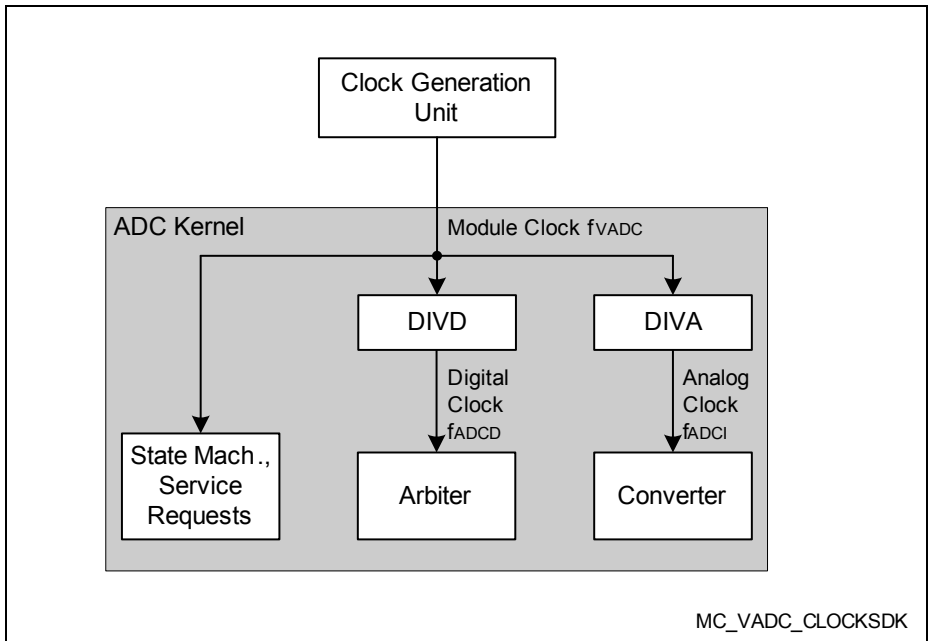


Figure 29-7 Clock Signal Summary

Versatile Analog-to-Digital Converter (VADC)

Global Configuration

The global configuration register provides global control and configuration options that are valid for all converters of the cluster.

GLOBCFG

Global Configuration Register (0080_H) Reset Value: 0000 000F_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SU CAL	0	0	0	0	0	0	0	0	0	0	0	DP CAL 3	DP CAL 2	DP CAL 1	DP CAL 0
w	r	r	r	r	r	r	r	r	r	r	r	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV WC	LO SUP	0	REF PC	0	0	DIVD	DC MSB	0	0	DIVA					
w	rw	r	rw	r	r	rw	rw	r	r	rw					

Field	Bits	Type	Description
DIVA	[4:0]	rw	Divider Factor for the Analog Internal Clock Defines the frequency of the basic converter clock f_{ADCI} (base clock for conversion and sample phase). 00 _H $f_{ADCI} = f_{VADC} / 2$ 01 _H $f_{ADCI} = f_{VADC} / 2$ 02 _H $f_{ADCI} = f_{VADC} / 3$... 1F _H $f_{ADCI} = f_{VADC} / 32$
0	[6:5]	r	Reserved, write 0, read as 0
DCMSB	7	rw	Double Clock for the MSB Conversion Selects an additional clock cycle for the conversion step of the MSB. 0 _B 1 clock cycle for the MSB (standard) 1 _B Reserved
DIVD	[9:8]	rw	Divider Factor for the Arbiter Clock Defines the frequency of the arbiter clock f_{ADCD} . 00 _B $f_{ADCD} = f_{VADC}$ 01 _B $f_{ADCD} = f_{VADC} / 2$ 10 _B $f_{ADCD} = f_{VADC} / 3$ 11 _B $f_{ADCD} = f_{VADC} / 4$
0	[11:10]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
REFPC	12	rw	Reference Precharge Control 0_B Use V_{DDM} for precharging and V_{AREF} for the final adjustment during a conversion 1_B Use V_{AREF} for the conversion
0	13	r	Reserved, write 0, read as 0
LOSUP	14	rw	Low Power Supply Voltage Select Adjusts the analog circuitry to the supply voltage used in the application system. 0_B 5 V power supply is connected 1_B 3.3 V power supply is connected <i>Note: Make sure to keep LOSUP = 0 in the case of a 5 V supply.</i>
DIVWC	15	w	Write Control for Divider Parameters 0_B No write access to divider parameters 1_B Bitfields DIVA, DCMSB, DIVD, REFPC, LOSUP can be written
DPCALx (x = 0 - 3)	x+16	rw	Disable Post-Calibration 0_B Automatic post-calibration after each conversion of group x 1_B No post-calibration <i>Note: This bit is only valid for the calibrated converters within the given product type.</i>
0	[30:20]	r	Reserved, write 0, read as 0
SUCAL	31	w	Start-Up Calibration The 0-1 transition of bit SUCAL initiates the start-up calibration phase of all calibrated analog converters. 0_B No action 1_B Initiate the start-up calibration phase (indication in bit GxARBCFG.CAL)

Versatile Analog-to-Digital Converter (VADC)
Priority Channel Assignment

Each channel of a group can be assigned to this group's request sources and is then regarded as a priority channel. An assigned priority channel can only be converted by its own group's request sources. A not assigned channel can also be converted by the background request source.

GxCHASS (x = 0 - 3)
Channel Assignment Register, Group x

$$(x * 0400_H + 0488_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS
CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS	ASS
CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH	CH
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ASSCH_y (y = 0 - 31)	y	rw	Assignment for Channel y 0 _B Channel y can be a background channel converted with lowest priority 1 _B Channel y is a priority channel within group x

Versatile Analog-to-Digital Converter (VADC)
Priority Result Register Assignment

Each result register of a group can be assigned to this group's request sources and is then regarded as a reserved resource. An assigned result register can only be written by its own group's request sources. A not assigned result register can also be written by the background request source.

GxRRASS (x = 0 - 3)
Result Assignment Register, Group x

$$(x * 0400_H + 048C_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ASS RR 15	ASS RR 14	ASS RR 13	ASS RR 12	ASS RR 11	ASS RR 10	ASS RR 9	ASS RR 8	ASS RR 7	ASS RR 6	ASS RR 5	ASS RR 4	ASS RR 3	ASS RR 2	ASS RR 1	ASS RR 0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
ASSRRy (y = 0 - 15)	y	rw	Assignment for Result Register y 0 _B Result register y can also be written by the background source 1 _B Result register y can only be written by sources within group x
0	[31:16]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)
29.3.4 Register Access Control

Several protection schemes are provided to prevent unintended write access to control bitfields of the VADC.

- The registers of the VADC are protected by the general access control mechanism that is configured by register **ACCEN0**.
- A specific register access control scheme provides a versatile protection scheme against unintended corruption of register contents. Registers ACCPROT0/1 allow the restriction of write accesses for several groups of registers. The registers to be protected can be selected by the user. **Table 29-3** lists the registers that belong to each register group.
Registers ACCPROT0/1 themselves are protected by the Safety Endinit feature.
- Groups of bitfields within a register may also be protected by an associated write control bit. This write control bit (xxWC) must be written with 1 along with the write access to the intended bitfield(s).

Valid for TC26/TC24: 4 groups

ACCPROT0
Access Protection Register
(0088_H)
Reset Value: 0000 0000_H

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AP GC	0	0	0	0	0	0	0	0	0	0	0	AP I3	AP I2	AP I1	AP I0	
	rw	r	r	r	r	r	r	r	r	r	r	rw	rw	rw	rw	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AP EM	0	0	0	0	0	0	0	0	0	0	0	AP C3	AP C2	AP C1	AP C0	
	rw	r	r	r	r	r	r	r	r	r	r	rw	rw	rw	rw	

Field	Bits	Type	Description
APCx (x = 0 - 3)	x	rw	Access Protection Channel Control, Group 0 - 3 0 _B Full access to registers 1 _B Write access to channel control registers is blocked
0	[14:4]	r	Reserved, write 0, read as 0
APEM	15	rw	Access Protection External Multiplexer 0 _B Full access to registers 1 _B Write access to external multiplexer registers is blocked

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
APIx (x = 0 - 3)	16 + x	rw	Access Protection Initialization, Group 0 - 3 0 _B Full access to registers 1 _B Write access to initialization registers is blocked
0	[30:20]	r	Reserved, write 0, read as 0
APGC	31	rw	Access Protection Global Configuration 0 _B Full access to register 1 _B Write access to global configuration register is blocked

Valid for TC26/TC24: 4 groups

ACCPROT1
Access Protection Register (008C_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	AP R3	AP R2	AP R1	AP R0
r	r	r	r	r	r	r	r	r	r	r	r	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AP TF	0	0	0	0	0	0	0	0	0	0	0	AP S3	AP S2	AP S1	AP S0
rw	r	r	r	r	r	r	r	r	r	r	r	rw	rw	rw	rw

Field	Bits	Type	Description
APsx (x = 0 - 3)	x	rw	Access Protection Service Request, Group 0 - 3 0 _B Full access to registers 1 _B Write access to service request registers is blocked
0	[14:4]	r	Reserved, write 0, read as 0
APTF	15	rw	Access Protection Test Function 0 _B Full access to register 1 _B Write access to test function register is blocked
APRx (x = 0 - 3)	16 + x	rw	Access Protection Result Registers, Group 0 - 3 0 _B Full access to registers 1 _B Write access to result registers is blocked

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
0	[31:20]	r	Reserved, write 0, read as 0

Table 29-3 Register Protection Groups

Control Bits	Registers	Notes
APCx	GxCHCTR0 ... GxCHCTRy	Channel control
APEM	EMUXSEL, GxEMUXCTR	External multiplexer control
APIx	GxARBCFG, GxARBPR, GxCHASS, GxRRASS, GxICLASS0/1, GxSYNCTR	Initialization
APGC	GLOBCFG	Global configuration
APsx	GxSEFLAG, GxSEVNP, GxCEFLAG, GxCEVNP0/1, GxREFLAG, GxREVNP0/1, GxSRACT	Service request control
APTF	GLOBTF	Test functions
APRx	GxRCR0 ... GxRCR15, GxBOUND, GxRES0 ... GxRES15	Result control

 Versatile Analog-to-Digital Converter (VADC)

29.4 Analog Module Activation and Control

The analog converter of the ADC is the functional block that generates the digital result values from the selected input voltage. It draws a permanent current during its operation and can be deactivated between conversions to reduce the consumed overall energy.

Note: After reset the analog converters are off. They must be enabled before triggering any action involving a converter.

The accuracy of the conversions is improved by calibrating the analog converter to compensate process, temperature, and voltage variations.

29.4.1 Analog Converter Control

The operating mode is determined by bitfield **GxARBCFG (x = 0 - 3).ANONS**:

- ANONS = 11_B: **Normal Operation**
The converter is active, conversions are started immediately.
Requires no wakeup time.
- ANONS = 10_B or 01_B: **Reserved**
- ANONS = 00_B: **Converter switched Off** (default after reset)
The converter is switched off. Furthermore, digital logic blocks are set to their initial state. If the arbiter is currently running, it completes the actual arbitration round and then stops.
Before starting a conversion, select the active mode for ANONS.
Requires the wakeup time (see below).

Note: Since switching off the analog part requires a wake-up phase, the optimum configuration depends on the actual application.

Entering Power Save Modes

To ensure proper operation of the converter, disable it (ANONS = 00_B) before putting the system into a power save mode. This makes sure that no conversion is disrupted by system activities.

Wakeup Time from Analog Powerdown

When the converter is activated, it needs a certain wakeup time to settle before a conversion can be properly executed. This wakeup time can be established by waiting the required period before starting a conversion, or by adding it to the intended sample time.

The wakeup time is approximately 15 μ s.
Exact numbers can be found in the respective Data Sheets.

Note: The wakeup time is also required after initially enabling the converter.

Versatile Analog-to-Digital Converter (VADC)

29.4.2 Alternate Reference Selection

The VADC control features allow selecting an alternate reference for each channel. This reference signal is taken from channel 0 of a group.

29.4.3 Calibration

Calibration automatically compensates deviations caused by process, temperature, and voltage variations. This ensures precise results throughout the operation time.

An initial start-up calibration is required once after a reset for all converters. All converters must be enabled ($ANONS = 11_B$). The start-up calibration is initiated globally by setting bit SUCAL in register GLOBCFG. By setting bit SUCAL in register CALCTR the start-up calibration can be initiated for the corresponding cluster converter. Conversions may be started after the initial calibration sequence. This is indicated by bit $CALS = 1_B$ AND bit $CAL = 0_B$.

The start-up calibration phase takes 4 352 cycles ($4\ 352 \times 10\text{ ns} = 43.5\ \mu\text{s}$).

After that, calibration cycles will compensate the effects of drifting parameters. The calibration cycles can be executed before or after (user configuration) a conversion sequence or can be disabled.

Note: The ADC error depends on the temperature. Therefore, the calibration must be repeated periodically.

29.5 Conversion Request Generation

The conversion request unit of a group autonomously handles the generation of conversion requests. Three request sources (2 group-specific sources and the background source) can generate requests for the conversion of an analog channel. The arbiter resolves concurrent requests and selects the channel to be converted next.

Upon a trigger event, the request source requests the conversion of a certain analog input channel or a sequence of channels.

- **Software triggers**
directly activate the respective request source.
- **External triggers**
synchronize the request source activation with external events, such as a trigger pulse from a timer generating a PWM signal or from a port pin.

Application software selects the trigger type and source, the channel(s) to be converted, and the request source priority. A request source can also be activated directly by software without requiring an external trigger.

The arbiter regularly scans the request sources for pending conversion requests and selects the conversion request with the highest priority. This conversion request is then forwarded to the converter to start the sampling and conversion of the requested channel.

Each request source can operate in single-shot or in continuous mode:

- **In single-shot mode,**
the programmed conversion (sequence) is requested once after being triggered. A subsequent conversion (sequence) must be triggered again.
- **In continuous mode,**
the programmed conversion (sequence) is automatically requested repeatedly after being triggered once.

For each request source of a group, external triggers are generated from one of 15 selectable trigger inputs (REQTRx[O:A]) and from one of 16 selectable gating inputs (REQGTx[P:A])¹⁾. The available trigger signals for the TC26x B-Step are listed in [Section 29.12.4](#).

Note: [Figure 29-3 “Conversion Request Unit” on Page 29-6](#) summarizes the request sources.

¹⁾ The selected gating input can be used as a trigger signal (REQTRxP).

Versatile Analog-to-Digital Converter (VADC)

Two types of request sources are available:

- **A queued source** can issue conversion requests for an arbitrary sequence of input channels. The channel numbers for this sequence can be freely programmed¹⁾. This supports application-specific conversion sequences that cannot be covered by a channel scan source. Also, multiple conversions of the same channel within a sequence are supported.

A queued source converts a series of input channels permanently or on a regular time base. For example, if programmed with medium priority, some input channels can be converted upon a specified event (e.g. synchronized to a PWM). Conversions of lower priority sources are suspended in the meantime.

Request source 0 is a group-specific 8-stage queued source.

- **A channel scan source** can issue conversion requests for a coherent sequence of input channels. This sequence begins with the highest enabled channel number and continues towards lower channel numbers. All available channels¹⁾ can be enabled for the scan sequence. Each channel is converted once per sequence.

A scan source converts a series of input channels permanently or on a regular time base. For example, if programmed with low priority, some input channels can be scanned in a background task to update information that is not time-critical.

- Request source 1 is a group-specific channel scan source.
- Request source 2 is a global channel scan source (background source).

The background source can request conversions of all channels of all groups.

1) The availability of input channels depends on the package of the used product type. A summary can be found in [Section 29.12.3](#).

The background source can only request non-priority channels, i.e. channels that are not selected in registers GxCHASS. Priority channels are reserved for the group-specific request sources 0 and 1.

29.5.1 Queued Request Source Handling

A queued request source supports short conversion sequences (up to 8) of arbitrary channels (contrary to a scan request source with a fixed conversion order for the enabled channels). The programmed sequence is stored in a queue buffer (based on a FIFO mechanism). The requested channel numbers are entered via the queue input, while queue stage 0 defines the channel to be converted next.

A conversion request is only issued to the request source arbiter if a valid entry is stored in queue stage 0.

If the arbiter aborts a conversion triggered by a queued request source due to higher priority requests, the corresponding conversion parameters are automatically saved in the backup stage. This ensures that an aborted conversion is not lost but takes part in the next arbitration round (before stage 0).

The trigger and gating unit generates trigger events from the selected external (outside the ADC) trigger and gating signals. For example, a timer unit can issue a request signal to synchronize conversions to PWM events.

Trigger events start a queued sequence and can be generated either via software or via the selected hardware triggers. The occurrence of a trigger event is indicated by bit QSRx.EV. This flag is cleared when the corresponding conversion is started or by writing to bit QMRx.CEV.

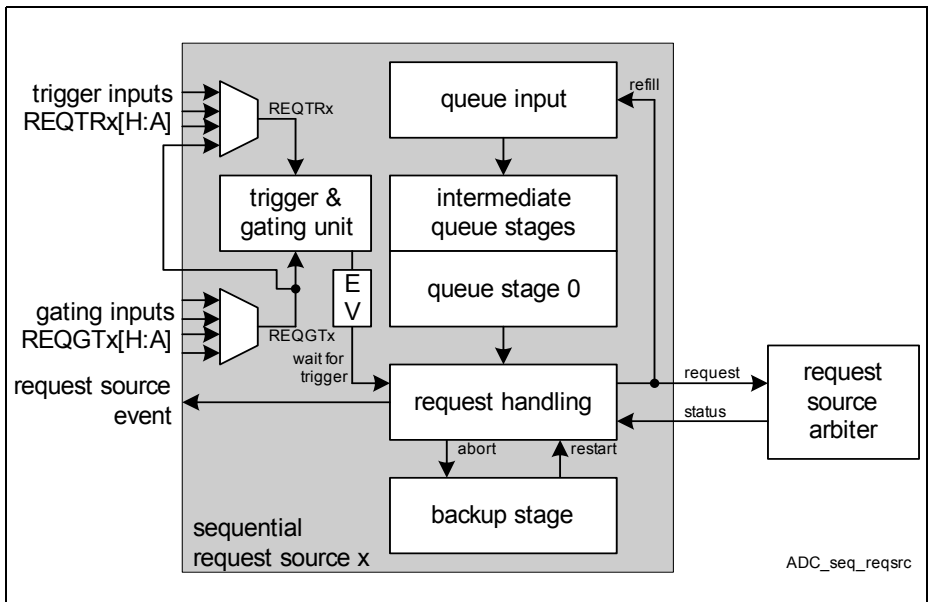


Figure 29-8 Queued Request Source

Versatile Analog-to-Digital Converter (VADC)

A sequence is defined by entering conversion requests into the queue input register (**GxQINR0 (x = 0 - 3)**). Each entry selects the channel to be converted and can enable an external trigger, generation of an interrupt, and an automatic refill (i.e. copy this entry to the top of the queue after conversion). The entries are stored in the queue buffer stages.

The content of stage 0 (**GxQ0R0 (x = 0 - 3)**) selects the channel to be converted next. When the requested conversion is started, the contents of this queue stage is invalidated and copied to the backup stage. Then the next queue entry can be handled (if available).

Note: The contents of the queue stages cannot be modified directly, but only by writing to the queue input or by flushing the queue.

*The current status of the queue is shown in register **GxQSR0 (x = 0 - 3)**.*

If all queue entries have automatic refill selected, the defined conversion sequence can be repeated without re-programming.

Properties of the Queued Request Source

Queued request source 0 provides 8 buffer stages and can handle sequences of up to 8 input channel entries. It supports short application-specific conversion sequences, especially for timing-critical sequences containing also multiple conversions of the same channel.

Queued Source Operation

Configure the queued request source by executing the following actions:

- Define the sequence by writing the entries to the queue input **GxQINR0 (x = 0 - 3)**. Initialize the complete sequence before enabling the request source, because with enabled refill feature, software writes to QINR0 are not allowed.
- If hardware trigger or gating is desired, select the appropriate trigger and gating inputs and the proper transitions by programming **GxQCTRL0 (x = 0 - 3)**. Enable the trigger and select the gating mode by programming bitfield ENGT in register **GxQMR0 (x = 0 - 3)**.¹⁾
- Enable the corresponding arbitration slot (0) to accept conversion requests from the queued source (see register **GxARBPR (x = 0 - 3)**).

Start a queued sequence by generating a trigger event:

- If a hardware trigger is selected and enabled, generate the configured transition at the selected input signal, e.g. from a timer or an input pin.
- Generate a software trigger event by setting **GxQMR0.TREV = 1**.

1) If PDOOUT signals from the ERU are used, initialize the ERU accordingly before enabling the gate inputs to avoid un expected signal transitions.

Versatile Analog-to-Digital Converter (VADC)

- Write a new entry to the queue input of an empty queue. This leads to a (new) valid queue entry that is forwarded to queue stage 0 and starts a conversion request (if enabled by GxQMR0.ENG and without waiting for an external trigger).

Note: If the refill mechanism is activated, a processed entry is automatically reloaded into the queue. This permanently repeats the respective sequence (autoscan). In this case, do not write to the queue input while the queued source is running. Write operations to a completely filled queue are ignored.

Stop or abort an ongoing queued sequence by executing one of the following actions:

- If external gating is enabled, switch the gating signal to the defined inactive level. This does not modify the queue entries, but only prevents issuing conversion requests to the arbiter.
- Disable the corresponding arbitration slot (0) in the arbiter. This does not modify the queue entries, but only prevents the arbiter from accepting requests from the request handling block.
- Disable the queued source by clearing bitfield ENG = 00_B.
 - Invalidate the next pending queue entry by setting bit GxQMR0.CLRV = 1. If the backup stage contains a valid entry, this one is invalidated, otherwise stage 0 is invalidated.
 - Remove all entries from the queue by setting bit GxQMR0.FLUSH = 1.

Queue Request Source Events and Service Requests

A request source event of a queued source occurs when a conversion is finished. A source event service request can be generated based on a request source event according to the structure shown in [Figure 29-9](#). If a request source event is detected, it sets the corresponding indication flag in register **GxSEFLAG (x = 0 - 3)**. These flags can also be set by writing a 1 to the corresponding bit position, whereas writing 0 has no effect. The indication flags can be cleared by SW by writing a 1 to the corresponding bit position in register **GxSEFCLR (x = 0 - 3)**.

The interrupt enable bit is taken from stage 0 for a normal sequential conversion, or from the backup stage for a repeated conversion after an abort.

The service request output line SR_x that is selected by the request source event interrupt node pointer bitfields in register **GxSEVNP (x = 0 - 3)** becomes activated each time the related request source event is detected (and enabled by GxQ0R0.ENS, or GxQBUR0.ENS respectively) or the related bit position in register **GxSEFLAG (x = 0 - 3)** is written with a 1 (this write action simulates a request source event).

Versatile Analog-to-Digital Converter (VADC)

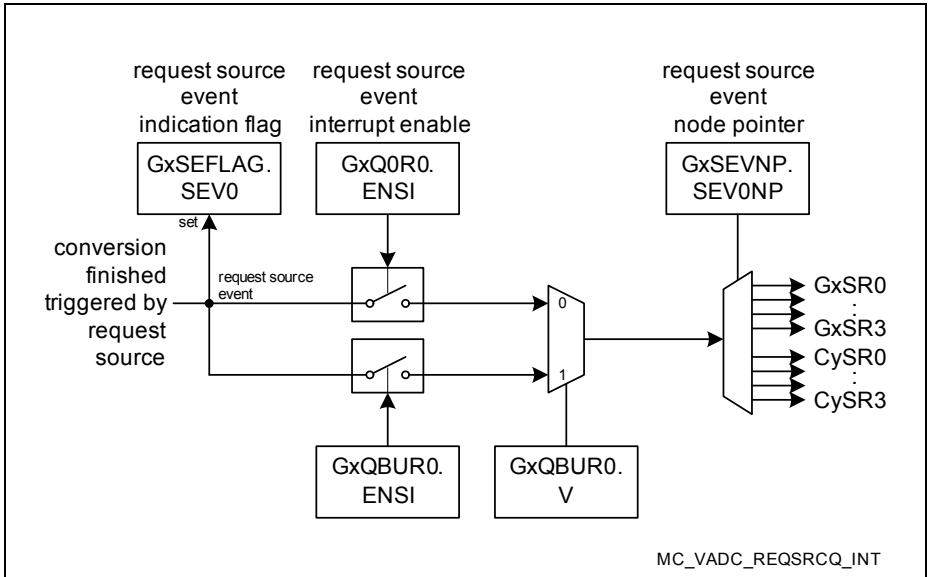


Figure 29-9 Interrupt Generation of a Queued Request Source

Versatile Analog-to-Digital Converter (VADC)

The control register of the queue source selects the external gate and/or trigger signals. Write control bits allow separate control of each function with a simple write access.

GxQCTRL0 (x = 0 - 3)
Queue 0 Source Control Register, Group x

$$(x * 0400_H + 0500_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TM WC	0	0	TM EN	0	0	0	0	GT WC	0	0	GT LVL			GT SEL	
w	r	r	rw	r	r	r	r	w	r	r	rh			rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XT WC	XT MODE		XT LVL				XT SEL	0	0	0	0		SRCRESREG		
w	rw		rh				rw	r	r	r	r		rw		

Field	Bits	Type	Description
SRCRESREG	[3:0]	rw	Source-specific Result Register 0000 _B Use GxCHCTRY.RESREG to select a group result register 0001 _B Store result in group result register GxRES1 ... 1111 _B Store result in group result register GxRES15
0	[7:4]	r	Reserved, write 0, read as 0
XTSEL	[11:8]	rw	External Trigger Input Selection The connected trigger input signals are listed in Table 29-13 “Digital Connections in the TC26x B-Step” on Page 29-157 <i>Note: XTSEL = 1111_B uses the selected gate input as trigger source (ENG_T must be 0X_B).</i>
XTLVL	12	rh	External Trigger Level Current level of the selected trigger input
XTMODE	[14:13]	rw	Trigger Operating Mode 00 _B No external trigger 01 _B Trigger event upon a falling edge 10 _B Trigger event upon a rising edge 11 _B Trigger event upon any edge

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
XTWC	15	w	Write Control for Trigger Configuration 0 _B No write access to trigger configuration 1 _B Bitfields XTMODE and XTSEL can be written
GTSEL	[19:16]	rw	Gate Input Selection The connected gate input signals are listed in Table 29-13 “Digital Connections in the TC26x B-Step” on Page 29-157
GTLVL	20	rh	Gate Input Level Current level of the selected gate input
0	[22:21]	r	Reserved, write 0, read as 0
GTWC	23	w	Write Control for Gate Configuration 0 _B No write access to gate configuration 1 _B Bitfield GTSEL can be written
0	[27:24]	r	Reserved, write 0, read as 0
TMEN	28	rw	Timer Mode Enable 0 _B No timer mode: standard gating mechanism can be used 1 _B Timer mode for equidistant sampling enabled: standard gating mechanism must be disabled
0	[30:29]	r	Reserved, write 0, read as 0
TMWC	31	w	Write Control for Timer Mode 0 _B No write access to timer mode 1 _B Bitfield TMEN can be written

Versatile Analog-to-Digital Converter (VADC)

The Queue Mode Register configures the operating mode of a queued request source.

GxQMR0 (x = 0 - 3)
Queue 0 Mode Register, Group x

$$(x * 0400_H + 0504_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	RPT DIS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	CEV	FLU SH	TR EV	CLR V	0	0	0	0	0	EN TR	ENGT	
r	r	r	r	w	w	w	w	r	r	r	r	r	rw	rw	

Field	Bits	Type	Description
ENGT	[1:0]	rw	Enable Gate Selects the gating functionality for source 0/2. 00 _B No conversion requests are issued 01 _B Conversion requests are issued if a valid conversion request is pending in the queue 0 register or in the backup register 10 _B Conversion requests are issued if a valid conversion request is pending in the queue 0 register or in the backup register and REQGT _x = 1 11 _B Conversion requests are issued if a valid conversion request is pending in the queue 0 register or in the backup register and REQGT _x = 0 <i>Note: REQGT_x is the selected gating signal.</i>
ENTR	2	rw	Enable External Trigger 0 _B External trigger disabled 1 _B The selected edge at the selected trigger input signal REQTR generates the trigger event
0	[7:3]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
CLR_V	8	w	Clear Valid Bit 0 _B No action 1 _B The next pending valid queue entry in the sequence and the event flag EV are cleared. If there is a valid entry in the queue backup register (QBUR.V = 1), this entry is cleared, otherwise the entry in queue register 0 is cleared.
TRE_V	9	w	Trigger Event 0 _B No action 1 _B Generate a trigger event by software
FLUSH	10	w	Flush Queue 0 _B No action 1 _B Clear all queue entries (including backup stage) and the event flag EV. The queue contains no more valid entry.
CE_V	11	w	Clear Event Flag 0 _B No action 1 _B Clear bit EV
0	[15:12]	r	Reserved, write 0, read as 0
RPTDIS	16	rw	Repeat Disable 0 _B A cancelled conversion is repeated 1 _B A cancelled conversion is discarded
0	[31:17]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

The Queue Status Register indicates the current status of the queued source. The filling level and the empty information refer to the queue intermediate stages (if available) and to the queue register 0. An aborted conversion stored in the backup stage is not indicated by these bits (therefore, see QBURx.V).

GxQSR0 (x = 0 - 3)
Queue 0 Status Register, Group x

$$(x * 0400_H + 0508_H)$$

Reset Value: 0000 0020_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	EV	REQ GT	0	EMP TY	0	FILL			
r	r	r	r	r	r	r	rh	rh	r	rh	r	rh			

Field	Bits	Type	Description
FILL	[3:0]	rh	Filling Level for Queue 0 Indicates the number of valid queue entries. It is incremented each time a new entry is written to QINR0 or by an enabled refill mechanism. It is decremented each time a requested conversion has been started. A new entry is ignored if the filling level has reached its maximum value. 0000 _B There is 1 (if EMPTY = 0) or no (if EMPTY = 1) valid entry in the queue 0001 _B There are 2 valid entries in the queue 0010 _B There are 3 valid entries in the queue ... 0111 _B There are 8 valid entries in the queue others: Reserved
0	4	r	Reserved, write 0, read as 0
EMPTY	5	rh	Queue Empty 0 _B There are valid entries in the queue (see FILL) 1 _B No valid entries (queue is empty)
0	6	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
REQGT	7	rh	Request Gate Level Monitors the level at the selected REQGT input. 0 _B The gate input is low 1 _B The gate input is high
EV	8	rh	Event Detected Indicates that an event has been detected while at least one valid entry has been in the queue (queue register 0 or backup stage). Once set, this bit is cleared automatically when the requested conversion is started. 0 _B No trigger event 1 _B A trigger event has been detected
0	[31:9]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

The Queue Input Register is the entry point for conversion requests of a queued request source.

GxQINR0 (x = 0 - 3)
Queue 0 Input Register, Group x

$$(x * 0400_H + 0510_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	EX TR	EN SI	RF	REQCHNR				
r	r	r	r	r	r	r	r	w	w	w	w				

Field	Bits	Type	Description
REQCHNR	[4:0]	w	Request Channel Number Defines the channel number to be converted
RF	5	w	Refill 0 _B No refill: this queue entry is converted once and then invalidated 1 _B Automatic refill: this queue entry is automatically reloaded into QINR0 when the related conversion is started
ENSI	6	w	Enable Source Interrupt 0 _B No request source interrupt 1 _B A request source event interrupt is generated upon a request source event (related conversion is finished)
EXTR	7	w	External Trigger Enables the external trigger functionality. 0 _B A valid queue entry immediately leads to a conversion request. 1 _B A valid queue entry waits for a trigger event to occur before issuing a conversion request.
0	[31:8]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

*Note: Register QINR0 shares addresses with register QBUR0.
Write operations target the control bits in register QINR0. Read operations return the status bits from register QBUR0.*

Versatile Analog-to-Digital Converter (VADC)

The queue register 0 monitors the status of the pending request (queue stage 0).

GxQ0R0 (x = 0 - 3)

Queue 0 Register 0, Group x (x * 0400_H + 050C_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	V	EX TR	EN SI	RF	REQCHNR				
r	r	r	r	r	r	r	rh	rh	rh	rh	rh				

Field	Bits	Type	Description
REQCHNR	[4:0]	rh	Request Channel Number Indicates the channel number to be converted.
RF	5	rh	Refill Indicates the handling of handled requests. 0 _B The request is discarded after the conversion start. 1 _B The request is automatically refilled into the queue after the conversion start.
ENSI	6	rh	Enable Source Interrupt 0 _B No request source interrupt 1 _B A request source event interrupt is generated upon a request source event (related conversion is finished)
EXTR	7	rh	External Trigger Enables external trigger events. 0 _B A valid queue entry immediately leads to a conversion request 1 _B The request handler waits for a trigger event
V	8	rh	Request Channel Number Valid Indicates a valid queue entry in queue register 0. 0 _B No valid queue entry 1 _B The queue entry is valid and leads to a conversion request

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
0	[31:9]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

The Queue Backup Registers monitor the status of an aborted queued request.

GxQBUR0 (x = 0 - 3)
Queue 0 Backup Register, Group x

$$(x * 0400_H + 0510_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	V	EXTR	ENSI	RF	REQCHNR				
r	r	r	r	r	r	r	rh	rh	rh	rh	rh				

Field	Bits	Type	Description
REQCHNR	[4:0]	rh	Request Channel Number The channel number of the aborted conversion that has been requested by this request source
RF	5	rh	Refill The refill control bit of the aborted conversion
ENSI	6	rh	Enable Source Interrupt The enable source interrupt control bit of the aborted conversion
EXTR	7	rh	External Trigger The external trigger control bit of the aborted conversion
V	8	rh	Request Channel Number Valid Indicates if the entry (REQCHNR, RF, TR, ENSI) in the queue backup register is valid. Bit V is set when a running conversion (that has been requested by this request source) is aborted, it is cleared when the aborted conversion is restarted. 0 _B Backup register not valid 1 _B Backup register contains a valid entry. This will be requested before a valid entry in queue register 0 (stage 0) will be requested.
0	[31:9]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

*Note: Register QBUR0 shares addresses with register QINR0.
Read operations return the status bits from register QBUR0. Write operations target the control bits in register QINR0.*

29.5.2 Channel Scan Request Source Handling

The VADC provides two types of channel scan sources:

- Source 1: Group scan source
This scan source can request all channels of the corresponding group.
- Source 2: Background scan source
This scan source can request all channels of all groups.
Priority channels selected in registers **GxCHASS (x = 0 - 3)** cannot take part in background conversion sequences.

Both sources operate in the same way and provide the same register interface. The background source provides more request/pending bits because it can request all channels of all groups.

Each analog input channel can be included in or excluded from the scan sequence by setting or clearing the corresponding channel select bit in register **GxASSEL (x = 0 - 3)** or **BRSELx (x = 0 - 3)**. The programmed register value remains unchanged by an ongoing scan sequence. The scan sequence starts with the highest enabled channel number and continues towards lower channel numbers.

Upon a load event, the request pattern is transferred to the pending bits in register **GxASPND (x = 0 - 3)** or **BRSPNDx (x = 0 - 3)**. The pending conversion requests indicate which input channels are to be converted in an ongoing scan sequence. Each conversion start that was triggered by the scan request source, automatically clears the corresponding pending bit. If the last conversion triggered by the scan source is finished and all pending bits are cleared, the current scan sequence is considered finished and a request source event is generated.

A conversion request is only issued to the request source arbiter if at least one pending bit is set.

If the arbiter aborts a conversion triggered by the scan request source due to higher priority requests, the corresponding pending bit is automatically set. This ensures that an aborted conversion is not lost but takes part in the next arbitration round.

The trigger and gating unit generates load events from the selected external (outside the ADC) trigger and gating signals. For example, a timer unit can issue a request signal to synchronize conversions to PWM events.

Load events start a scan sequence and can be generated either via software or via the selected hardware triggers. The request source event can also generate an automatic load event, so the programmed sequence is automatically repeated.

Versatile Analog-to-Digital Converter (VADC)

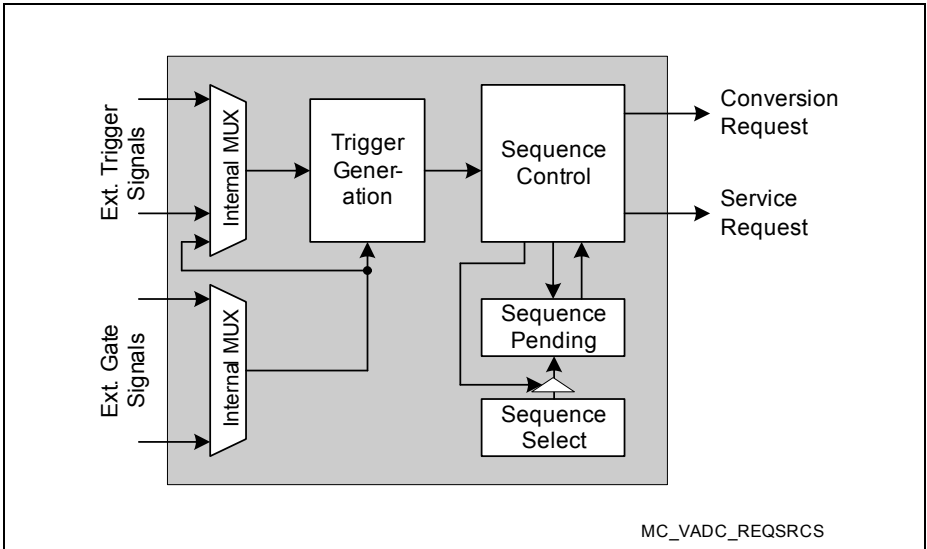


Figure 29-10 Scan Request Source

Versatile Analog-to-Digital Converter (VADC)
Scan Source Operation

Configure the scan request source by executing the following actions:

- Select the input channels for the sequence by programming **GxASSEL (x = 0 - 3)** or **BRSELx (x = 0 - 3)**
- If hardware trigger or gating is desired, select the appropriate trigger and gating inputs and the proper signal transitions by programming **GxASCTRL (x = 0 - 3)** or **BRCTRL**. Enable the trigger and select the gating mode by programming **GxASMR (x = 0 - 3)** or **BRSMR**.¹⁾
- Define the load event operation (handling of pending bits, autoscan mode) by programming **GxASMR (x = 0 - 3)** or **BRSMR**.
 A load event with bit LDM = 0 copies the content of **GxASSEL (x = 0 - 3)** or **BRSELx (x = 0 - 3)** to **GxASPND (x = 0 - 3)** or **BRSPNDx (x = 0 - 3)** (overwrite mode). This starts a new scan sequence and aborts any pending conversions from a previous scan sequence.
 A load event with bit LDM = 1 OR-combines the content of **GxASSEL (x = 0 - 3)** or **BRSELx (x = 0 - 3)** to **GxASPND (x = 0 - 3)** or **BRSPNDx (x = 0 - 3)** (combine mode). This starts a scan sequence that includes pending conversions from a previous scan sequence.
- Enable the corresponding arbitration slot (1) to accept conversion requests from the channel scan source (see register **GxARBPR (x = 0 - 3)**).

Start a channel scan sequence by generating a load event:

- If a hardware trigger is selected and enabled, generate the configured transition at the selected input signal, e.g. from a timer or an input pin.
- Generate a software load event by setting LDEV = 1 (**GxASMR (x = 0 - 3)** or **BRSMR**).
- Generate a load event by writing the scan pattern directly to the pending bits in **GxASPND (x = 0 - 3)** or **BRSPNDx (x = 0 - 3)**. The pattern is copied to **GxASSEL (x = 0 - 3)** or **BRSELx (x = 0 - 3)** and a load event is generated automatically.
 In this case, a scan sequence can be defined and started with a single data write action, e.g. under PEC control (provided that the pattern fits into one register).

Note: If autoscan is enabled, a load event is generated automatically each time a request source event occurs when the scan sequence has finished. This permanently repeats the defined scan sequence (autoscan).

Stop or abort an ongoing scan sequence by executing the following actions:

- If external gating is enabled, switch the gating signal to the defined inactive level. This does not modify the conversion pending bits, but only prevents issuing conversion requests to the arbiter.

1) If PDOUT signals from the ERU are used, initialize the ERU accordingly before enabling the gate inputs to avoid un expected signal transitions.

Versatile Analog-to-Digital Converter (VADC)

- Disable the corresponding arbitration slot (1 or 2) in the arbiter. This does not modify the contents of the conversion pending bits, but only prevents the arbiter from accepting requests from the request handling block.
- Disable the channel scan source by clearing bitfield ENGT = 00_B. Clear the pending request bits by setting bit CLRPND = 1 (**GxASMR (x = 0 - 3)** or **BRSMR**).

Scan Request Source Events and Service Requests

A request source event of a scan source occurs if the last conversion of a scan sequence is finished (all pending bits = 0). A request source event interrupt can be generated based on a request source event. If a request source event is detected, it sets the corresponding indication flag in register **GxSEFLAG (x = 0 - 3)**. These flags can also be set by writing a 1 to the corresponding bit position, whereas writing 0 has no effect.

The service request output SR_x that is selected by the request source event interrupt node pointer bitfields in register **GxSEVNP (x = 0 - 3)** becomes activated each time the related request source event is detected (and enabled by ENSI) or the related bit position in register **GxSEFLAG (x = 0 - 3)** is written with a 1 (this write action simulates a request source event).

The indication flags can be cleared by SW by writing a 1 to the corresponding bit position in register **GxSEFCLR (x = 0 - 3)**.¹⁾

1) Refer to **“Service Request Generation”** on Page 29-133.

Versatile Analog-to-Digital Converter (VADC)
Registers of Group Scan Source

There is a separate register set for each group scan source. These sources can be operated independently.

The control register of the autoscan source selects the external gate and/or trigger signals.

Write control bits allow separate control of each function with a simple write access.

GxASCTRL (x = 0 - 3)
Autoscan Source Control Register, Group x

(x * 0400_H + 0520_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TM WC	0	0	TM EN	0	0	0	0	GT WC	0	0	GT LVL			GT SEL	
w	r	r	rw	r	r	r	r	w	r	r	rh			rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XT WC	XT MODE	XT LVL			XT SEL			0	0	0	0	SRCRESREG			
w	rw	rh			rw			r	r	r	r				rw

Field	Bits	Type	Description
SRCRESREG	[3:0]	rw	Source-specific Result Register 0000 _B Use GxCHCTRY.RESREG to select a group result register 0001 _B Store result in group result register GxRES1 ... 1111 _B Store result in group result register GxRES15
0	[7:4]	r	Reserved, write 0, read as 0
XTSEL	[11:8]	rw	External Trigger Input Selection The connected trigger input signals are listed in Table 29-13 “Digital Connections in the TC26x B-Step” on Page 29-157 <i>Note: XTSEL = 1111_B uses the selected gate input as trigger source (ENGT must be 0X_B).</i>
XTLVL	12	rh	External Trigger Level Current level of the selected trigger input

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
XTMODE	[14:13]	rw	Trigger Operating Mode 00 _B No external trigger 01 _B Trigger event upon a falling edge 10 _B Trigger event upon a rising edge 11 _B Trigger event upon any edge
XTWC	15	w	Write Control for Trigger Configuration 0 _B No write access to trigger configuration 1 _B Bitfields XTMODE and XTSEL can be written
GTSEL	[19:16]	rw	Gate Input Selection The connected gate input signals are listed in Table 29-13 “Digital Connections in the TC26x B-Step” on Page 29-157
GTLVL	20	rh	Gate Input Level Current level of the selected gate input
0	[22:21]	r	Reserved, write 0, read as 0
GTWC	23	w	Write Control for Gate Configuration 0 _B No write access to gate configuration 1 _B Bitfield GTSEL can be written
0	[27:24]	r	Reserved, write 0, read as 0
TMEN	28	rw	Timer Mode Enable 0 _B No timer mode: standard gating mechanism can be used 1 _B Timer mode for equidistant sampling enabled: standard gating mechanism must be disabled
0	[30:29]	r	Reserved, write 0, read as 0
TMWC	31	w	Write Control for Timer Mode 0 _B No write access to timer mode 1 _B Bitfield TMEN can be written

Versatile Analog-to-Digital Converter (VADC)

The Conversion Request Mode Register configures the operating mode of the channel scan request source.

GxASMR (x = 0 - 3)
Autoscan Source Mode Register, Group x
(x * 0400_H + 0524_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	RPT DIS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	LD EV	CLR PND	REQ GT	0	LDM	SCAN	EN SI	EN TR	ENGT	
r	r	r	r	r	r	w	w	rh	r	rw	rw	rw	rw	rw	

Field	Bits	Type	Description
ENGT	[1:0]	rw	Enable Gate Selects the gating functionality for source 1. 00 _B No conversion requests are issued 01 _B Conversion requests are issued if at least one pending bit is set 10 _B Conversion requests are issued if at least one pending bit is set and REQGTx = 1. 11 _B Conversion requests are issued if at least one pending bit is set and REQGTx = 0. <i>Note: REQGTx is the selected gating signal.</i>
ENTR	2	rw	Enable External Trigger 0 _B External trigger disabled 1 _B The selected edge at the selected trigger input signal REQTR generates the load event
ENSI	3	rw	Enable Source Interrupt 0 _B No request source interrupt 1 _B A request source interrupt is generated upon a request source event (last pending conversion is finished)

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
SCAN	4	rw	Autoscan Enable 0_B No autoscan 1_B Autoscan functionality enabled: a request source event automatically generates a load event
LDM	5	rw	Autoscan Source Load Event Mode 0_B Overwrite mode: Copy all bits from the select registers to the pending registers upon a load event 1_B Combine mode: Set all pending bits that are set in the select registers upon a load event (logic OR)
0	6	r	Reserved, write 0, read as 0
REQGT	7	rh	Request Gate Level Monitors the level at the selected REQGT input. 0_B The gate input is low 1_B The gate input is high
CLRPND	8	w	Clear Pending Bits 0_B No action 1_B The bits in register GxASPNDx are cleared
LDEV	9	w	Generate Load Event 0_B No action 1_B A load event is generated
0	[15:10]	r	Reserved, write 0, read as 0
RPTDIS	16	rw	Repeat Disable 0_B A cancelled conversion is repeated 1_B A cancelled conversion is discarded
0	[31:17]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

The Channel Select Register selects the channels to be converted by the group scan request source. Its bits are used to update the pending register, when a load event occurs.

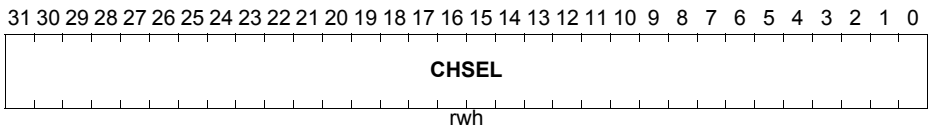
The number of valid channel bits depends on the channels available in the respective product type (refer to **“Product-Specific Configuration” on Page 29-147**).

GxASSEL (x = 0 - 3)

Autoscan Source Channel Select Register, Group x

$$(x * 0400_H + 0528_H)$$

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CHSEL	[31:0]	rwh	<p>Channel Selection</p> <p>Each bit (when set) enables the corresponding input channel of the respective group to take part in the scan sequence.</p> <p>0_B Ignore this channel</p> <p>1_B This channel is part of the scan sequence</p>

Note: Register GxASSEL is also updated when writing a pattern to register GxASPND.

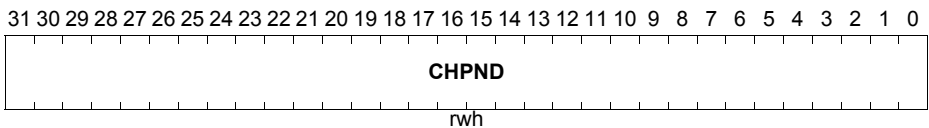
The Channel Pending Register indicates the channels to be converted in the current conversion sequence. They are updated from the select register, when a load event occurs.

GxASPND (x = 0 - 3)

Autoscan Source Pending Register, Group x

$$(x * 0400_H + 052C_H)$$

Reset Value: 0000 0000_H



Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
CHPND	[31:0]	rwh	Channels Pending Each bit (when set) request the conversion of the corresponding input channel of the respective group. 0 _B Ignore this channel 1 _B Request conversion of this channel

Note: Writing to register GxASPND automatically updates register GxASSEL.

Versatile Analog-to-Digital Converter (VADC)
Registers of Background Scan Source

There is a single register set for the background scan source. This source is common for the complete VADC.

The control register of the background request source selects the external gate and/or trigger signals.

Write control bits allow separate control of each function with a simple write access.

BRCTRL
Background Request Source Control Register
(0200_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	GT WC	0	0	GT LVL			GT SEL	
r	r	r	r	r	r	r	r	w	r	r	rh			rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XT WC	XT MODE	XT LVL			XT SEL			0	0	0	0			SRCRESREG	
w	rw	rh			rw			r	r	r	r				rw

Field	Bits	Type	Description
SRCRESREG	[3:0]	rw	Source-specific Result Register 0000 _B Use GxCHCTRY.RESREG to select a group result register 0001 _B Store result in group result register GxRES1 ... 1111 _B Store result in group result register GxRES15
0	[7:4]	r	Reserved, write 0, read as 0
XTSEL	[11:8]	rw	External Trigger Input Selection The connected trigger input signals are listed in Table 29-13 “Digital Connections in the TC26x B-Step” on Page 29-157 <i>Note: XTSEL = 1111_B uses the selected gate input as trigger source (ENGT must be 0X_B).</i>
XTLVL	12	rh	External Trigger Level Current level of the selected trigger input

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
XTMODE	[14:13]	rw	Trigger Operating Mode 00 _B No external trigger 01 _B Trigger event upon a falling edge 10 _B Trigger event upon a rising edge 11 _B Trigger event upon any edge
XTWC	15	w	Write Control for Trigger Configuration 0 _B No write access to trigger configuration 1 _B Bitfields XTMODE and XTSEL can be written
GTSEL	[19:16]	rw	Gate Input Selection The connected gate input signals are listed in Table 29-13 “Digital Connections in the TC26x B-Step” on Page 29-157
GTLVL	20	rh	Gate Input Level Current level of the selected gate input
0	[22:21]	r	Reserved, write 0, read as 0
GTWC	23	w	Write Control for Gate Configuration 0 _B No write access to gate configuration 1 _B Bitfield GTSEL can be written
0	[31:24]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

The Conversion Request Mode Register configures the operating mode of the background request source.

BRSMR
**Background Request Source Mode Register
(0204_H)**
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	RPT DIS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	LD EV	CLR PND	REQ GT	0	LDM	SCAN	EN SI	EN TR	ENGT	
r	r	r	r	r	r	w	w	rh	r	rw	rw	rw	rw	rw	

Field	Bits	Type	Description
ENGT	[1:0]	rw	Enable Gate Selects the gating functionality for source 1. 00 _B No conversion requests are issued 01 _B Conversion requests are issued if at least one pending bit is set 10 _B Conversion requests are issued if at least one pending bit is set and REQGTx = 1. 11 _B Conversion requests are issued if at least one pending bit is set and REQGTx = 0. <i>Note: REQGTx is the selected gating signal.</i>
ENTR	2	rw	Enable External Trigger 0 _B External trigger disabled 1 _B The selected edge at the selected trigger input signal REQTR generates the load event
ENSI	3	rw	Enable Source Interrupt 0 _B No request source interrupt 1 _B A request source interrupt is generated upon a request source event (last pending conversion is finished)

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
SCAN	4	rw	Autoscan Enable 0_B No autoscan 1_B Autoscan functionality enabled: a request source event automatically generates a load event
LDM	5	rw	Autoscan Source Load Event Mode 0_B Overwrite mode: Copy all bits from the select registers to the pending registers upon a load event 1_B Combine mode: Set all pending bits that are set in the select registers upon a load event (logic OR)
0	6	r	Reserved, write 0, read as 0
REQGT	7	rh	Request Gate Level Monitors the level at the selected REQGT input. 0_B The gate input is low 1_B The gate input is high
CLRPND	8	w	Clear Pending Bits 0_B No action 1_B The bits in registers BRSPNDx are cleared
LDEV	9	w	Generate Load Event 0_B No action 1_B A load event is generated
0	[15:10]	r	Reserved, write 0, read as 0
RPTDIS	16	rw	Repeat Disable 0_B A cancelled conversion is repeated 1_B A cancelled conversion is discarded
0	[31:17]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

The Channel Select Registers select the channels to be converted by the background request source (channel scan source). Its bits are used to update the pending registers, when a load event occurs.

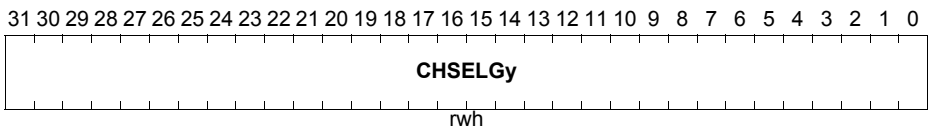
The number of valid channel bits depends on the channels available in the respective product type (refer to **“Product-Specific Configuration” on Page 29-147**).

*Note: Priority channels selected in registers **GxCHASS (x = 0 - 3)** will not be converted.*

BRSELx (x = 0 - 3)

Background Request Source Channel Select Register, Group x

(0180_H + x * 0004_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
CHSELGy	[31:0]	rwh	<p>Channel Selection Group x Each bit (when set) enables the corresponding input channel of the respective group to take part in the background scan sequence.</p> <p>0_B Ignore this channel 1_B This channel is part of the scan sequence</p>

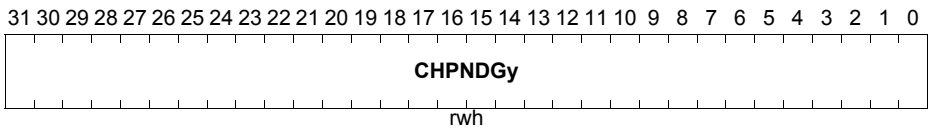
Versatile Analog-to-Digital Converter (VADC)

The Channel Pending Registers indicate the channels to be converted in the current conversion sequence. They are updated from the select registers, when a load event occurs.

BRSPNDx (x = 0 - 3)

Background Request Source Pending Register, Group x

(01C0_H + x * 0004_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
CHPNDGy	[31:0]	rwh	Channels Pending Group x Each bit (when set) request the conversion of the corresponding input channel of the respective group. 0 _B Ignore this channel 1 _B Request conversion of this channel

Note: Writing to any of registers BRSPNDx automatically updates the corresponding register BRSELx and generates a load event that copies all bits from all registers BRSELx to BRSPNDx.

Use this shortcut only when writing the last word of the request pattern.

Versatile Analog-to-Digital Converter (VADC)

29.5.3 Request Source Arbitration

The request source arbiter regularly polls the request sources, one after the other, for pending conversion requests. Each request source is assigned to a certain time slot within an arbitration round, called arbitration slot. The duration of an arbitration slot is user-configurable via register GLOB_CFG, see [Global Configuration](#).

The priority of each request source is user-configurable via register GxARBPR (x = 0 - 3), so the arbiter can select the next channel to be converted, in the case of concurrent requests from multiple sources, according to the application requirements.

An unused arbitration slot is considered empty and does not take part in the arbitration. After reset, all slots are disabled and must be enabled (register GxARBPR (x = 0 - 3)) to take part in the arbitration process.

Figure 29-11 summarizes the arbitration sequence. An arbitration round consists of one arbitration slot for each available request source. The synchronization source is always evaluated in the last slot and has a higher priority than all other sources. At the end of each arbitration round, the arbiter has determined the highest priority conversion request.

If a conversion is started in an arbitration round, this arbitration round does not deliver an arbitration winner. In the TC26x B-Step, the following request sources are available:

- Arbitration slot 0: **Group Queued source**, 8-stage sequences in arbitrary order
- Arbitration slot 1: **Group Scan source**, sequences in defined order within group
- Arbitration slot 2: **Background Scan source**, sequences in defined order, all groups
- Last arbitration slot: **Synchronization source**, synchronized conversion requests from another ADC kernel (always handled with the highest priority in a synchronization slave kernel).

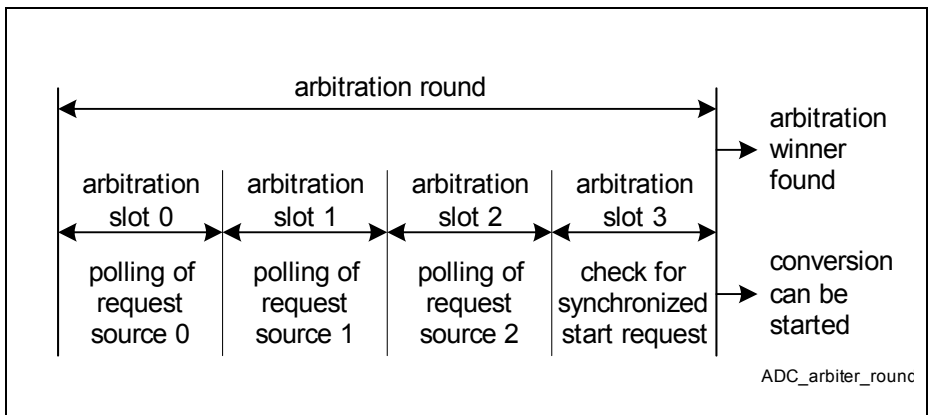


Figure 29-11 Arbitration Round with 4 Arbitration Slots

 Versatile Analog-to-Digital Converter (VADC)

29.5.3.1 Arbiter Operation and Configuration

The timing of the arbiter (i.e. of an arbitration round) is determined by the number of arbitration slots within an arbitration round and by the duration of an arbitration slot.

An arbitration round consist of 4...20 arbitration slots (defined by bitfield **GxARBCFG (x = 0 - 3).ARBRND**). 4 slots are sufficient for the TC26x B-Step, more can be programmed to obtain the same arbiter timing for different products.

The duration of an arbitration slot is configurable $t_{\text{Slot}} = (\text{DIVD}+1) / f_{\text{VADC}}$.

The duration of an arbitration round, therefore, is $t_{\text{ARB}} = 4 \times t_{\text{Slot}}$.

The period of the arbitration round introduces a timing granularity to detect an incoming conversion request signal and the earliest point to start the related conversion. This granularity can introduce a jitter of maximum one arbitration round. The jitter can be reduced by minimizing the period of an arbitration round.

To achieve a reproducible reaction time (constant delay without jitter) between the trigger event of a conversion request (e.g. by a timer unit or due to an external event) and the start of the related conversion, mainly the following two options exist. For both options, the converter has to be idle and other conversion requests must not be pending for at least one arbiter round before the trigger event occurs:

- If bit **GxARBCFG (x = 0 - 3).ARBM = 0**, the **arbiter runs permanently**. In this mode, synchronized conversions of more than one ADC kernel are possible.¹⁾
The trigger for a conversion request has to be generated synchronously to the arbiter timing. Incoming triggers should have exactly n-times the granularity of the arbiter ($n = 1, 2, 3, \dots$). In order to allow some flexibility, the duration of an arbitration slot can be programmed in cycles of f_{VADC} .
- If bit **GxARBCFG (x = 0 - 3).ARBM = 1**, the **arbiter stops after an arbitration round** when no conversion request have been found pending any more. The arbiter is started again if at least one enabled request source indicates a pending conversion request. The trigger for a conversion request does not need to be synchronous to the arbiter timing.

In this mode, parallel conversions are not possible for synchronization slave kernels.

Each request source has a configurable priority, so the arbiter can resolve concurrent conversion requests from different sources. The request with the highest priority is selected for conversion. These priorities can be adapted to the requirements of a given application (see register **GxARBPR (x = 0 - 3)**).

The **Conversion Start Mode** determines the handling of the conversion request that has won the arbitration.

1) For more information, refer to **"Synchronization of Conversions" on Page 29-117**.

Versatile Analog-to-Digital Converter (VADC)

The Arbitration Configuration Register selects the timing and the behavior of the arbiter.

GxARBCFG (x = 0 - 3)
Arbitration Configuration Register, Group x
(x * 0400_H + 0480_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SAM PLE	BU SY	CAL S	CAL	0	0	SYN RUN	CHNR				CSRC			ANONS	
rh	rh	rh	rh	r	r	rh	rh				rh			rh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	ARB M	0	ARBRND		0	0	ANONC	
r	r	r	r	r	r	r	r	rw	r	rw		r	r	rw	

Field	Bits	Type	Description
ANONC	[1:0]	rw	Analog Converter Control Defines the value of bitfield ANONS in a stand-alone converter or a converter in master mode. Coding see ANONS or Section 29.4.1 .
0	[3:2]	r	Reserved, write 0, read as 0
ARBRND	[5:4]	rw	Arbitration Round Length Defines the number of arbitration slots per arb. round (arbitration round length = t_{ARB}). ¹⁾ 00 _B 4 arbitration slots per round ($t_{ARB} = 4 / f_{ADCD}$) 01 _B 8 arbitration slots per round ($t_{ARB} = 8 / f_{ADCD}$) 10 _B 16 arbitration slots per round ($t_{ARB} = 16 / f_{ADCD}$) 11 _B 20 arbitration slots per round ($t_{ARB} = 20 / f_{ADCD}$)
0	6	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
ARBM	7	rw	Arbitration Mode 0_B The arbiter runs permanently. This setting is required for a synchronization slave (see Section 29.8.1) and for equidistant sampling using the signal ARBCNT (see Section 29.8.2). 1_B The arbiter only runs if at least one conversion request of an enabled request source is pending. This setting ensures a reproducible latency from an incoming request to the conversion start, if the converter is idle. Synchronized conversions are not supported.
0	[15:8]	r	Reserved, write 0, read as 0
ANONS	[17:16]	rh	Analog Converter Control Status Defined by bitfield ANONC in a stand-alone kernel or a kernel in master mode. In slave mode, this bitfield is defined by bitfield ANONC of the respective master kernel. See also Section 29.4.1 . 00_B Analog converter off 01_B Reserved 10_B Reserved 11_B Normal operation (permanently on)
CSRC	[19:18]	rh	Currently Converted Request Source Indicates the arbitration slot number of the current (BUSY = 1) or of the last (BUSY = 0) conversion. This bitfield is updated when a conversion is started. 00_B Current/last conversion for request source 0 01_B Current/last conversion for request source 1 10_B Current/last conversion for background source 11_B Current/last conversion for synchronization request (slave converter)
CHNR	[24:20]	rh	Channel Number Indicates the current or last converted analog input channel. This bitfield is updated when a conversion is started.

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
SYNRUN	25	rh	Synchronous Conversion Running Indicates that a synchronized (= parallel) conversion is currently running. 0_B Normal conversion or no conversion running 1_B A synchronized conversion is running (cannot be cancelled by higher priority requests!)
0	[27:26]	r	Reserved, write 0, read as 0
CAL	28	rh	Start-Up Calibration Active Indication Indicates the start-up calibration phase of the corresponding analog converter. 0_B Completed or not yet started 1_B Start-up calibration phase is active (set one clock cycle after setting bit SUCAL) <i>Note: Start conversions only after the start-up calibration phase is complete.²⁾</i>
CALS	29	rh	Start-Up Calibration Started Indicates that the start-up calibration has begun. 0_B Requested but not yet started 1_B Start-up calibration has begun <i>Note: Bit CALS is cleared when setting bit SUCAL and is set together with bit CAL.</i>
BUSY	30	rh	Converter Busy Flag 0_B Not busy 1_B Converter is busy with a conversion
SAMPLE	31	rh	Sample Phase Flag 0_B Converting or idle 1_B Input signal is currently sampled

- 1) The default setting of 4 arbitration slots is sufficient for correct arbitration. The duration of an arbitration round can be increased if required to synchronize requests.
- 2) The completion of the start-up calibration is indicated by CAL = 0 AND CALS = 1.
 CAL = CALS = 0 indicates that the start-up calibration was requested but has not yet begun.
 CAL = CALS = 1 indicates an ongoing calibration phase.

The Arbitration Priority Register defines the request source priority and the conversion start mode for each request source.

Note: Only change priority and conversion start mode settings of a request source while this request source is disabled, and a currently running conversion requested by this source is finished.

Versatile Analog-to-Digital Converter (VADC)

GxARBPR (x = 0 - 3)
Arbitration Priority Register, Group x

$$(x * 0400_H + 0484_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	AS EN2	AS EN1	AS EN0	0	0	0	0	0	0	0	0
r	r	r	r	r	rw	rw	rw	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	CSM 2	0	PRIO 2	CSM 1	0	PRIO 1	CSM 0	0	PRIO 0	0	PRIO 0	0
r	r	r	r	rw	r	rw	rw	r	rw	rw	r	rw	r	rw	r

Field	Bits	Type	Description
PRIO0, PRIO1, PRIO2	[1:0], [5:4], [9:8]	rw	Priority of Request Source x Arbitration priority of request source x (in slot x) 00 _B Lowest priority is selected. ... 11 _B Highest priority is selected.
CSM0, CSM1, CSM2	3, 7, 11	rw	Conversion Start Mode of Request Source x 0 _B Wait-for-start mode 1 _B Cancel-inject-repeat mode, i.e. this source can cancel conversion of other sources.
0	2, 6, 10, [23:12]	r	Reserved, write 0, read as 0
ASENy (y = 0 - 2)	24 + y	rw	Arbitration Slot y Enable Enables the associated arbitration slot of an arbiter round. The request source bits are not modified by write actions to ASENr. 0 _B The corresponding arbitration slot is disabled and considered as empty. Pending conversion requests from the associated request source are disregarded. 1 _B The corresponding arbitration slot is enabled. Pending conversion requests from the associated request source are arbitrated.
0	[31:27]	r	Reserved, write 0, read as 0

 Versatile Analog-to-Digital Converter (VADC)

29.5.3.2 Conversion Start Mode

When the arbiter has selected the request to be converted next, the handling of this channel depends on the current activity of the converter:

- Converter is currently idle: the conversion of the arbitration winner is started immediately.
- Current conversion has same or higher priority: the current conversion is completed, the conversion of the arbitration winner is started after that.
- Current conversion has lower priority: the action is user-configurable:

- **Wait-for-start mode:** the current conversion is completed, the conversion of the arbitration winner is started after that. This mode provides maximum throughput, but can produce a jitter for the higher priority conversion.

Example in [Figure 29-12](#):

Conversion A is requested (t_1) and started (t_2). Conversion B is then requested (t_3), but started only after completion of conversion A (t_4).

- **Cancel-inject-repeat mode:** the current conversion is aborted, the conversion of the arbitration winner is started after the abortion ($3 f_{VADC}$ cycles).

The aborted conversion request is restored in the corresponding request source and takes part again in the next arbitration round. This mode provides minimum jitter for the higher priority conversions, but reduces the overall throughput.

Example in [Figure 29-12](#):

Conversion A is requested (t_6) and started (t_7). Conversion B is then requested (t_8) and started (t_9), while conversion A is aborted but requested again. When conversion B is complete (t_{10}), conversion A is restarted.

Exception: If both requests target the same result register with wait-for-read mode active (see [Section 29.7.3](#)), the current conversion cannot be aborted.

Note: A cancelled conversion can be repeated automatically in each case, or it can be discarded if it was cancelled. This is selected for each source by bit RPTDIS in the corresponding source's mode register.

Versatile Analog-to-Digital Converter (VADC)

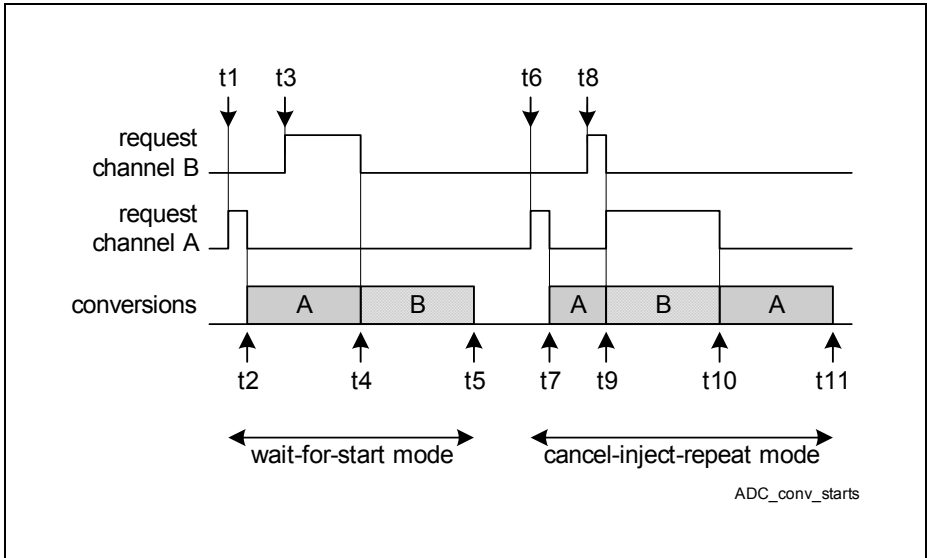


Figure 29-12 Conversion Start Modes

The conversion start mode can be individually programmed for each request source by bits in register **GxARBPR (x = 0 - 3)** and is applied to all channels requested by the source. In this example, channel A is issued by a request source with a lower priority than the request source requesting the conversion of channel B.

 Versatile Analog-to-Digital Converter (VADC)

29.6 Analog Input Channel Configuration

For each analog input channel a number of parameters can be configured that control the conversion of this channel. The channel control registers define the following parameters:

- **Channel Parameters:** The sample time for this channel and the data width of the result are defined via input classes. Each channel can select one of two classes of its own group or one of two global classes.
- **Reference selection:** an alternate reference voltage can be selected for most channels (exceptions are marked in [Section 29.12.3](#))
- **Result target:** The conversion result values are stored either in a group-specific result register or in the global result register. The group-specific result registers are selected in several ways:
 - channel-specific, selected by bitfield RESREG in register **G0CHCTRY** ($y = 0 - 15$) etc., with bitfield SRCRESREG = 0000_B
 - source-specific, selected by bitfield SRCRESREG in register **GxQCTRL0** ($x = 0 - 3$), **GxASCTRL** ($x = 0 - 3$) or **BRCTRL** with bitfield SRCRESREG $\neq 0000_B$
- **Result position:** The result values can be stored left-aligned or right-aligned. The exact position depends also on the configured result width and on the data accumulation mode.
See also [Figure 29-19 “Result Storage Options” on Page 29-106](#).
- **Compare with Standard Conversions (Limit Checking):** Channel events can be generated whenever a new result value becomes available. Channel event generation can be restricted to values that lie inside or outside a user-configurable band.
In Fast Compare Mode, channel events can be generated depending on the transitions of the (1-bit) result.
- **Broken Wire Detection:** This safety feature can detect a missing connection to an analog signal source (sensor).
- **Synchronization of Conversions:** Synchronized conversions are executed at the same time on several converters.

The **Alias Feature** redirects conversion requests for channels CH0 and/or CH1 to other channels.

29.6.1 Channel Parameters

Each analog input channel is configured by its associated channel control register.

Note: For the safety feature “Broken Wire Detection”, refer to [Section 29.9.1](#).

The following features can be defined for each channel:

- The conversion class defines the result width and the sample time
- Generation of channel events and the result value band, if used
- Target of the result defining the target register and the position within the register

Versatile Analog-to-Digital Converter (VADC)

G0CHCTry (y = 0 - 15)
Group 0, Channel y Ctrl. Reg. (0600_H + y * 0004_H) **Reset Value: 0000 0000_H**
G1CHCTry (y = 0 - 15)
Group 1, Channel y Ctrl. Reg. (0A00_H + y * 0004_H) **Reset Value: 0000 0000_H**
G2CHCTry (y = 0 - 15)
Group 2, Channel y Ctrl. Reg. (0E00_H + y * 0004_H) **Reset Value: 0000 0000_H**
G3CHCTry (y = 0 - 11)
Group 3, Channel y Ctrl. Reg. (1200_H + y * 0004_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0	BWD EN	BWD CH	0	0	0	0	0	0	0	RES POS	RES TBS	RESREG				
r	rw	rw	r	r	r	r	r	r	r	rw	rw	rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
BNDSELX			REF SEL	SY NC	CHEV MODE	BNDSELU	BNDSSELL	0	0	ICLSEL						
rw			rw	rw	rw	rw	rw	rw	r	r	rw					

Field	Bits	Type	Description
ICLSEL	[1:0]	rw	Input Class Select 00 _B Use group-specific class 0 01 _B Use group-specific class 1 10 _B Use global class 0 11 _B Use global class 1
0	[3:2]	r	Reserved, write 0, read as 0
BNDSSELL	[5:4]	rw	Lower Boundary Select¹⁾ 00 _B Use group-specific boundary 0 01 _B Use group-specific boundary 1 10 _B Use global boundary 0 11 _B Use global boundary 1
BNDSELU	[7:6]	rw	Upper Boundary Select¹⁾ 00 _B Use group-specific boundary 0 01 _B Use group-specific boundary 1 10 _B Use global boundary 0 11 _B Use global boundary 1

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
CHEVMODE	[9:8]	rw	Channel Event Mode Generate a channel event either in normal compare mode (NCM) with limit checking ²⁾ or in Fast Compare Mode (FCM) ³⁾ 00 _B Never 01 _B NCM: If result is inside the boundary band FCM: If result becomes high (above cmp. val.) 10 _B NCM: If result is outside the boundary band FCM: If result becomes low (below cmp. val.) 11 _B NCM: Always (ignore band) FCM: If result switches to either level
SYNC	10	rw	Synchronization Request 0 _B No synchroniz. request, standalone operation 1 _B Request a synchronized conversion of this channel (only taken into account for a master)
REFSEL	11	rw	Reference Input Selection Defines the reference voltage input to be used for conversions on this channel. 0 _B Standard reference input V_{AREF} 1 _B Alternate reference input from CH0 ⁴⁾
BNDSELX	[15:12]	rw	Boundary Extension¹⁾ 0000 _B Standard mode: select boundaries via BNDSELU/BNDSSELL 0001 _B Use result reg. GxRES1 as upper boundary ... 1111 _B Use result reg. GxRES15 as upper boundary
RESREG	[19:16]	rw	Result Register 0000 _B Store result in group result register GxRES0 ... 1111 _B Store result in group result register GxRES15
RESTBS	20	rw	Result Target for Background Source 0 _B Store results in the selected group result register register 1 _B Store results in the global result register
RESPOS	21	rw	Result Position 0 _B Store results left-aligned 1 _B Store results right-aligned
0	[27:22]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
BWDCH	[29:28]	rw	Broken Wire Detection Channel 00 _B Select V_{AREF} 01 _B Select V_{AGND} 10 _B Reserved 11 _B Reserved
BWDEN	30	rw	Broken Wire Detection Enable 0 _B Normal operation 1 _B Additional preparation phase is enabled
0	31	r	Reserved, write 0, read as 0

- 1) While $BNDSELX \neq 0000_B$, bitfields $BNDSELU$ and $BNDSELL$ are concatenated and select the corresponding result register as lower boundary.
- 2) The boundary band is defined as the area where the result is less than or equal to the selected upper boundary and greater than or equal to the selected lower boundary, see [Section 29.6.4](#).
- 3) The result is bit FCR in the selected result register.
- 4) Some channels cannot select an alternate reference.

Input Class Registers

The group-specific input class registers define the sample time and data conversion mode for each channel of the respective group that selects them via bitfield ICLSEL in its channel control register GxCHCTR_x.

The global input class registers define the sample time and data conversion mode for each channel of any group that selects them via bitfield ICLSEL in its channel control register GxCHCTR_x.

Versatile Analog-to-Digital Converter (VADC)
GxICLASS0 (x = 0 - 3)
Input Class Register 0, Group x

$$(x * 0400_H + 04A0_H)$$

Reset Value: 0000 0000_H
GxICLASS1 (x = 0 - 3)
Input Class Register 1, Group x

$$(x * 0400_H + 04A4_H)$$

Reset Value: 0000 0000_H
GLOBICLASSy (y = 0 - 1)
Input Class Register y, Global

$$(00A0_H + y * 0004_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	CME		0	0	0	STCE					
r	r	r	r	r	rw		r	r	r	rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	CMS		0	0	0	STCS					
r	r	r	r	r	rw		r	r	r	rw					

Field	Bits	Type	Description
STCS	[4:0]	rw	Sample Time Control for Standard Conversions Number of additional clock cycles to be added to the minimum sample phase of 2 analog clock cycles: Coding and resulting sample time see Table 29-4 . For conversions of external channels, the value from bitfield STCE can be used.
0	[7:5]	r	Reserved, write 0, read as 0
CMS	[10:8]	rw	Conversion Mode for Standard Conversions 000 _B 12-bit conversion 001 _B 10-bit conversion 010 _B 8-bit conversion 011 _B Reserved 100 _B Reserved 101 _B 10-bit fast compare mode 110 _B Reserved 111 _B Reserved
0	[15:11]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
STCE	[20:16]	rw	Sample Time Control for EMUX Conversions Number of additional clock cycles to be added to the minimum sample phase of 2 analog clock cycles: Coding and resulting sample time see Table 29-4 . For conversions of standard channels, the value from bitfield STCS is used.
0	[23:21]	r	Reserved, write 0, read as 0
CME	[26:24]	rw	Conversion Mode for EMUX Conversions 000 _B 12-bit conversion 001 _B 10-bit conversion 010 _B 8-bit conversion 011 _B Reserved 100 _B Reserved 101 _B 10-bit fast compare mode 110 _B Reserved 111 _B Reserved
0	[31:27]	r	Reserved, write 0, read as 0

Table 29-4 Sample Time Coding

STCS / STCE	Additional Clock Cycles	Sample Time
0 0000 _B	0	$2 / f_{\text{ADCI}}$
0 0001 _B	1	$3 / f_{\text{ADCI}}$
...
0 1111 _B	15	$17 / f_{\text{ADCI}}$
1 0000 _B	16	$18 / f_{\text{ADCI}}$
1 0001 _B	32	$34 / f_{\text{ADCI}}$
...
1 1110 _B	240	$242 / f_{\text{ADCI}}$
1 1111 _B	256	$258 / f_{\text{ADCI}}$

Note: Find more information in section **“Conversion Timing” on Page 29-93**.

Versatile Analog-to-Digital Converter (VADC)

29.6.2 Alias Feature

The Alias Feature redirects conversion requests for channels CH0 and/or CH1 to other channel numbers. This feature can be used to trigger conversions of the same input channel by independent events and to store the conversion results in different result registers.

- The same signal can be measured twice without the need to read out the conversion result to avoid data loss. This allows triggering both conversions quickly one after the other and being independent from CPU/DMA service request latency.
- The sensor signal is connected to only one analog input (instead of two analog inputs). This saves input pins in low-cost applications and only the leakage of one input has to be considered in the error calculation.
- Even if the analog input CH0 is used as alternative reference (see [Figure 29-13](#)), the internal trigger and data handling features for channel CH0 can be used.
- The channel settings for both conversions can be different (boundary values, service requests, etc.).

In typical low-cost AC-drive applications, only one common current sensor is used to determine the phase currents. Depending on the applied PWM pattern, the measured value has different meanings and the sample points have to be precisely located in the PWM period. [Figure 29-13](#) shows an example where the sensor signal is connected to one input channel (CHx) but two conversions are triggered for two different channels (CHx and CH0). With the alias feature, a conversion request for CH0 leads to a conversion of the analog input CHx instead of CH0, but taking into account the settings for CH0. Although the same analog input (CHx) has been measured, the conversion results can be stored and read out from the result registers RESx (conversion triggered for CHx) and RESy (conversion triggered for CH0). Additionally, different interrupts or limit boundaries can be selected, enabled or disabled.

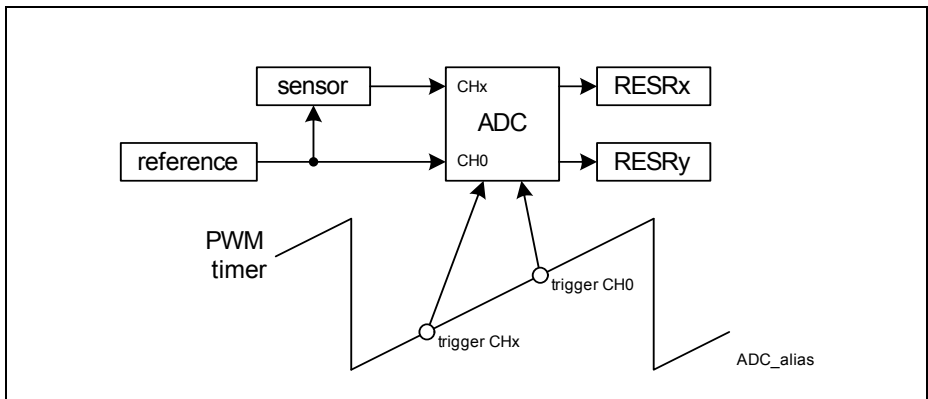


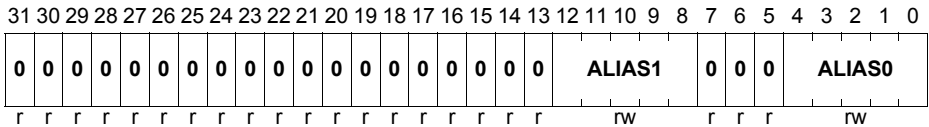
Figure 29-13 Alias Feature

Versatile Analog-to-Digital Converter (VADC)

The alias register can replace the channel numbers of channels CH0 and CH1 with another channel number. The reset value disables this redirection.

GxALIAS (x = 0 - 3)

Alias Register, Group x (x * 0400_H + 04B0_H) **Reset Value: 0000 0100_H**



Field	Bits	Type	Description
ALIAS0	[4:0]	rw	Alias Value for CH0 Conversion Requests Indicates the channel that is converted instead of channel CH0. The conversion is done with the settings defined for channel CH0.
0	[7:5]	r	Reserved, write 0, read as 0
ALIAS1	[12:8]	rw	Alias Value for CH1 Conversion Requests Indicates the channel that is converted instead of channel CH1. The conversion is done with the settings defined for channel CH1.
0	[31:13]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

29.6.3 Conversion Modes

A conversion can be executed in several ways. The conversion mode is selected according to the requested resolution of the digital result and according to the acceptable conversion time ([Section 29.6.7](#)).

Use bitfield CMS/CME in register **GxICLASS0 (x = 0 - 3)** etc. to select a mode.

Standard Conversions

A standard conversion returns a result value with a predefined resolution. 8-bit, 10-bit, and 12-bit resolution can be selected.

These result values can be accumulated, filtered, or used for digital limit checking and determination of extrema.

Note: The converters can operate with and without post-calibration.

Fast Compare Mode

In Fast Compare Mode, the selected input voltage is directly compared with a digital value that is stored in the corresponding result register. This compare operation returns a binary result indicating if the compared input voltage is above or below the given reference value. This result is generated quickly and thus supports monitoring of boundary values.

Fast Compare Mode uses a 10-bit compare value stored left-aligned at bit position 11. Separate positive and negative delta values define an arbitrary hysteresis band.

Selecting Compare Values

Values for digital or analog compare operations can be selected from several sources. The separate GxBOUND registers provide software-defined compare values. Compare values can also be taken from a result register where it can be provided by another channel building a reference.

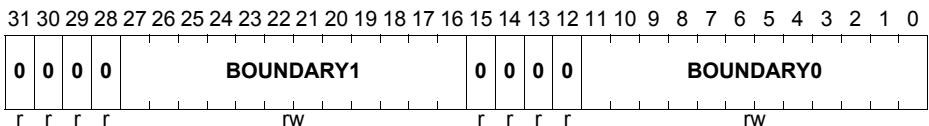
In Fast Compare Mode, the result registers provide the compare value while the bitfields of local GxBOUND registers define positive and negative delta values.

Versatile Analog-to-Digital Converter (VADC)

The local boundary register GxBOUND defines group-specific boundary values or delta limits for Fast Compare Mode.

The global boundary register GLOBBOUND defines general compare values for all channels.

Depending on the conversion width, the respective left 12/10/8 bits of a bitfield are used. For 10/8-bit results, the lower 2/4 bits must be zero!

GxBOUND (x = 0 - 3)
Boundary Select Register, Group x
 $(x * 0400_H + 04B8_H)$
Reset Value: 0000 0000_H
GLOBBOUND
Global Boundary Select Register (00B8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
BOUNDARY0	[11:0]	rw	Boundary Value 0 for Limit Checking Standard Mode: This value is compared against the left-aligned conversion result. Fast Compare Mode: This value is added to the reference value (upper delta).
0	[15:12]	r	Reserved, write 0, read as 0
BOUNDARY1	[27:16]	rw	Boundary Value 1 for Limit Checking Standard Mode: This value is compared against the left-aligned conversion result. Fast Compare Mode: This value is subtracted from the reference value (lower delta).
0	[31:28]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

29.6.4 Compare with Standard Conversions (Limit Checking)

The limit checking mechanism can automatically compare each digital conversion result to an upper and a lower boundary value. A channel event can then be generated when the result of a conversion/comparison is inside or outside a user-defined band (see bitfield CHEVMODE and [Figure 29-14](#)).

This feature supports automatic range monitoring and minimizes the CPU load by issuing service requests only under certain predefined conditions.

Note: Channel events can also be generated for each result value (ignoring the band) or they can be suppressed completely.

The boundary values to which results are compared can be selected from several sources (see register GxCHCTRY).

While bitfield BNDSELX = 0000_B, bitfields BNDSELU and BNDSELL select the valid upper/lower boundary value either from the group-specific boundary register **GxBOUND (x = 0 - 3)** or from the global boundary register **GLOBBOUND**. The group boundary register can be selected for each channel of the respective group, the global boundary register can be selected by each available channel.

Otherwise, the compare values are taken from result registers, where bitfield BNDSELX selects the upper boundary value (GxRES1 ... GxRES15), the concatenated bitfields BNDSELU||BNDSELL select the lower boundary value (GxRES0 ... GxRES15).

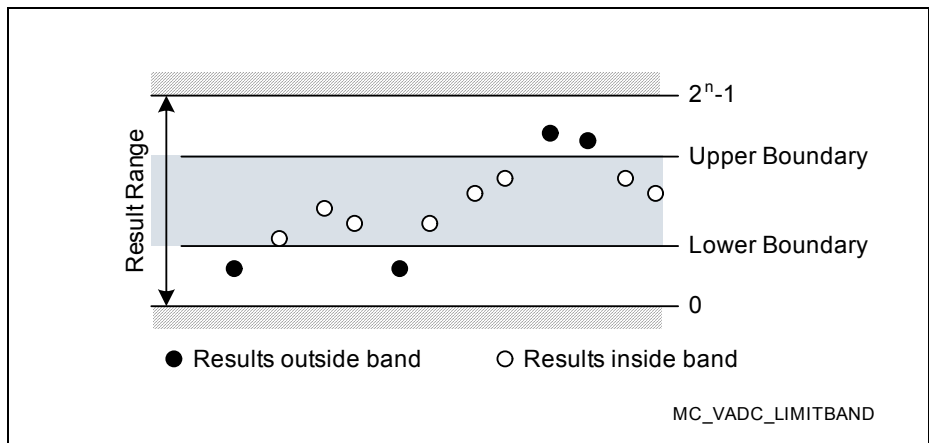


Figure 29-14 Result Monitoring through Limit Checking

Versatile Analog-to-Digital Converter (VADC)

A result value is considered inside the defined band when both of the following conditions are true:

- the value is less than or equal to the selected upper boundary
- the value is greater than or equal to the selected lower boundary

The result range can also be divided into two areas:

To select the lower part as valid band, set the lower boundary to the minimum value (000_H) and set the upper boundary to the highest intended value.

To select the upper part as valid band, set the upper boundary to the maximum value (FFF_H) and set the lower boundary to the lowest intended value.

Finding Extrema (Peak Detection)

The limit checking mechanism uses standard conversions and, therefore, always can provide the actual conversion result that was used for comparison. Combining this with a special FIFO mode, that only updates the corresponding FIFO stage if the result was above (or below) the current value of the stage, provides the usual conversion results and at the same time stores the highest (or lowest) result of a conversion sequence. For this operation the FIFO stage below the standard result must be selected as the compare value. Mode selection is done via bitfield FEN in register GxRCRy.

Before starting a peak detection sequence, write a reasonable start value to the result bitfield in the peak result register (e.g. 0000_H to find the maximum and $FFFF_H$ to find the minimum).

29.6.5 Utilizing Fast Compare Mode

In Fast Compare Mode, the input signal is directly compared to a value stored in bitfield RESULT in the associated result register. This comparison just provides a binary result (above/below), which is available in bit FCR in the same result register. If the exact result value is not required, this saves conversion time. A channel event can then be generated when the input signal becomes higher (or lower) than the compare value (see bitfield CHEVMODE and [Figure 29-15](#)).

The compare value in Fast Compare Mode is taken from the result register. Bitfields BOUNDARY1 and BOUNDARY0 in register **GxBOUND (x = 0 - 3)** define delta limits in this case. These deltas are added to (or subtracted from) the original compare value and allow defining an arbitrary hysteresis band.

The actual used compare value depends on the Fast Compare Result FCR (see registers **GORES_y (y = 0 - 15)**, etc.):

- GxRES_y.FCR = 0: reference value + upper delta
(GxRES_y.RESULT + GxBOUND.BOUNDARY0)
- GxRES_y.FCR = 1: reference value - lower delta
(GxRES_y.RESULT - GxBOUND.BOUNDARY1)

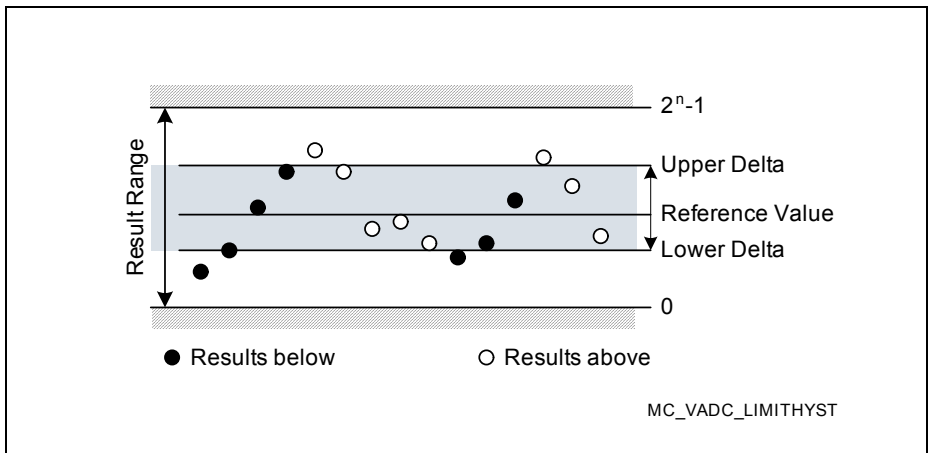


Figure 29-15 Result Monitoring through Compare with Hysteresis

29.6.6 Boundary Flag Control

Both limit checking mechanisms can be configured to automatically control the boundary flags. These boundary flags are also available as control signals for other modules. The flags can be set or cleared when the defined level is exceeded and the polarity of the output signal can be selected. A gate signal can be selected to enable the boundary flag operation while the gate is active.

Each boundary flag is available at group-specific output lines. Node pointers additionally route them to one of four boundary signals or one of the associated common service request lines, see register **GxBFLNP (x = 0 - 3)**.

For standard conversions, a boundary flag will be set/cleared when the conversion result is above the defined band, and will be cleared/set when the conversion result is below the defined band.

The band between the two boundary values defines a hysteresis for setting/clearing the boundary flags.

Using this feature on three channels that monitor linear hall elements can produce signals to feed the three hall position inputs of a unit that generates the corresponding PWM control signals.

In Fast Compare Mode, a boundary flag reflects the result of the comparisons, i.e. it will be set/cleared when the compared signal level is above the compare value, and will be cleared/set when the signal level is below the compare value. The delta values define a hysteresis band around the compare value.

Note: Clear register GxBOUND (i.e. the deltas) if a hysteresis is not wanted.

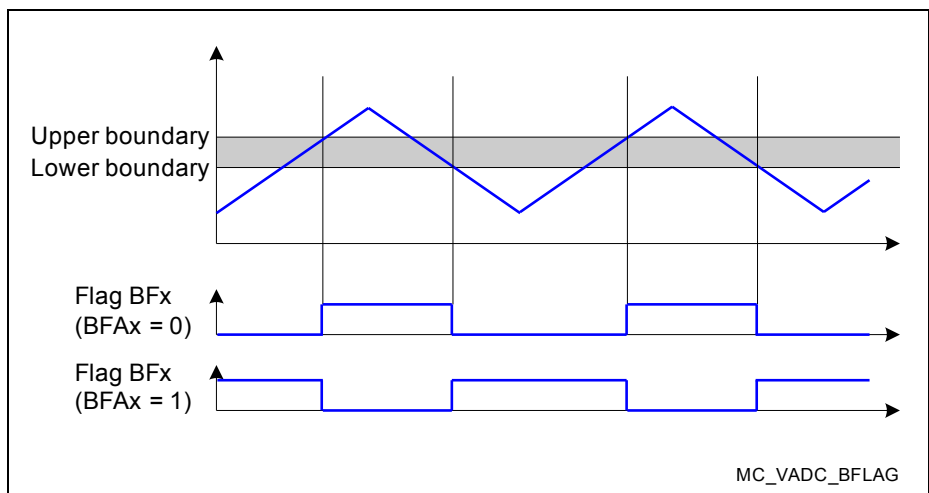


Figure 29-16 Boundary Flag Switching

Versatile Analog-to-Digital Converter (VADC)

Boundary flags can be switched by each compare operation, or the influence of compare operations can be restricted to the active phases of the corresponding request source gate signal (see **GxBFLC (x = 0 - 3)**).

Note: If a boundary flag is used together with Fast Compare Mode, it is recommended not to direct results from other channels to the corresponding result register.

The output signal derived from a boundary flag can be controlled by other modules. A select input and a data input are available to temporarily replace the boundary flag signal before sending it to the output pin (see **Figure 29-17**).

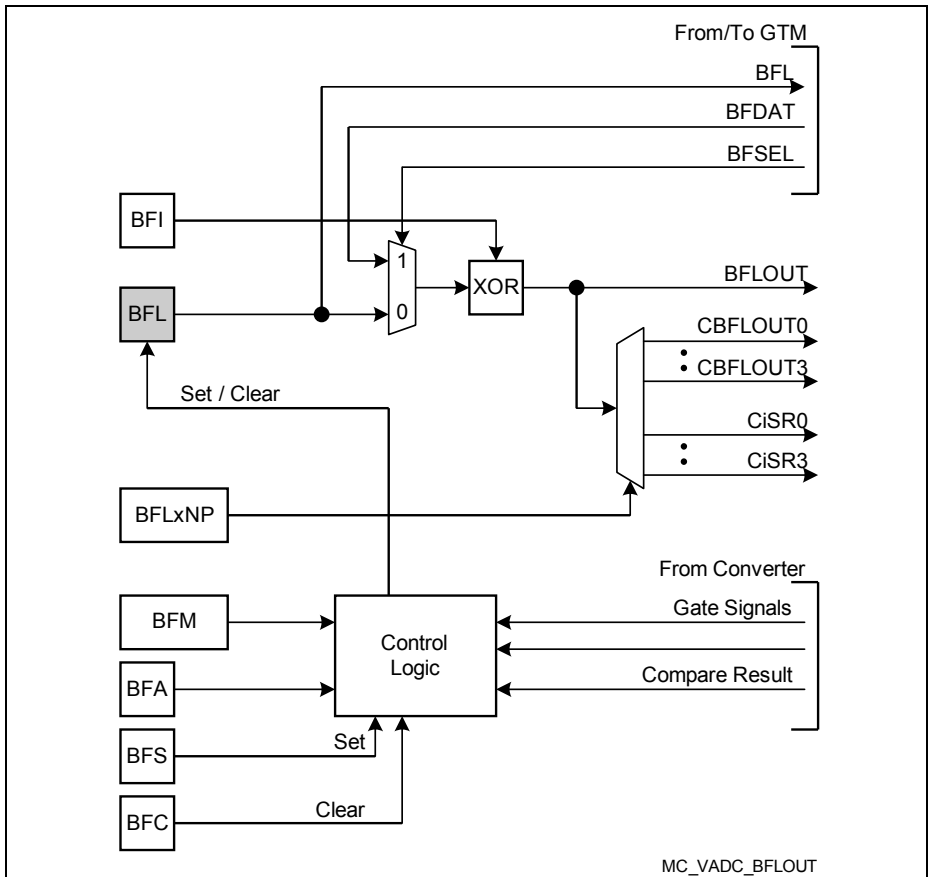


Figure 29-17 Boundary Flag Control

A boundary flag BFLy is assigned to result register GxRESy and thus to an arbitrary channel.

Versatile Analog-to-Digital Converter (VADC)

The Boundary Flag Register holds the boundary flags themselves together with bits to select the activation condition and the output signal polarity for each flag.

GxBFL (x = 0 - 3)
Boundary Flag Register, Group x
 $(x * 0400_H + 04C8_H)$
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	BFI 3	BFI 2	BFI 1	BFI 0
r	r	r	r	r	r	r	r	r	r	r	r	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	BFA 3	BFA 2	BFA 1	BFA 0	0	0	0	0	BFL 3	BFL 2	BFL 1	BFL 0
r	r	r	r	rw	rw	rw	rw	r	r	r	r	rh	rh	rh	rh

Field	Bits	Type	Description
BFLy (y = 0 - 3)	y	rh	Boundary Flag y 0 _B Passive state: result has not yet crossed the activation boundary (see bitfield BFAy), or selected gate signal is inactive, or this boundary flag is disabled 1 _B Active state: result has crossed the activation boundary
0	[7:4]	r	Reserved, write 0, read as 0
BFAy (y = 0 - 3)	8 + y	rw	Boundary Flag y Activation Select 0 _B Set boundary flag BFLy if result is above the defined band or compare value, clear if below 1 _B Set boundary flag BFLy if result is below the defined band or compare value, clear if above
0	[15:12]	r	Reserved, write 0, read as 0
BFLy (y = 0 - 3)	16 + y	rw	Boundary Flag y Inversion Control 0 _B Use BFLy directly 1 _B Invert value and use $\overline{\text{BFLy}}$
0	[31:20]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

The Boundary Flag Software Register provides means to set or clear each flag by software.

GxBFLS (x = 0 - 3)
Boundary Flag Software Register, Group x
 $(x * 0400_H + 04CC_H)$
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	BFS 3	BFS 2	BFS 1	BFS 0
r	r	r	r	r	r	r	r	r	r	r	r	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	BFC 3	BFC 2	BFC 1	BFC 0
r	r	r	r	r	r	r	r	r	r	r	r	w	w	w	w

Field	Bits	Type	Description
BFCy (y = 0 - 3)	y	w	Boundary Flag y Clear 0 _B No action 1 _B Clear bit BFLy
0	[15:4]	r	Reserved, write 0, read as 0
BFSy (y = 0 - 3)	16 + y	w	Boundary Flag y Set 0 _B No action 1 _B Set bit BFLy
0	[31:20]	r	Reserved, write 0, read as 0

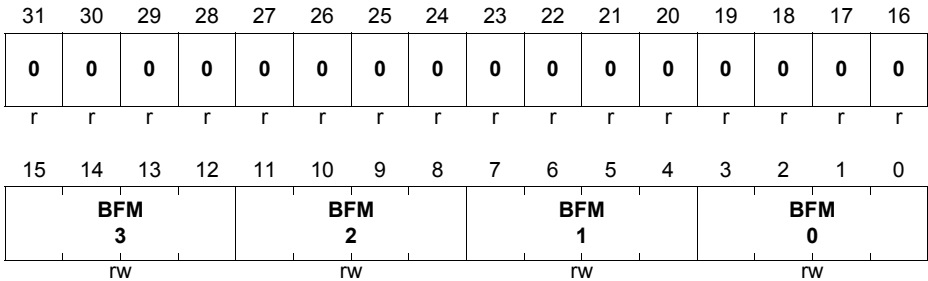
Note: If a boundary flag is used together with Fast Compare Mode, it is recommended not to direct results from other channels to the corresponding result register.

Versatile Analog-to-Digital Converter (VADC)

The Boundary Flag Control Register selects the basic operation of the boundary flags.

GxBFLC (x = 0 - 3)
Boundary Flag Control Register, Group x

$$(x * 0400_H + 04D0_H)$$

Reset Value: 0000 0000_H


Field	Bits	Type	Description
BFM0, BFM1, BFM2, BFM3	[3:0], [7:4], [11:8], [15:12]	rw	Boundary Flag y Mode Control 0000 _B Disable boundary flag, BFLy is not changed 0001 _B Always enable boundary flag (follow compare results) 0010 _B Enable boundary flag while gate of source 0 is active, clear BFLy while gate is inactive 0011 _B Enable boundary flag while gate of source 1 is active, clear BFLy while gate is inactive others: Reserved
0	[31:16]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

The Boundary Flag Node Pointer Register directs signal VADCGxBFLy to alternate on-chip connections with other modules (in addition to the group-specific outputs). Possible targets are the corresponding common service request lines or the common boundary flag outputs (CBFLOUT0 ... CBFLOUT3).

GxBFLNP (x = 0 - 3)
Boundary Flag Node Pointer Register, Group x
 $(x * 0400_H + 04D4_H)$
Reset Value: 0000 FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BFL3NP				BFL2NP				BFL1NP				BFL0NP			
rw				rw				rw				rw			

Field	Bits	Type	Description
BFL0NP, BFL1NP, BFL2NP, BFL3NP	[3:0], [7:4], [11:8], [15:12]	rw	Boundary Flag y Node Pointer 0000 _B Select common boundary flag output 0 ... 0011 _B Select common boundary flag output 3 0100 _B Select shared service request line 0 ... 0111 _B Select shared service request line 3 1111 _B Disabled, no common output signal others: Reserved <i>Note: For shared service request lines see common groups in Section 29.12.1.</i>
0	[31:16]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)
29.6.7 Conversion Timing

The total time required for a conversion comprises the time from the start of the sample phase¹⁾ until the availability of the result.

The frequency at which conversions are triggered also depends on several configurable factors:

- The selected conversion time, according to the input class definitions. For conversions using an external multiplexer, also the extended sample times count.
- Delays induced by cancelled conversions that must be repeated.
- Delays due to equidistant sampling of other channels.
- The configured arbitration cycle time.
- The frequency of external trigger signals, if enabled.

The conversion timing depends on several user-definable factors:

- The ADC conversion clock frequency, where $f_{\text{ADCI}} = f_{\text{VADC}} / (\text{DIVA} + 1)^2$
- The selected sample time, where $t_{\text{S}} = (2 + \text{STC}) \times t_{\text{ADCI}}$
(STC = additional sample time, see also [Table 29-4](#))
- The selected operating mode (normal conversion / fast compare mode)
- The result width N (8/10/12 bits) for normal conversions
- The post-calibration time PC, if selected (PC = 2, otherwise 0)
- Synchronization steps done at module clock speed

The conversion time is the sum of sample time, conversion steps, and synchronization. It can be computed with the following formulas:

Standard conversions: $t_{\text{CN}} = (2 + \text{STC} + \text{N} + \text{PC}) \times t_{\text{ADCI}} + 2 \times t_{\text{VADC}}$

Fast compare mode: $t_{\text{CN}} = (2 + \text{STC} + 2) \times t_{\text{ADCI}} + 2 \times t_{\text{VADC}}$

Timing Examples

System assumptions:

$f_{\text{VADC}} = 100 \text{ MHz}$ i.e. $t_{\text{VADC}} = 10 \text{ ns}$, $\text{DIVA} = 5$, $f_{\text{ADCI}} = 16.67 \text{ MHz}$ i.e. $t_{\text{ADCI}} = 60 \text{ ns}$

According to the given formulas the following minimum conversion times can be achieved:

12-bit calibrated conversion:

$$t_{\text{CN}12\text{C}} = (2 + 12 + 2) \times t_{\text{ADCI}} + 2 \times t_{\text{VADC}} = 16 \times 60 \text{ ns} + 2 \times 10 \text{ ns} = 980 \text{ ns}$$

10-bit uncalibrated conversion:

$$t_{\text{CN}10} = (2 + 10) \times t_{\text{ADCI}} + 2 \times t_{\text{VADC}} = 12 \times 60 \text{ ns} + 2 \times 10 \text{ ns} = 740 \text{ ns}$$

Fast comparison:

$$t_{\text{FCM}} = (2 + 2) \times t_{\text{ADCI}} + 2 \times t_{\text{VADC}} = 4 \times 60 \text{ ns} + 2 \times 10 \text{ ns} = 340 \text{ ns}$$

1) The time from the trigger event that requests the corresponding conversion until the start of the sample phase depends on the arbitration and can, therefore, only be determined when the system setup is known.

2) The minimum prescaler factor for calibrated converters is 2.

Versatile Analog-to-Digital Converter (VADC)

29.7 Conversion Result Handling

The A/D converters can preprocess the conversions result data to a certain extent before storing them for retrieval by the CPU or a DMA channel. This supports the subsequent handling of result data by the application software.

Conversion result handling comprises the following functions:

- **Storage of Conversion Results** to user-configurable registers
- **Data Alignment** according to result width and endianness
- **Wait-for-Read Mode** to avoid loss of data
- **Result Event Generation**
- Data reduction or anti-aliasing filtering (see [Section 29.7.6](#))

29.7.1 Storage of Conversion Results

The conversion result values of a certain group can be stored in one of the 16 associated group result registers or in the common global result register (can be used, for example, for the channels of the background source (see [Selecting a Result Register](#)).

This structure provides different locations for the conversion results of different sets of channels. Depending on the application needs (data reduction, auto-scan, alias feature, etc.), the user can distribute the conversion results to minimize CPU load and/or optimize the performance of DMA transfers.

Each result register has an individual data valid flag (VF) associated with it. This flag indicates when “new” valid data has been stored in the corresponding result register and can be read out.

For standard conversions, result values are available in bitfield RESULT. Conversions in Fast Compare Mode use bitfield RESULT for the reference value, so the result of the operation is stored in bit FCR.

Result registers can be read via two different views. These views use different addresses but access the same register data:

- When a result register is read via the **application view**, the corresponding valid flag is automatically cleared when the result is read. This provides an easy handshake between result generation and retrieval. This also supports wait-for-read mode.
- When a result register is read via the **debug view**, the corresponding valid flag remains unchanged when the result is read. This supports debugging by delivering the result value without disturbing the handshake with the application.

The application can retrieve conversion results through several result registers:

- Group result register:
Returns the result value and the channel number
- Global result register:
Returns the result value and the channel number and the group number

Versatile Analog-to-Digital Converter (VADC)

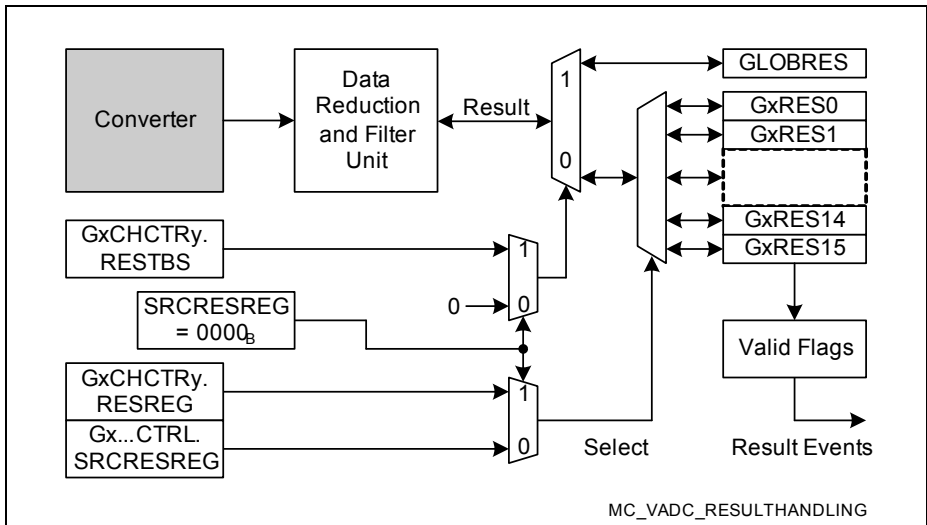


Figure 29-18 Conversion Result Storage

Selecting a Result Register

Conversion results are stored in result registers that can be assigned by the user according to the requirements of the application. The following bitfields direct the results to a register:

- SRCRESREG in register **GxQCTRL0 (x = 0 - 3)**, **GxASCTRL (x = 0 - 3)** or **BRCTRL**
Selects the group-specific result register GxRES1 ... GxRES15 when source-specific result registers are used
- RESTBS in register GxCHCTRY
Selects the global result register for results requested by the background source
- RESREG in register GxCHCTRY
Selects the group-specific result register GxRES0 ... GxRES15 when channel-specific result registers are used (see below).

Using source-specific result registers allows separating results from the same channel that are requested by different request sources. Usually these request sources are used by different tasks and are triggered at different times.

Versatile Analog-to-Digital Converter (VADC)

The group result control registers select the behavior of the result registers of a given group.

G0RCRy (y = 0 - 15)
Group 0 Result Control Reg. y (0680_H + y * 0004_H) Reset Value: 0000 0000_H
G1RCRy (y = 0 - 15)
Group 1 Result Control Reg. y (0A80_H + y * 0004_H) Reset Value: 0000 0000_H
G2RCRy (y = 0 - 15)
Group 2 Result Control Reg. y (0E80_H + y * 0004_H) Reset Value: 0000 0000_H
G3RCRy (y = 0 - 15)
Group 3 Result Control Reg. y (1280_H + y * 0004_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRG EN	0	0	0	0	FEN	WFR	0	0	DMM	DRCTR					
rw	r	r	r	r	rw	rw	r	r	rw	rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
0	[15:0]	r	Reserved, write 0, read as 0
DRCTR	[19:16]	rw	Data Reduction Control Defines how result values are stored/accumulated in this register for the final result. The data reduction counter DRC can be loaded from this bitfield. The function of bitfield DRCTR is determined by bitfield DMM.
DMM	[21:20]	rw	Data Modification Mode 00 _B Standard data reduction (accumulation) 01 _B Result filtering mode ¹⁾ 10 _B Difference mode 11 _B Reserved See “Data Modification” on Page 29-110
0	[23:22]	r	Reserved, write 0, read as 0
WFR	24	rw	Wait-for-Read Mode Enable 0 _B Overwrite mode 1 _B Wait-for-read mode enabled for this register

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
FEN	[26:25]	rw	FIFO Mode Enable 00 _B Separate result register 01 _B Part of a FIFO structure: copy each new valid result 10 _B Maximum mode: copy new result if bigger 11 _B Minimum mode: copy new result if smaller
0	[30:27]	r	Reserved, write 0, read as 0
SRGEN	31	rw	Service Request Generation Enable 0 _B No service request 1 _B Service request after a result event

1) The filter registers are cleared while bitfield DMM ≠ 01_B.

Versatile Analog-to-Digital Converter (VADC)

The group result registers provide a selectable storage location for all channels of a given group.

Note: The preset value used in fast compare mode is written to the respective result register. The debug result registers are not writable.

G0RESy (y = 0 - 15)

Group 0 Result Register y $(0700_H + y * 0004_H)$ **Reset Value: 0000 0000_H**

G1RESy (y = 0 - 15)

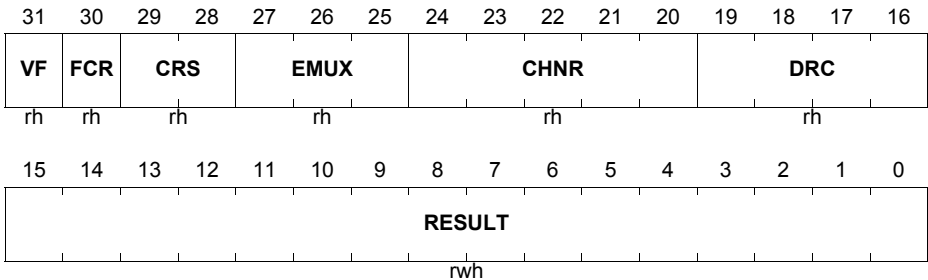
Group 1 Result Register y $(0B00_H + y * 0004_H)$ **Reset Value: 0000 0000_H**

G2RESy (y = 0 - 15)

Group 2 Result Register y $(0F00_H + y * 0004_H)$ **Reset Value: 0000 0000_H**

G3RESy (y = 0 - 15)

Group 3 Result Register y $(1300_H + y * 0004_H)$ **Reset Value: 0000 0000_H**



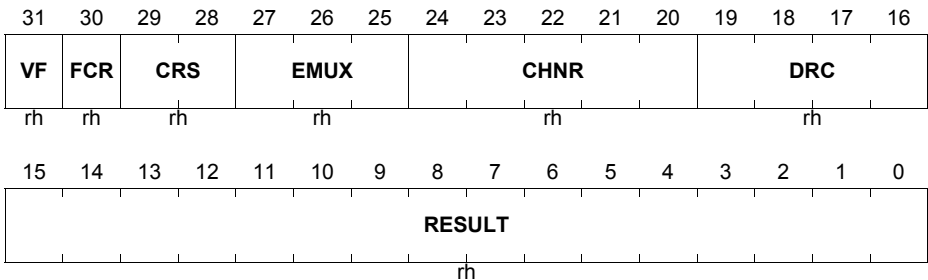
Field	Bits	Type	Description
RESULT	[15:0]	rwh	Result of Most Recent Conversion The position of the result bits within this bitfield depends on the configured operating mode. Refer to Section 29.7.2 .
DRC	[19:16]	rh	Data Reduction Counter Indicates the number of values still to be accumulated for the final result. The final result is available and valid flag VF is set when bitfield DRC becomes zero (by decrementing or by reload). See “Data Modification” on Page 29-110
CHNR	[24:20]	rh	Channel Number Indicates the channel number corresponding to the value in bitfield RESULT.

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
EMUX	[27:25]	rh	External Multiplexer Setting Indicates the setting of the external multiplexer, corresponding to the value in bitfield RESULT. <i>Note: Available in GxRES0 only. Use GxRES0 if EMUX information is required.</i>
CRS	[29:28]	rh	Converted Request Source Indicates the request source that as requested the conversion to which the result value in bitfield RESULT belongs. 00 _B Request source 0 01 _B Request source 1 10 _B Request source 2 11 _B Synchronized conversion
FCR	30	rh	Fast Compare Result Indicates the result of an operation in Fast Compare Mode. 0 _B Signal level was below compare value 1 _B Signal level was above compare value
VF	31	rh	Valid Flag Indicates a new result in bitfield RESULT or bit FCR. 0 _B No new result available 1 _B Bitfield RESULT has been updated with new result value and has not yet been read, or bit FCR has been updated

The debug view of the group result registers provides access to all result registers of a given group, however, without clearing the valid flag.

Versatile Analog-to-Digital Converter (VADC)

G0RESDy (y = 0 - 15)
Group 0 Result Reg. y, Debug ($0780_H + y * 0004_H$) **Reset Value:** 0000 0000_H
G1RESDy (y = 0 - 15)
Group 1 Result Reg. y, Debug ($0B80_H + y * 0004_H$) **Reset Value:** 0000 0000_H
G2RESDy (y = 0 - 15)
Group 2 Result Reg. y, Debug ($0F80_H + y * 0004_H$) **Reset Value:** 0000 0000_H
G3RESDy (y = 0 - 15)
Group 3 Result Reg. y, Debug ($1380_H + y * 0004_H$) **Reset Value:** 0000 0000_H


Field	Bits	Type	Description
RESULT	[15:0]	rh	Result of Most Recent Conversion The position of the result bits within this bitfield depends on the configured operating mode. Refer to Section 29.7.2 .
DRC	[19:16]	rh	Data Reduction Counter Indicates the number of values still to be accumulated for the final result. The final result is available and valid flag VF is set when bitfield DRC becomes zero (by decrementing or by reload). See “Data Modification” on Page 29-110
CHNR	[24:20]	rh	Channel Number Indicates the channel number corresponding to the value in bitfield RESULT.
EMUX	[27:25]	rh	External Multiplexer Setting Indicates the setting of the external multiplexer, corresponding to the value in bitfield RESULT. <i>Note: Available in GxRES0 only. Use GxRES0 if EMUX information is required.</i>

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
CRS	[29:28]	rh	Converted Request Source Indicates the request source that as requested the conversion to which the result value in bitfield RESULT belongs. 00 _B Request source 0 01 _B Request source 1 10 _B Request source 2 11 _B Synchronized conversion
FCR	30	rh	Fast Compare Result Indicates the result of an operation in Fast Compare Mode. 0 _B Signal level was below compare value 1 _B Signal level was above compare value
VF	31	rh	Valid Flag Indicates a new result in bitfield RESULT or bit FCR. 0 _B No new result available 1 _B Bitfield RESULT has been updated with new result value and has not yet been read, or bit FCR has been updated

Versatile Analog-to-Digital Converter (VADC)

The global result control register selects the behavior of the global result register.

GLOBRCR

Global Result Control Register (0280_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRGEN	0	0	0	0	0	0	WFR	0	0	0	0	DRCTR			
rw	r	r	r	r	r	r	rw	r	r	r	r	rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
0	[15:0]	r	Reserved, write 0, read as 0
DRCTR	[19:16]	rw	Data Reduction Control Defines how result values are stored/accumulated in this register for the final result. The data reduction counter DRC can be loaded from this bitfield. 0000 _B Data reduction disabled others: see “Function of Bitfield DRCTR” on Page 29-110¹⁾
0	[23:20]	r	Reserved, write 0, read as 0
WFR	24	rw	Wait-for-Read Mode Enable 0 _B Overwrite mode 1 _B Wait-for-read mode enabled for this register
0	[30:25]	r	Reserved, write 0, read as 0
SRGEN	31	rw	Service Request Generation Enable 0 _B No service request 1 _B Service request after a result event

1) Only standard data reduction is available for the global result register, i.e. DMM is assumed as 00_B.

Versatile Analog-to-Digital Converter (VADC)

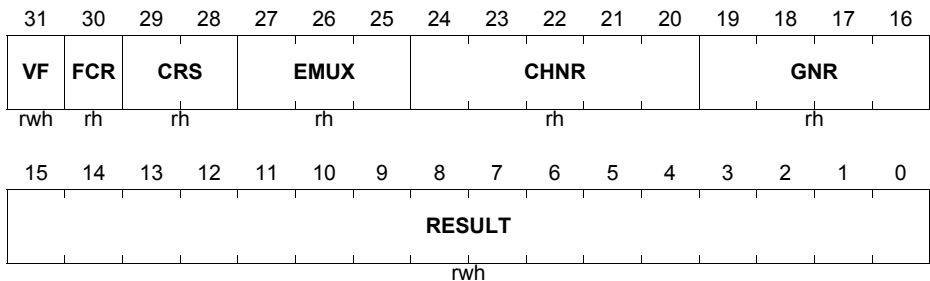
The global result register provides a common storage location for all channels of all groups.

GLOBRES

Global Result Register (0300_H) **Reset Value: 0000 0000_H**

GLOBRESD

Global Result Register, Debug (0380_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RESULT	[15:0]	rwh	Result of most recent conversion The position of the result bits within this bitfield depends on the configured operating mode. ¹⁾ Refer to Section 29.7.2 .
GNR	[19:16]	rh	Group Number Indicates the group to which the channel number in bitfield CHNR refers.
CHNR	[24:20]	rh	Channel Number Indicates the channel number corresponding to the value in bitfield RESULT.
EMUX	[27:25]	rh	External Multiplexer Setting Indicates the setting of the external multiplexer, corresponding to the value in bitfield RESULT.
CRS	[29:28]	rh	Converted Request Source Indicates the request source that as requested the conversion to which the result value in bitfield RESULT belongs.

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
FCR	30	rh	Fast Compare Result Indicates the result of an operation in Fast Compare Mode. 0 _B Signal level was below compare value 1 _B Signal level was above compare value
VF	31	rwh	Valid Flag Indicates a new result in bitfield RESULT or bit FCR. 0 _B Read access: No new valid data available Write access: No effect 1 _B Read access: Bitfield RESULT contains valid data and has not yet been read, or bit FCR has been updated Write access: Clear this valid flag and the data reduction counter (overrides a hardware set action) ¹⁾

1) Only writable in register GLOBRES, not in register GLOBRESD.

Versatile Analog-to-Digital Converter (VADC)

The valid flag register summarizes the valid flags of all result registers.

GxVFR (x = 0 - 3)

Valid Flag Register, Group x ($x * 0400_H + 05F8_H$) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VF15	VF14	VF13	VF12	VF11	VF10	VF9	VF8	VF7	VF6	VF5	VF4	VF3	VF2	VF1	VF0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
VFy (y = 0 - 15)	y	rwh	Valid Flag of Result Register x Indicates a new result in bitfield RESULT or in bit FCR. 0 _B Read access: No new valid data available Write access: No effect 1 _B Read access: Result register x contains valid data and has not yet been read, or bit FCR has been updated Write access: Clear this valid flag and bitfield DRC in register GxRESy (overrides a hardware set action)
0	[31:16]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

29.7.2 Data Alignment

The position of a conversion result value within the selected result register depends on 3 configurations (summary in **Figure 29-19**):

- The selected result width (12/10/8 bits, selected by the conversion mode)
- The selected result position (Left/Right-aligned)
- The selected data accumulation mode (data reduction)

These options provide the conversion results in a way that minimizes data handling for the application software.

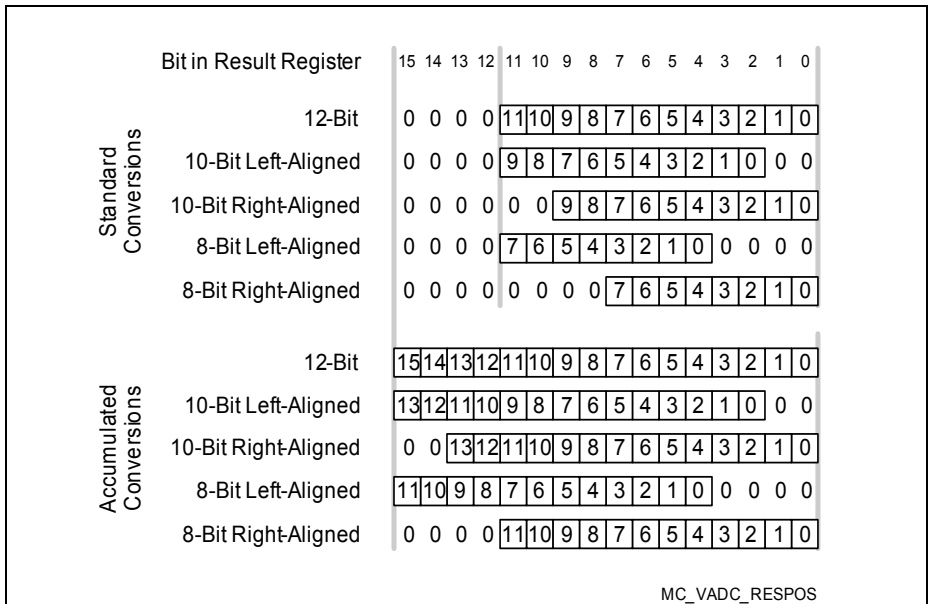


Figure 29-19 Result Storage Options

Bitfield RESULT can be written by software to provide the reference value for Fast Compare Mode. In this mode, bits 11-2 are evaluated, the other bits are ignored.

29.7.3 Wait-for-Read Mode

The wait-for-read mode prevents data loss due to overwriting a result register with a new conversion result before the CPU (or DMA) has read the previous data. For example, auto-scan conversion sequences or other sequences with “relaxed” timing requirements may use a common result register. However, the results come from different input channels, so an overwrite would destroy the result from the previous conversion¹⁾.

Wait-for-read mode automatically suspends the start of a conversion for this channel from this source until the current result has been read. So a conversion or a conversion sequence can be requested by a hardware or software trigger, while each conversion is only started after the result of the previous one has been read. This automatically aligns the conversion sequence with the CPU/DMA capability to read the formerly converted result (latency).

If wait-for-read mode is enabled for a result register (bit GxRCRy.WFR = 1), a request source does not generate a conversion request while the targeted result register contains valid data (indicated by the valid flag VF = 1) or if a currently running conversion targets the same result register.

If two request sources target the same result register with wait-for-read mode selected, a higher priority source cannot interrupt a lower priority conversion request started before the higher priority source has requested its conversion. Cancel-inject-repeat mode does not work in this case. In particular, this must be regarded if one of the involved sources is the background source (which usually has lowest priority). If the higher priority request targets a different result register, the lower priority conversion can be cancelled and repeated afterwards.

Note: Wait-for-read mode is ignored for synchronized conversions of synchronization slaves (see [Section 29.8](#)).

1) Repeated conversions of a single channel that use a separate result register will not destroy other results, but rather update their own previous result value. This way, always the actual signal data is available in the result register.

Versatile Analog-to-Digital Converter (VADC)

29.7.4 Result FIFO Buffer

Result registers can either be used as direct target for conversion results or they can be concatenated with other result registers of the same ADC group to form a result FIFO buffer (first-in-first-out buffer mechanism). A result FIFO stores several measurement results that can be read out later with a “relaxed” CPU response timing. It is possible to set up more than one FIFO buffer structure with the available result registers.

Result FIFO structures of two or more registers are built by concatenating result registers to their following “neighbor” result register (with next higher index, see [Figure 29-20](#)). This is enabled by setting bitfield GxRCRy.FEN = 01_B.

Conversion results are stored to the register with the highest index of a FIFO structure. Software reads the values from the FIFO register with the lowest index.

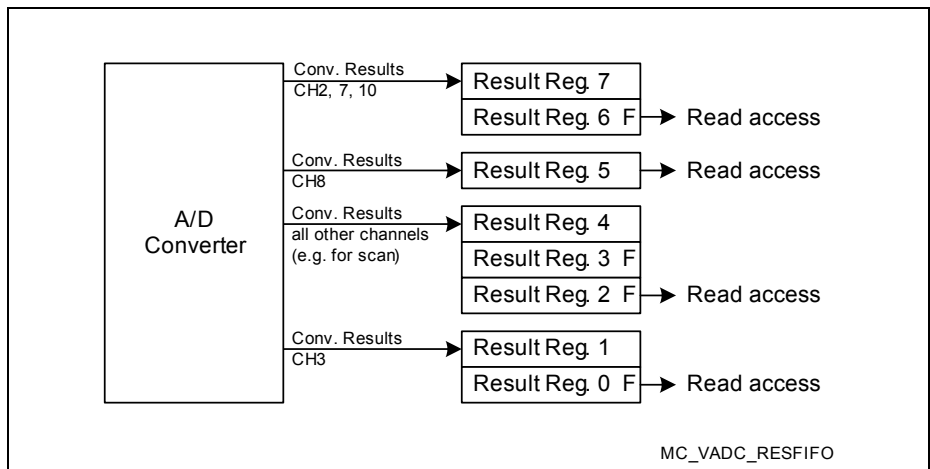


Figure 29-20 Result FIFO Buffers

In the example shown the result registers have been configured in the following way:

- 2-stage buffer consisting of result registers 7-6
- dedicated result register 5
- 3-stage buffer consisting of result registers 4-3-2
- 2-stage buffer consisting of result registers 1-0

[Table 29-5](#) summarizes the required configuration of result registers if they are combined to build result FIFO buffers.

Table 29-5 Properties of Result FIFO Registers

Function	Input Stage	Intermed. Stage	Output Stage
Result target	YES	no	no
Application read	no	no	YES
Data reduction mode	YES	no	no
Wait-for-read mode	YES	no	no
Result event interrupt	no	no	YES
FIFO enable (FEN)	00 _B	01 _B	01 _B
Registers in example	7, 4, 1	3	6, 2, 0

Note: If enabled, a result interrupt is generated for each data word in the FIFO.

29.7.5 Result Event Generation

A result event can be generated when a new value is stored to a result register. Result events can be restricted due to data accumulation and be generated only if the accumulation is complete.

Result events can also be suppressed completely.

Versatile Analog-to-Digital Converter (VADC)
29.7.6 Data Modification

The data resulting from conversions can be automatically modified before being used by an application. Several options can be selected (bitfield DMM in register **G0RCRy (y = 0 - 15)** etc.) which reduce the CPU/DMA load required to unload and/or process the conversion data.

- **Standard Data Reduction Mode (for GxRES0 ... GxRES15):**
Accumulates 2, 3, or 4 result values within each result register before generating a result interrupt. This can remove some noise from the input signal.
- **Result Filtering Mode (FIR, for GxRES7, GxRES15):**
Applies a 3rd order Finite Impulse Response Filter (FIR) with selectable coefficients to the conversion results for the selected result register.
- **Result Filtering Mode (IIR, for GxRES7, GxRES15):**
Applies a 1st order Infinite Impulse Response Filter (IIR) with selectable coefficients to the conversion results for the selected result register.
- **Difference Mode (for GxRES1 ... GxRES15):**
Subtracts the contents of result register GxRES0 from the conversion results for the selected result register. Bitfield DRCTR is not used in this mode.

Table 29-6 Function of Bitfield DRCTR

DRCTR	Standard Data Reduction Mode (DMM = 00_B)	DRCTR	Result Filtering Mode (DMM = 01_B)¹⁾
0000 _B	Data Reduction disabled	0000 _B	FIR filter: a=2, b=1, c=0
0001 _B	Accumulate 2 result values	0001 _B	FIR filter: a=1, b=2, c=0
0010 _B	Accumulate 3 result values	0010 _B	FIR filter: a=2, b=0, c=1
0011 _B	Accumulate 4 result values	0011 _B	FIR filter: a=1, b=1, c=1
0100 _B	Reserved	0100 _B	FIR filter: a=1, b=0, c=2
0101 _B	Reserved	0101 _B	FIR filter: a=3, b=1, c=0
0110 _B	Reserved	0110 _B	FIR filter: a=2, b=2, c=0
0111 _B	Reserved	0111 _B	FIR filter: a=1, b=3, c=0
1000 _B	Reserved	1000 _B	FIR filter: a=3, b=0, c=1
1001 _B	Reserved	1001 _B	FIR filter: a=2, b=1, c=1
1010 _B	Reserved	1010 _B	FIR filter: a=1, b=2, c=1
1011 _B	Reserved	1011 _B	FIR filter: a=2, b=0, c=2
1100 _B	Reserved	1100 _B	FIR filter: a=1, b=1, c=2
1101 _B	Reserved	1101 _B	FIR filter: a=1, b=0, c=3

Versatile Analog-to-Digital Converter (VADC)**Table 29-6 Function of Bitfield DRCTR (cont'd)**

DRCTR	Standard Data Reduction Mode (DMM = 00_B)	DRCTR	Result Filtering Mode (DMM = 01_B)¹⁾
1110 _B	Reserved	1110 _B	IIR filter: a=2, b=2
1111 _B	Reserved	1111 _B	IIR filter: a=3, b=4

1) The filter registers are cleared while bitfield DMM ≠ 01_B.

Versatile Analog-to-Digital Converter (VADC)

Standard Data Reduction Mode

The data reduction mode can be used as digital filter for anti-aliasing or decimation purposes. It accumulates a maximum of 4 conversion values to generate a final result.

Each result register can be individually enabled for data reduction, controlled by bitfield DRCTR in registers **G0RCRy (y = 0 - 15)**ff and **GLOBRCR**. The data reduction counter DRC indicates the actual status of the accumulation.

Note: Conversions for other result registers can be inserted between conversions to be accumulated.

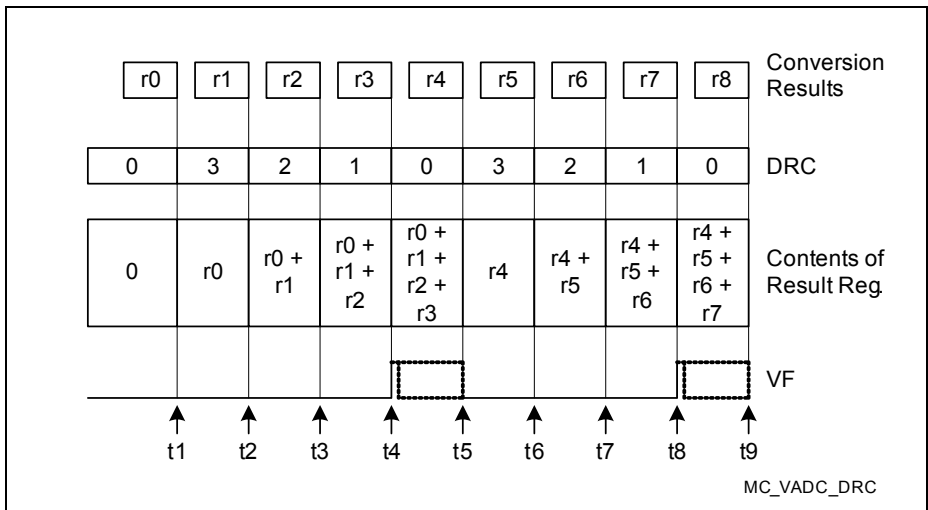


Figure 29-21 Standard Data Reduction Filter

This example shows a data reduction sequence of 4 accumulated conversion results. Eight conversion results (r0 ... r7) are accumulated and produce 2 final results.

When a conversion is complete and stores data to a result register that has data reduction mode enabled, the data handling is controlled by the data reduction counter DRC:

- If DRC = 0 (t1, t5, t9 in the example), the conversion result is stored to the register. DRC is loaded with the contents of bitfield DRCTR (i.e. the accumulation begins).
- If DRC > 0 (t2, t3, t4 and t6, t7, t8 in the example), the conversion result is added to the value in the result register. DRC is decremented by 1.
- If DRC becomes 0, either decremented from 1 (t4 and t8 in the example) or loaded from DRCTR, the valid bit for the respective result register is set and a result register event occurs.

Versatile Analog-to-Digital Converter (VADC)

The final result must be read before the next data reduction sequence starts (before t5 or t9 in the example). This automatically clears the valid flag.

*Note: Software can clear the data reduction counter DRC by clearing the corresponding valid Flag (via **GxVFR (x = 0 - 3)**).*

The response time to read the final data reduction results can be increased by associating the adjacent result register to build a result FIFO (see **Figure 29-22**). In this case, the final result of a data reduction sequence is loaded to the adjacent register. The value can be read from this register until the next data reduction sequence is finished (t8 in the 2nd example).

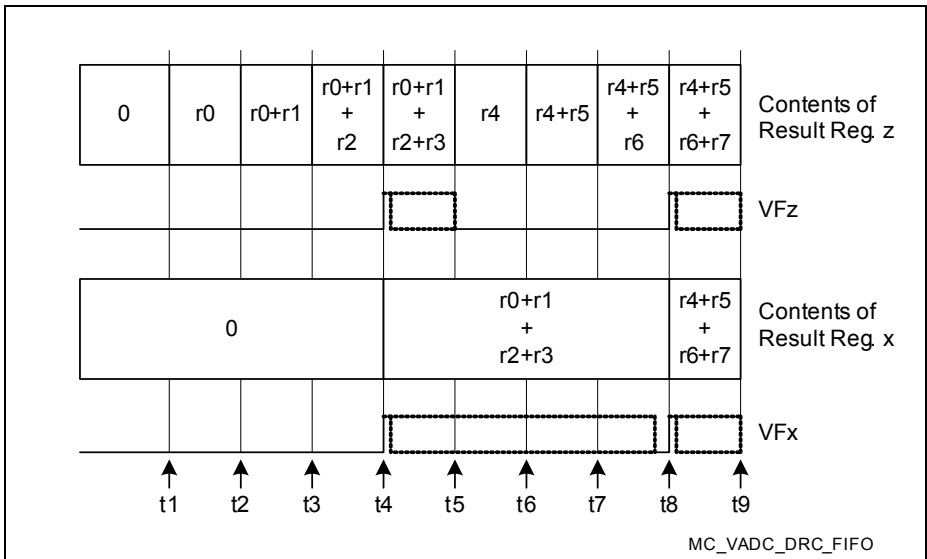


Figure 29-22 Standard Data Reduction Filter with FIFO Enabled

Versatile Analog-to-Digital Converter (VADC)

Finite Impulse Response Filter Mode (FIR)

The FIR filter (see [Figure 29-23](#)) provides 2 result buffers for intermediate results (RB1, RB2) and 3 configurable tap coefficients (a, b, c).

The conversion result and the intermediate result buffer values are added weighted with their respective coefficients to form the final value for the result register. Several predefined sets of coefficients can be selected via bitfield DRCTR (coding listed in [Table 29-6](#)) in registers **G0RCRy** ($y = 0 - 15$)ff and **GLOBRCR**. These coefficients lead to a gain of 3 or 4 to the ADC result producing a 14-bit value. The valid flag (VF) is activated for each sample after activation, i.e. for each sample generates a valid result.

Note: Conversions for other result registers can be inserted between conversions to be filtered.

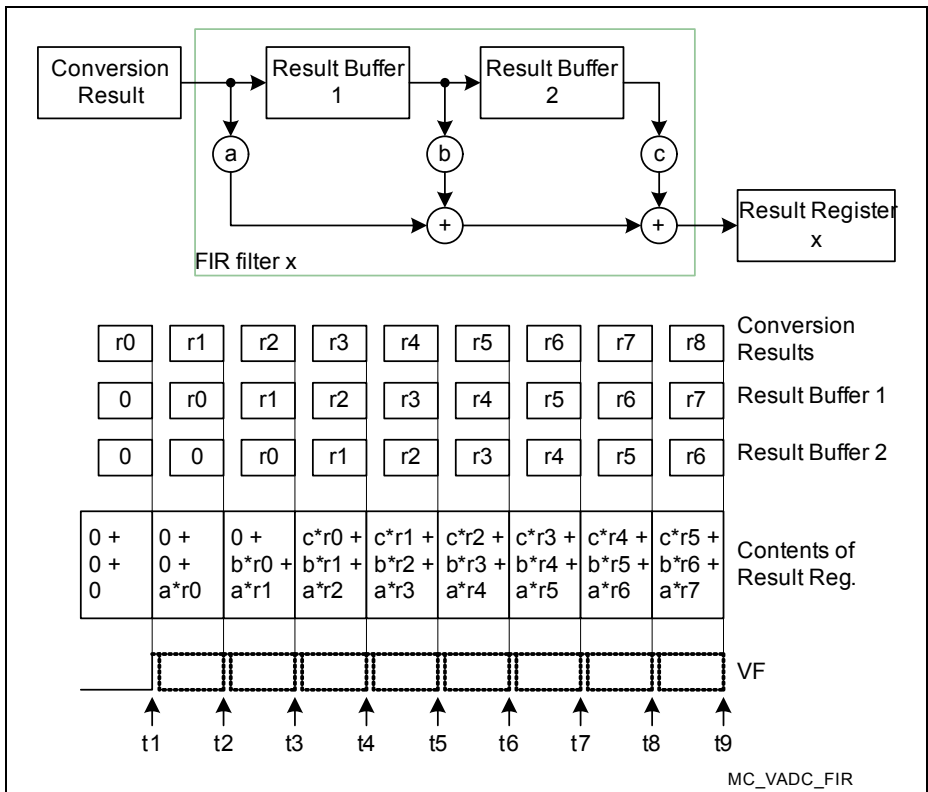


Figure 29-23 FIR Filter Structure

Note: The filter registers are cleared while bitfield DMM ≠ 01_B.

Versatile Analog-to-Digital Converter (VADC)

Infinite Impulse Response Filter Mode (IIR)

The IIR filter (see [Figure 29-24](#)) provides a result buffer (RB) and 2 configurable coefficients (a, b). It represents a first order low-pass filter.

The conversion result, weighted with the respective coefficient, and a fraction of the previous result are added to form the final value for the result register. Several predefined sets of coefficients can be selected via bitfield DRCTR (coding listed in [Table 29-6](#)) in registers **G0RCRy** ($y = 0 - 15$)ff and **GLOBRCR**. These coefficients lead to a gain of 4 to the ADC result producing a 14-bit value. The valid flag (VF) is activated for each sample after activation, i.e. for each sample generates a valid result.

Note: Conversions for other result registers can be inserted between conversions to be filtered.

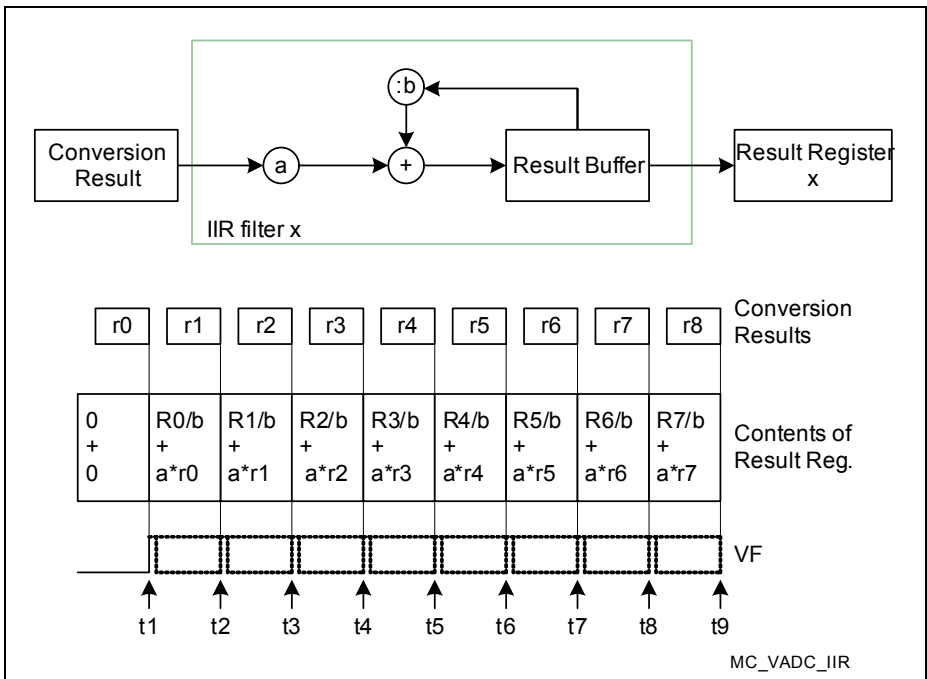


Figure 29-24 IIR Filter Structure

Note: The filter registers are cleared while bitfield DMM ≠ 01_B.

Versatile Analog-to-Digital Converter (VADC)

Difference Mode

Subtracting the contents of result register 0 from the actual result puts the results of the respective channel in relation to another signal. No software action is required.

The reference channel must store its result(s) into result register 0. The reference value can be determined once and then be used for a series of conversions, or it can be converted before each related conversion.

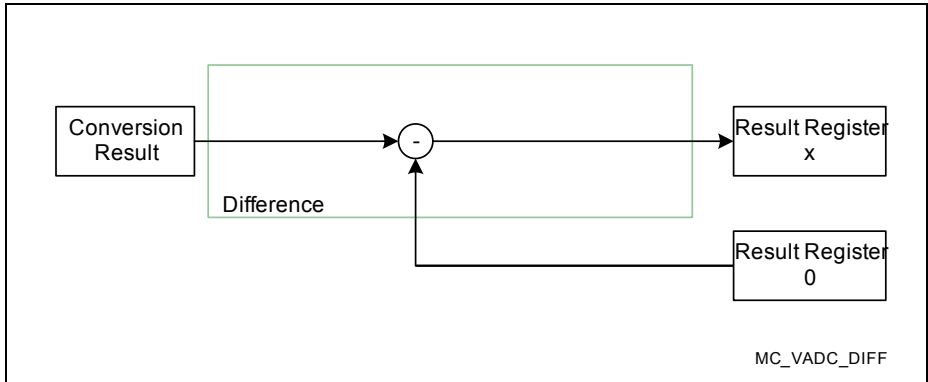


Figure 29-25 Result Difference

Versatile Analog-to-Digital Converter (VADC)

29.8 Synchronization of Conversions

The conversions of an ADC kernel can be scheduled either self-timed according to the kernel's configuration or triggered by external (outside the ADC) signals:

Synchronized Conversions for Parallel Sampling support parallel conversion of channels within a synchronization group¹⁾. This optimizes e.g. the control of electrical drives.

Equidistant Sampling supports conversions in a fixed raster with minimum jitter. This optimizes e.g. filter algorithms or audio applications.

29.8.1 Synchronized Conversions for Parallel Sampling

Several independent ADC kernels¹⁾ implemented in the TC26x B-Step can be synchronized for simultaneous measurements of analog input channels. While no parallel conversion is requested, the kernels can work independently.

The synchronization mechanism for parallel conversions ensures that the sample phases of the related channels start simultaneously. Synchronized kernels convert the same channel that is requested by the master. Different values for the resolution and the sample phase length of each kernel for a parallel conversion are supported.

A parallel conversion can be requested individually for each input channel (one or more). In the example shown in **Figure 29-26**, input channels CH3 of the ADC kernels 0, 1, 2 are converted synchronously, whereas other input channels do not lead to parallel conversions.

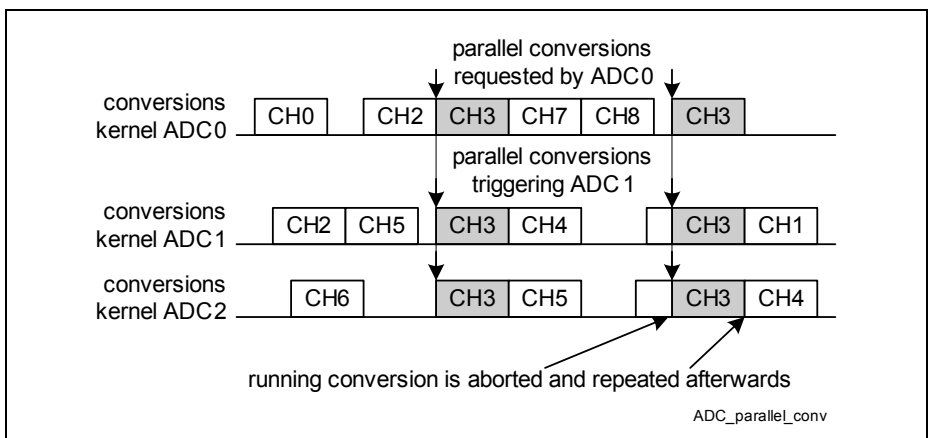


Figure 29-26 Parallel Conversions

1) For a summary, refer to **“Synchronization Groups in the TC26x B-Step”** on **Page 29-148**.

Versatile Analog-to-Digital Converter (VADC)

One kernel operates as synchronization master, the other kernel(s) operate(s) as synchronization slave. Each kernel can play either role. The arbiters of all involved kernels must run synchronously. This is achieved by switching off all involved kernels before the initialization and switching on the master kernel at the end of the initialization sequence.

Master and slave kernels form a “conversion group” to control parallel sampling:

- The arbiters must run permanently (bits **GxARBCFG (x = 0 - 3).ARBM = 0**). Initialize the slave before the master to have the arbiters run synchronously. Set the master’s **GxARBCFG.ANONC** at the end of the initialization.
- **The synchronization master** controls the slave(s) by providing the control information **GxARBCFG (x = 0 - 3).ANONS** (see **Figure 29-27**) and the requested channel number.
 - Bitfield **GxSYNCTR (x = 0 - 3).STSEL = 00_B** selects the master’s ANON information as the source of the ANON information for all kernels of the synchronization group.¹⁾
 - The ready signals indicate when a slave kernel is ready to start the sample phase of a parallel conversion.
 - Bit **GxSYNCTR (x = 0 - 3).EVALRy = 1** enables the control by the ready signal (in the example, kernels 1 and 2 are slaves, so **EVALR1 = EVALR2 = 1**).
 - The master requests a synchronized conversion of a certain channel (**SYNC = 1** in the corresponding channel control register **GxCHCTRy**), which is also requested in the connected slave ADC kernel(s).
 - Wait-for-read mode is supported for the master.
- **The synchronization slave** reacts to incoming synchronized conversion requests from the master. While no synchronized conversions are requested, the slave kernel can execute “local” conversions.
 - Bitfield **GxSYNCTR (x = 0 - 3).STSEL = 01_B/10_B/11_B** selects the master’s ANON information as the source of the ANON information for all kernels of the synchronization group¹⁾ (in the example, kernel 0 is the master, so **STSEL = 01_B**).
 - The ready signals indicate when the master kernel and the other slave kernels are ready to start the sample phase of a parallel conversion.
 - Bit **GxSYNCTR (x = 0 - 3).EVALRy = 1** enables the control by the ready signal (in the example, kernel 0 is the master, so **EVALR1 = 1**, kernel 1 and 2 are slaves, so **EVALR2 = 1**).
 - The slave timing must be configured according to the master timing (**ARBRND** in register **GxARBCFG (x = 0 - 3)**) to enable parallel conversions.
 - A parallel conversion request is always handled with highest priority and cancel-inject-repeat mode.

1) **STSEL = 00_B** selects the own ANON information. The other control inputs (**STSEL = 01_B/10_B/11_B**) are connected to the other kernels of a synchronization group in ascending order (see also **Table 29-9 “Synchronization Groups in the TC26x B-Step” on Page 29-148**).

Versatile Analog-to-Digital Converter (VADC)

- Wait-for-read mode is ignored in the slave. Previous results may be overwritten, in particular, if the same result register is used by other conversions.
- Once started, a parallel conversion cannot be aborted.

Note: Synchronized conversions request the same channel number, defined by the master. Using the alias feature (see [Section 29.6.2](#)), analog signals from different input channels can be converted. This is advantageous if e.g. CH0 is used as alternate reference.

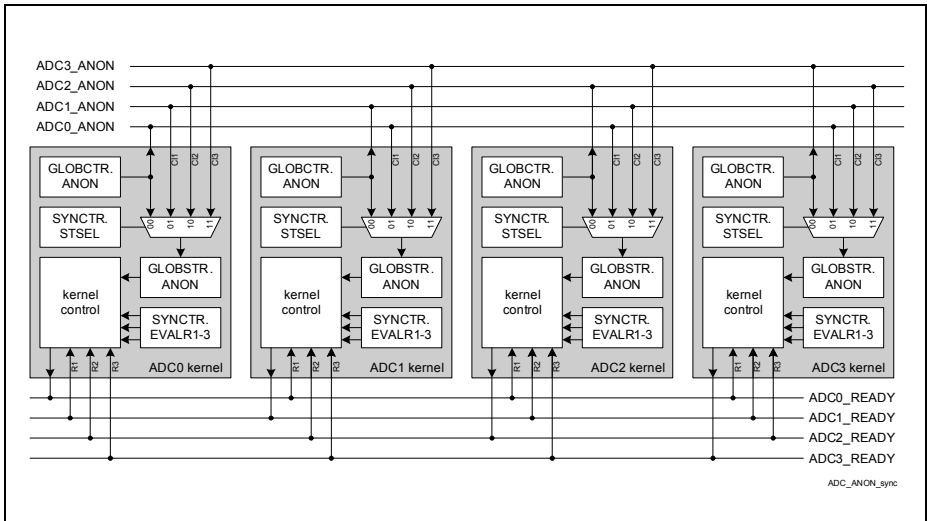


Figure 29-27 Synchronization via ANON and Ready Signals

Versatile Analog-to-Digital Converter (VADC)

The Synchronization Control Register controls the synchronization of kernels for parallel conversions.

Note: Program register GxSYNCTR only while bitfield GxARBCFG.ANONS = 00_B in all ADC kernels of the conversion group. Set the master's bitfield ANONC to 11_B afterwards.

GxSYNCTR (x = 0 - 3)
Synchronization Control Register, Group x
(x * 0400_H + 04C0_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	EVA LR3	EVA LR2	EVA LR1	0	0	STSEL	
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
STSEL	[1:0]	rw	Start Selection Controls the synchronization mechanism of the ADC kernel. 00 _B Kernel is synchronization master: Use own bitfield GxARBCFG.ANONC 01 _B Kernel is synchronization slave: Control information from input CI1 10 _B Kernel is synchronization slave: Control information from input CI2 11 _B Kernel is synchronization slave: Control information from input CI3 <i>Note: Control inputs CIx see Figure 29-27, connected kernels see Table 29-9.</i>
0	[3:2]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
EVALR1, EVALR2, EVALR3	4, 5, 6	rw	Evaluate Ready Input Rx¹⁾ Enables the ready input signal for a kernel of a conversion group. 0_B No ready input control 1_B Ready input Rx is considered for the start of a parallel conversion of this conversion group
0	[31:7]	r	Reserved, write 0, read as 0

1) Make sure to enable the ready inputs from the master and all selected slaves in the master and in all selected slaves.

29.8.2 Equidistant Sampling

To optimize the input data e.g. for filter or audio applications, conversions can be executed in a fixed timing raster. Conversions for equidistant sampling are triggered by an external signal (e.g. a timer). To generate the trigger signal synchronous to the arbiter, the ADC provides an output signal (ARBCNT) that is activated once per arbitration round and serves as timing base for the trigger timer. In this case, the arbiter must run permanently (**GxARBCFG (x = 0 - 3).ARBM = 0**). If the timer has an independent time base, the arbiter can be stopped while no requests are pending. The preface time (see **Figure 29-28**) must be longer than one arbitration round and the highest possible conversion time.

Select timer mode (TMEN = 1 in register **GxQCTRL0 (x = 0 - 3)** or **GxASCTRL (x = 0 - 3)**) for the intended source of equidistant conversions. In timer mode, a request of this source is triggered and arbitrated, but only started when the trigger signal is removed (see **Figure 29-28**) and the converter is idle.

To ensure that the converter is idle and the start of conversion can be controlled by the trigger signal, the equidistant conversion requests must receive highest priority. The preface time between request trigger and conversion start must be long enough for a currently active conversion to finish.

The frequency of signal REQTRx defines the sampling rate and its high time defines the preface time interval where the corresponding request source takes part in the arbitration.

Depending on the used request source, equidistant sampling is also supported for a sequence of channels. It is also possible to do equidistant sampling for more than one request source in parallel if the preface times and the equidistant conversions do not overlap.

Versatile Analog-to-Digital Converter (VADC)

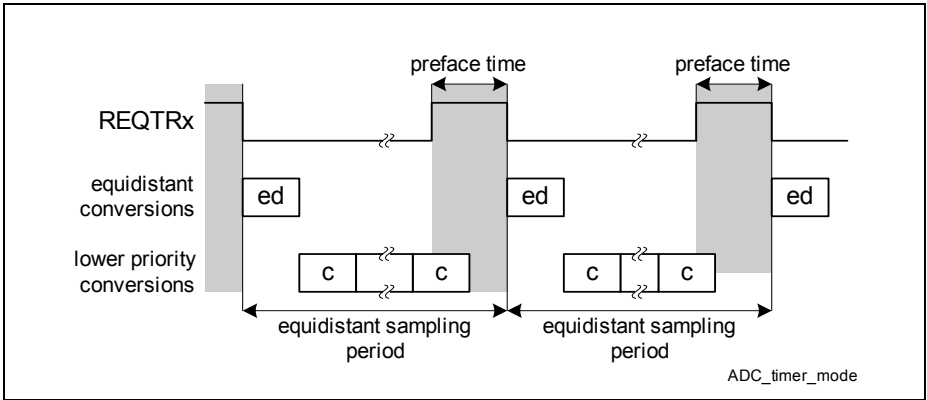


Figure 29-28 Timer Mode for Equidistant Sampling

29.9 Safety Features

Several test features can be enabled to verify the validity of the analog input signals of an application. These test features aim at different sections of the signal flow:

- **Broken Wire Detection** validates the connection from the sensor to the input pin,
- **Multiplexer Diagnostics** validates the operation of the internal analog input multiplexer,
- **Converter Diagnostics** validates the operation of the Analog/Digital converter itself.

29.9.1 Broken Wire Detection

To test the proper connection of an external analog sensor to its input pin, the converter's capacitor can be precharged to a selectable value before the regular sample phase. If the connection to the sensor is interrupted the subsequent conversion value will rather represent the precharged value than the expected sensor result. By using a precharge voltage outside the expected result range (broken wire detection preferably uses V_{AGND} and/or V_{AREF}) a valid measurement (sensor connected) can be distinguished from a failure (sensor detached).

While broken wire detection is disabled, the converter's capacitor is precharged to $V_{AREF}/2$.

Note: The duration of the complete conversion is increased by the preparation phase (same as the sample phase) if the broken wire detection is enabled. This influences the timing of conversion sequences.

Broken wire detection can be enabled for each channel separately by bitfield BWDEN in the corresponding channel control register (GxCHCTRY). Bitfield BWDCH selects the level for the preparation phase.

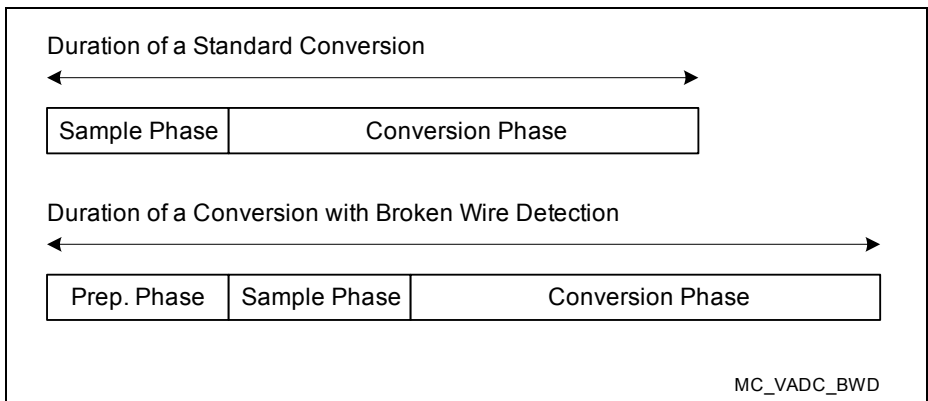


Figure 29-29 Broken Wire Detection

Versatile Analog-to-Digital Converter (VADC)

29.9.2 Signal Path Test Modes

Additional test structures can be activated to test the signal path from the sensor to the input pin and the internal signal path from the input pin through the multiplexer to the converter. These test structures apply additional loads to the signal path (see summary in [Figure 29-30](#)).

Multiplexer Diagnostics

To test the proper operation of the internal analog input multiplexer, additional pull-up and/or pull-down devices can be connected to channels CH1 and CH2. In combination with a known external input signal this test function shows if the multiplexer connects any pin to the converter input and if this is the correct pin.

Pull-Down Diagnostics

One single input channel provides a further strong pull-down (R_{PDD}) that can be activated to verify the external connection to a sensor.

Converter Diagnostics

To test the proper operation of the converter itself, several signals can be connected to the converter input. The test signals can be connected to the converter input either instead of the standard input signal or in parallel to the standard input signal.

To use the converter diagnostic without a connected input, select a non-existent channel number via the alias feature ([Section 29.6.2](#)). Redirect channel 0 or 1 to channel 24 (not connected), the conversion result can then be read from the selected result register for channel 0 or 1.

The test signal can be selected from four different signals as shown in [Figure 29-30](#).

Attention: *If the Multiplexer Diagnostics are used on channels that are overlaid on IO ports, enable these functions by setting the corresponding control bits in registers Pxx_PCSR.*

The corresponding control bits in registers Pxx_PDISC must be cleared for both Multiplexer Diagnostics and Pull-Down Diagnostics.

Versatile Analog-to-Digital Converter (VADC)

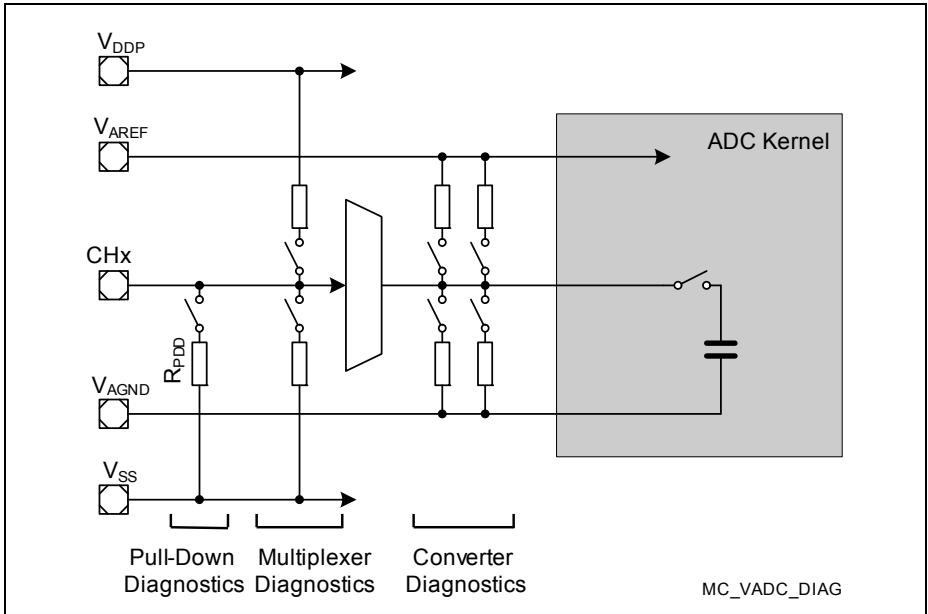


Figure 29-30 Signal Path Test

Note: When internal pull devices are activated, they may need some time to pull the input signal to the intended level, depending on the connected external circuitry. This can be accomplished by increasing the sample time for those conversions.

Versatile Analog-to-Digital Converter (VADC)
29.9.3 Configuration of Test Functions

The pull-up and pull-down devices for the test functions can be enabled individually under software control. Various test levels can be applied controlling the devices in an adequate way. Because these test functions interfere with the normal operation of the A/D Converters, they are controlled by a separate register set.

Not all test options are available for each channel. Selecting an unavailable function has no effect.

GLOBTF
Global Test Functions Register (0160_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	MD WC	0	0	0	0	MD PU	MD PD	PDD
r	r	r	r	r	r	r	r	w	r	r	r	r	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CD WC	0	0	0	0	CD SEL	CD EN	CDGR					CDCH			
w	r	r	r	r	rw	rw	rw					rw			

Field	Bits	Type	Description
CDCH	[3:0]	rw	Control Diagnostics Channel Defines the channel number to be used for diagnostic conversions. Applies to MDPU, MDPD.
CDGR	[7:4]	rw	Control Diagnostics Group Defines the group number to be used for diagnostic conversions. Applies to all test functions.
CDEN	8	rw	Converter Diagnostics Enable 0 _B All diagnostic pull devices are disconnected 1 _B Diagnostic pull devices connected as selected by bitfield CDSEL
CDSEL	[10:9]	rw	Converter Diagnostics Pull-Devices Select 00 _B Connected to VAREF 01 _B Connected to VAGND 10 _B Connected to 1/3rd VAREF 11 _B Connected to 2/3rd VAREF

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
0	[14:11]	r	Reserved, write 0, read as 0
CDWC	15	w	Write Control for Conversion Diagnostics 0_B No write access to parameters 1_B Bitfields CDSEL, CDEN, CDGR, CDCH can be written
PDD	16	rw	Pull-Down Diagnostics Enable 0_B Disconnected 1_B The pull-down diagnostics device is active <i>Note: Channels with pull-down diagnostics device are marked in Table 29-12.</i>
MDPD, MDPU	17, 18	rw	Multiplexer Diagnostics Pull-Devices Enable 0_B Disconnected 1_B The respective device is active Connecting combinations of pull-up and/or pull-down devices generate various loads for testing. <i>Note: Channels with pull-down diagnostics device are marked in Table 29-12.</i>
0	[22:19]	r	Reserved, write 0, read as 0
MDWC	23	w	Write Control for Multiplexer Diagnostics 0_B No write access to parameters 1_B Bitfields MDPD, MDPU, PDD can be written
0	[31:24]	r	Reserved, write 0, read as 0

Note: All diagnostic functions are activated for the selected group (CDGR). Converter diagnostics (CDEN, CDSEL) and pull-down diagnostics (PDD) are activated directly by setting the associated bits, multiplexer diagnostics (MDPD, MDPU) are activated for the selected channel (CDCH).

29.10 External Multiplexer Control

The number of analog input channels can be increased by connecting external analog multiplexers to an input MUX channel. The ADC can be configured to control these external multiplexers automatically.

For each available EMUX interface (see register **EMUXSEL**) one MUX channel or a set of MUX channels from one group can be selected for this operating mode. The ADC supports 1-out-of-8 multiplexers with several control options:

- **Sequence mode** automatically converts all configured external channels when the selected MUX channel is encountered. In the example in **Figure 29-31** the following conversions are done: --4-32-31-30-2-1-0--4-32-31-30-2-1-0--...
- **Single-step mode** converts one external channel of the configured sequence when the selected MUX channel is encountered. In the example in **Figure 29-31** the following conversions are done: --4-32-2-1-0--4-31-2-1-0--4-30-2-1-0--4-32-... (Single-step mode works best with one channel)
- **Steady mode** converts the configured external channel when the selected MUX channel is encountered. In the example in **Figure 29-31** the following conversions are done: --4-32-2-1-0--4-32-2-1-0--4-32-2-1-0--...

*Note: The example in **Figure 29-31** has an external multiplexer connected to MUX channel CH3. The start selection value **EMUXSET** is assumed as 2.*

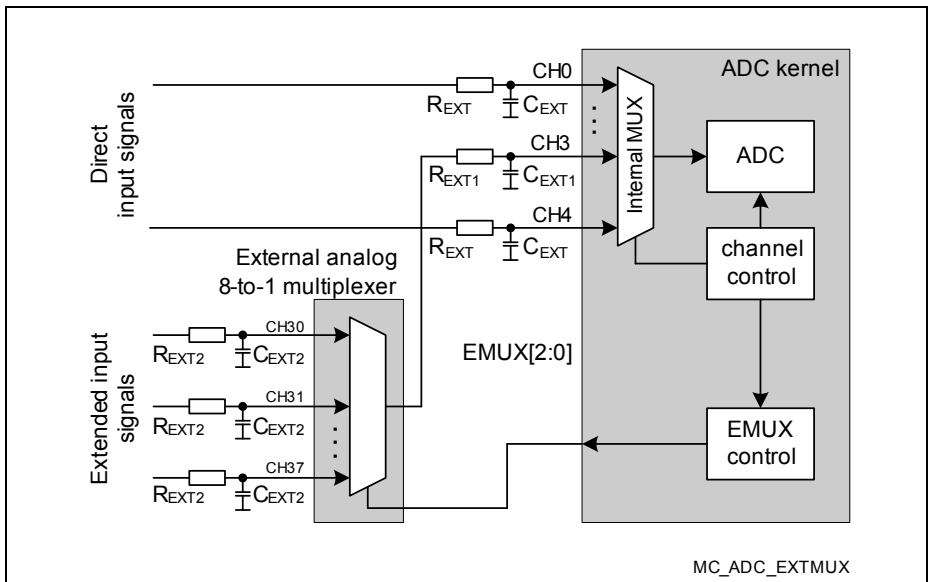


Figure 29-31 External Analog Multiplexer Example

Versatile Analog-to-Digital Converter (VADC)

Bitfield EMUXACT determines the control information sent to the external multiplexer. In single-step mode, EMUXACT is updated after each conversion of an enabled MUX channel. If EMUXACT = 000_B it is reloaded from bitfield EMUXSET, otherwise it is decremented by 1.

Additional external channels may have different properties due to the modified signal path. Local filters may be used at the additional inputs (R_{EXT2} - C_{EXT2} on CH3x in **Figure 29-31**). For applications where the external multiplexer is located far from the ADC analog input, it is recommended to add an RC filter directly at the analog input of the ADC (R_{EXT1} - C_{EXT1} on CH3 in **Figure 29-31**).

Note: Each RC filter limits the bandwidth of the analog input signal.

Conversions for external channels, therefore, use the alternate conversion mode setting CME. This automatically selects a different conversion mode if required.

Switching the external multiplexer usually requires an additional settling time for the input signal. Therefore, the alternate sample time setting STCE is applied each time the external channel is changed. This automatically fulfills the different sampling time requirements in this case.

Control Signals

The external channel number that controls the external multiplexer can be output in standard binary format or Gray-coded. Gray code avoids intermediate multiplexer switching when selecting a sequence of channels, because only one bit changes at a time. **Table 29-7** indicates the resulting codes.

Table 29-7 EMUX Control Signal Coding

Channel	0	1	2	3	4	5	6	7
Binary	000 _B	001 _B	010 _B	011 _B	100 _B	101 _B	110 _B	111 _B
Gray	000	001	011	010	110	111	101	100

Operation Without External Multiplexer

If no external multiplexers are used in an application, the reset values of the control registers provide the appropriate setup.

EMUXMODE = 00_B disables the automatic EMUX control.

Since the control output signals are alternate port output signals, they are only visible at the respective pins if explicitly selected.

Versatile Analog-to-Digital Converter (VADC)

In each group an arbitrary MUX channel or set of MUX channels can be assigned to external multiplexer control (register **GxEMUXCTR** ($x = 0 - 3$)). Each available port interface selects the group whose control lines are output (register **EMUXSEL**).

GxEMUXCTR ($x = 0 - 3$)

External Multiplexer Control Register, Group x
 $(x * 0400_H + 05F0_H)$
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EMX WC	EMX CSS	EMX ST	EMX COD	EMUX MODE	EMUX CH										
w	rw	rw	rw	rw	rw										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	EMUX ACT	0	0	0	0	0	EMUX SET				
r	r	r	r	r	rh	r	r	r	r	r	rw				

Field	Bits	Type	Description
EMUXSET	[2:0]	rw	External Multiplexer Start Selection¹⁾ Defines the initial setting for the external multiplexer.
0	[7:3]	r	Reserved, write 0, read as 0
EMUXACT	[10:8]	rh	External Multiplexer Actual Selection Defines the current value for the external multiplexer selection. This bitfield is loaded from bitfield EMUXSET and modified according to the operating mode selected by bitfield EMUXMODE.
0	[15:11]	r	Reserved, write 0, read as 0
EMUXCH	[25:16]	rw	External Multiplexer Channel Select Defines the channel(s) to which the external multiplexer control is applied. EMXCSS = 0: Channel number , the lower 5 bits select an arbitrary channel (valid numbers are limited by the number of available channels, unused bits shall be 0) EMXCSS = 1: Channel enable , each bit enables the associated channel (multiple channels can be selected/enabled)

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
EMUXMODE	[27:26]	rw	External Multiplexer Mode 00 _B Software control (no hardware action) 01 _B Steady mode (use EMUXSET value) 10 _B Single-step mode ¹⁾²⁾ 11 _B Sequence mode ¹⁾
EMXCOD	28	rw	External Multiplexer Coding Scheme 0 _B Output the channel number in binary code 1 _B Output the channel number in Gray code
EMXST	29	rw	External Multiplexer Sample Time Control 0 _B Use STCE whenever the setting changes 1 _B Use STCE for each conversion of an external channel
EMXCSS	30	r	External Multiplexer Channel Selection Style 0 _B Channel number: Bitfield EMUXCH selects an arbitrary channel 1 _B Channel enable: Each bit of bitfield EMUXCH selects the associated channel for EMUX control
EMXWC	31	w	Write Control for EMUX Configuration 0 _B No write access to EMUX cfg. 1 _B Bitfields EMXMODE, EMXCOD, EMXST, EMXCSS can be written

1) For single-step mode and sequence mode: Select the start value before selecting the respective mode.

2) Single-step mode modifies the EMUX channel number each time an EMUX-enabled channel is converted. Therefore, single-step mode works best with a single channel, because otherwise some external channels may be skipped.

Versatile Analog-to-Digital Converter (VADC)

Register EMUXSEL is a global register which assigns an arbitrary group to each of the EMUX interfaces.

EMUXSEL

External Multiplexer Select Register

(03F0_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	EMUX GRP1				EMUX GRP0			
r	r	r	r	r	r	r	r	rw				rw			

Field	Bits	Type	Description
EMUXGRP0, EMUXGRP1	[3:0], [7:4]	rw	External Multiplexer Group for Interface x Defines the group whose external multiplexer control signals are routed to EMUX interface x. ¹⁾
0	[31:8]	r	Reserved, write 0, read as 0

1) The pins that are associated with each EMUX interface are listed in [Table 29-13 “Digital Connections in the TC26x B-Step”](#) on [Page 29-157](#).

29.11 Service Request Generation

Each A/D Converter can activate up to 4 group-specific service request output signals and up to 4 shared service request output signals to issue an interrupt or to trigger a DMA channel. Two common service request groups are available, see [“Product-Specific Configuration” on Page 29-147](#).

Several events can be assigned to each service request output. Service requests can be generated by three types of events:

- **Request source events:** indicate that a request source completed the requested conversion sequence and the application software can initiate further actions.
For a scan source (group or background), the event is generated when the complete defined set of channels (pending bits) has been converted.
For a group queue source, the event is generated according to the programming, i.e. when a channel with enabled source interrupt has been converted or when an invalid entry is encountered.
- **Channel events:** indicate that a conversion is finished. Optionally, channel events can be restricted to result values within a programmable value range. This offloads the CPU/DMA from background tasks, i.e. a service request is only activated if the specified conversion result range is met or exceeded.
- **Result events:** indicate a new valid result in a result register. Usually, this triggers a read action by the CPU (or DMA). Optionally, result events can be generated only at a reduced rate if data reduction is active.
For example, a single DMA channel can read the results for a complete auto-scan sequence, if all channels of the sequence target the same result register and the transfers are triggered by result events.

Each ADC event is indicated by a dedicated flag that can be cleared by software. If a service request is enabled for a certain event, the service request is generated for each event, independent of the status of the corresponding event indication flag. This ensures efficient DMA handling of ADC events (the ADC event can generate a service request without the need to clear the indication flag).

Event flag registers indicate all types of events that occur during the ADC's operation. Software can set each flag by writing a 1 to the respective position in register GxCEFLAG/GxRFLAG to trigger an event. Software can clear each flag by writing a 1 to the respective position in register GxCEFCLR/GxREFCLR. If enabled, service requests are generated for each occurrence of an event, even if the associated flag remains set.

Versatile Analog-to-Digital Converter (VADC)
GxSEFLAG (x = 0 - 3)
Source Event Flag Register, Group x

$$(x * 0400_H + 0588_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	SEV 1	SEV 0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rwh	rwh

Field	Bits	Type	Description
SEV0, SEV1	0, 1	rwh	Source Event 0/1 0 _B No source event 1 _B A source event has occurred
0	[31:2]	r	Reserved, write 0, read as 0

Note: Software can set all flags in register GxSEFLAG and trigger the corresponding event by writing 1 to the respective bit. Writing 0 has no effect.

Software can clear all flags in register GxSEFLAG by writing 1 to the respective bit in register GxSEFCLR.

GxCEFLAG (x = 0 - 3)
Channel Event Flag Register, Group x

$$(x * 0400_H + 0580_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CEV 15	CEV 14	CEV 13	CEV 12	CEV 11	CEV 10	CEV 9	CEV 8	CEV 7	CEV 6	CEV 5	CEV 4	CEV 3	CEV 2	CEV 1	CEV 0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
CEVy (y = 0 - 15)	y	rwh	Channel Event for Channel y 0 _B No channel event 1 _B A channel event has occurred
0	[31:16]	r	Reserved, write 0, read as 0

*Note: Software can set all flags in register GxCEFLAG and trigger the corresponding event by writing 1 to the respective bit. Writing 0 has no effect.
Software can clear all flags in register GxCEFLAG by writing 1 to the respective bit in register GxCEFCLR.*

GxREFLAG (x = 0 - 3)
Result Event Flag Register, Group x

$$(x * 0400_H + 0584_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV	REV	REV	REV	REV	REV	REV	REV	REV	REV	REV	REV	REV	REV	REV	REV
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
REVy (y = 0 - 15)	y	rwh	Result Event for Result Register y 0 _B No result event 1 _B New result was stored in register GxRESy
0	[31:16]	r	Reserved, write 0, read as 0

*Note: Software can set all flags in register GxREFLAG and trigger the corresponding event by writing 1 to the respective bit. Writing 0 has no effect.
Software can clear all flags in register GxREFLAG by writing 1 to the respective bit in register GxREFCLR.*

Versatile Analog-to-Digital Converter (VADC)
GxSEFCLR (x = 0 - 3)
Source Event Flag Clear Register, Group x

$$(x * 0400_H + 0598_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	SEV 1	SEV 0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	w	w

Field	Bits	Type	Description
SEV0, SEV1	0, 1	w	Clear Source Event 0/1 0 _B No action 1 _B Clear the source event flag in GxSEFLAG
0	[31:2]	r	Reserved, write 0, read as 0

GxCEFCLR (x = 0 - 3)
Channel Event Flag Clear Register, Group x

$$(x * 0400_H + 0590_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CEV 15	CEV 14	CEV 13	CEV 12	CEV 11	CEV 10	CEV 9	CEV 8	CEV 7	CEV 6	CEV 5	CEV 4	CEV 3	CEV 2	CEV 1	CEV 0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
CEVy (y = 0 - 15)	y	w	Clear Channel Event for Channel y 0 _B No action 1 _B Clear the channel event flag in GxCEFLAG

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
0	[31:16]	r	Reserved, write 0, read as 0

GxREFCLR (x = 0 - 3)
Result Event Flag Clear Register, Group x

$$(x * 0400_H + 0594_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV 15	REV 14	REV 13	REV 12	REV 11	REV 10	REV 9	REV 8	REV 7	REV 6	REV 5	REV 4	REV 3	REV 2	REV 1	REV 0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
REVy (y = 0 - 15)	y	w	Clear Result Event for Result Register y 0_B No action 1_B Clear the result event flag in GxREFLAG
0	[31:16]	r	Reserved, write 0, read as 0

GLOBEFLAG
Global Event Flag Register

$$(00E0_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	REV GLB CLR	0	0	0	0	0	0	0	SEV GLB CLR
r	r	r	r	r	r	r	w	r	r	r	r	r	r	r	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	REV GLB	0	0	0	0	0	0	0	SEV GLB
r	r	r	r	r	r	r	rwh	r	r	r	r	r	r	r	rwh

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
SEVGLB	0	rwh	Source Event (Background) 0 _B No source event 1 _B A source event has occurred
0	[7:1]	r	Reserved, write 0, read as 0
REVGLB	8	rwh	Global Result Event 0 _B No result event 1 _B New result was stored in register GLOBRES
0	[15:9]	r	Reserved, write 0, read as 0
SEVGLBCLR	16	w	Clear Source Event (Background) 0 _B No action 1 _B Clear the source event flag SEVGLB
0	[23:17]	r	Reserved, write 0, read as 0
REVGLBCLR	24	w	Clear Global Result Event 0 _B No action 1 _B Clear the result event flag REVGLB
0	[31:25]	r	Reserved, write 0, read as 0

Note: Software can set flags REVGLB and SEVGLB and trigger the corresponding event by writing 1 to the respective bit. Writing 0 has no effect.

Software can clear these flags by writing 1 to bit REVGLBCLR and SEVGLBCLR, respectively.

Setting both bits (e.g. SEVGLB and SEVGLBCLR) simultaneously clears the corresponding flag (e.g. SEVGLB).

Node Pointer Registers

Requests from each event source can be directed to a set of service request nodes via associated node pointers. Requests from several sources can be directed to the same node; in this case, they are ORed to the service request output signal.

Versatile Analog-to-Digital Converter (VADC)

GxSEVNP (x = 0 - 3)

Source Event Node Pointer Register, Group x

$$(x * 0400_H + 05C0_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	SEV1NP				SEV0NP			
r	r	r	r	r	r	r	r	rw				rw			

Field	Bits	Type	Description
SEV0NP, SEV1NP	[3:0], [7:4]	rw	<p>Service Request Node Pointer Source Event i¹⁾ Routes the corresponding event trigger to one of the service request lines (nodes). 0000_BSelect service request line 0 of group x ... 0011_BSelect service request line 3 of group x 0100_BSelect shared service request line 0 ... 0111_BSelect shared service request line 3 1xxx_BReserved <i>Note: For shared service request lines see common groups in Section 29.12.1.</i></p>
0	[31:8]	r	Reserved, write 0, read as 0

1) Source 0 is an 8-stage queued source, source 1 is a channel scan source.

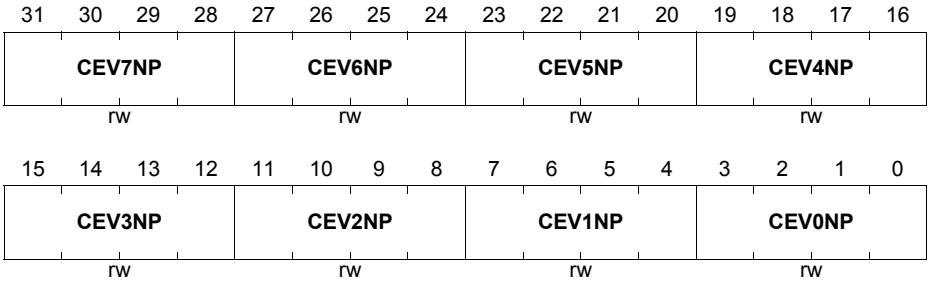
Versatile Analog-to-Digital Converter (VADC)

GxCEVNP0 (x = 0 - 3)

Channel Event Node Pointer Register 0, Group x

(x * 0400_H + 05A0_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CEV0NP, CEV1NP, CEV2NP, CEV3NP, CEV4NP, CEV5NP, CEV6NP, CEV7NP	[3:0], [7:4], [11:8], [15:12], [19:16], [23:20], [27:24], [31:28]	rw	<p>Service Request Node Pointer Channel Event i</p> <p>Routes the corresponding event trigger to one of the service request lines (nodes).</p> <p>0000_BSelect service request line 0 of group x</p> <p>...</p> <p>0011_BSelect service request line 3 of group x</p> <p>0100_BSelect shared service request line 0</p> <p>...</p> <p>0111_BSelect shared service request line 3</p> <p>1xxx_BReserved</p> <p><i>Note: For shared service request lines see common groups in Section 29.12.1.</i></p>

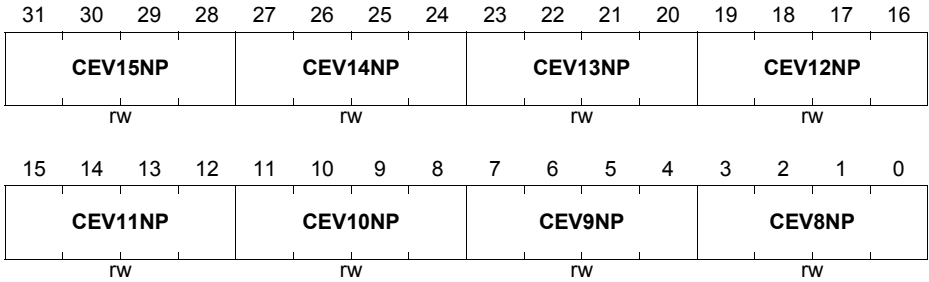
Versatile Analog-to-Digital Converter (VADC)

GxCEVNP1 (x = 0 - 3)

Channel Event Node Pointer Register 1, Group x

(x * 0400_H + 05A4_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CEV8NP, CEV9NP, CEV10NP, CEV11NP, CEV12NP, CEV13NP, CEV14NP, CEV15NP	[3:0], [7:4], [11:8], [15:12], [19:16], [23:20], [27:24], [31:28]	rw	Service Request Node Pointer Channel Event i Routes the corresponding event trigger to one of the service request lines (nodes). 0000 _B Select service request line 0 of group x ... 0011 _B Select service request line 3 of group x 0100 _B Select shared service request line 0 ... 0111 _B Select shared service request line 3 1xxx _B Reserved <i>Note: For shared service request lines see common groups in Section 29.12.1.</i>

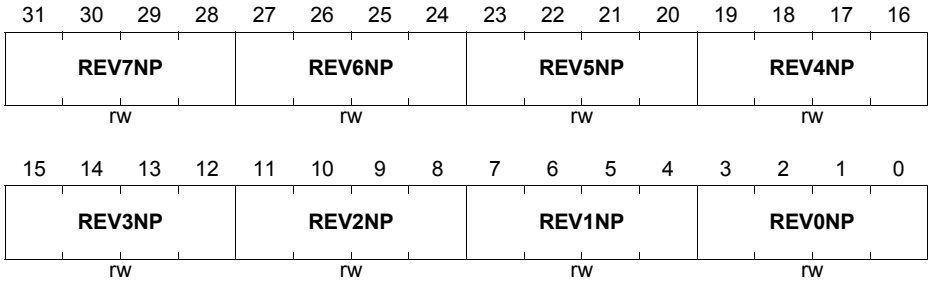
Versatile Analog-to-Digital Converter (VADC)

GxREVNP0 (x = 0 - 3)

Result Event Node Pointer Register 0, Group x

(x * 0400_H + 05B0_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
REV0NP, REV1NP, REV2NP, REV3NP, REV4NP, REV5NP, REV6NP, REV7NP	[3:0], [7:4], [11:8], [15:12], [19:16], [23:20], [27:24], [31:28]	rw	<p>Service Request Node Pointer Result Event i</p> <p>Routes the corresponding event trigger to one of the service request lines (nodes).</p> <p>0000_BSelect service request line 0 of group x</p> <p>...</p> <p>0011_BSelect service request line 3 of group x</p> <p>0100_BSelect shared service request line 0</p> <p>...</p> <p>0111_BSelect shared service request line 3</p> <p>1xxx_BReserved</p> <p><i>Note: For shared service request lines see common groups in Section 29.12.1.</i></p>

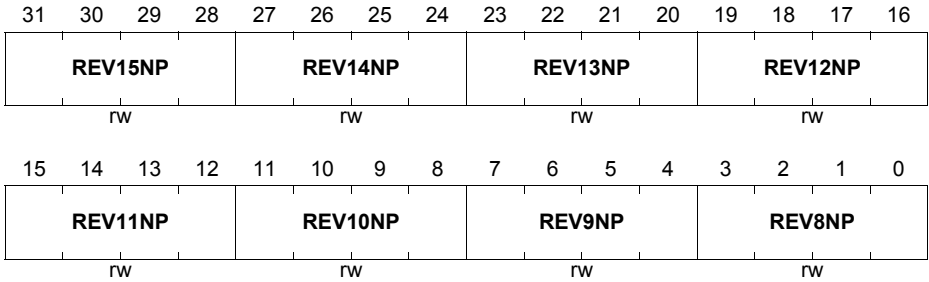
Versatile Analog-to-Digital Converter (VADC)

GxREVNP1 (x = 0 - 3)

Result Event Node Pointer Register 1, Group x

(x * 0400_H + 05B4_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
REV8NP, REV9NP, REV10NP, REV11NP, REV12NP, REV13NP, REV14NP, REV15NP	[3:0], [7:4], [11:8], [15:12], [19:16], [23:20], [27:24], [31:28]	rw	Service Request Node Pointer Result Event i Routes the corresponding event trigger to one of the service request lines (nodes). 0000 _B Select service request line 0 of group x ... 0011 _B Select service request line 3 of group x 0100 _B Select shared service request line 0 ... 0111 _B Select shared service request line 3 1xxx _B Reserved <i>Note: For shared service request lines see common groups in Section 29.12.1.</i>

Versatile Analog-to-Digital Converter (VADC)

GLOBEVNP
Global Event Node Pointer Register

 (0140_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	REV0NP			
r	r	r	r	r	r	r	r	r	r	r	r	rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	SEV0NP			
r	r	r	r	r	r	r	r	r	r	r	r	rw			

Field	Bits	Type	Description
SEV0NP	[3:0]	rw	Service Request Node Pointer Backgr. Source Routes the corresponding event trigger to one of the service request lines (nodes). 0000 _B Select shared service request line 0 of common service request group 0 ... 0011 _B Select shared service request line 3 of common service request group 0 0100 _B Select shared service request line 0 of common service request group 1 ... 0111 _B Select shared service request line 3 of common service request group 1 1xxx _B Reserved <i>Note: For shared service request lines see common groups in Section 29.12.1.</i>
0	[15:4]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)

Field	Bits	Type	Description
REV0NP	[19:16]	rw	<p>Service Request Node Pointer Backgr. Result Routes the corresponding event trigger to one of the service request lines (nodes).</p> <p>0000_BSelect shared service request line 0 of common service request group 0</p> <p>...</p> <p>0011_BSelect shared service request line 3 of common service request group 0</p> <p>0100_BSelect shared service request line 0 of common service request group 1</p> <p>...</p> <p>0111_BSelect shared service request line 3 of common service request group 1</p> <p>1xxx_B Reserved</p> <p><i>Note: For shared service request lines see common groups in Section 29.12.1.</i></p>
0	[31:20]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)
Software Service Request Activation

Each service request can be activated via software by setting the corresponding bit in register **GxSRACT (x = 0 - 3)**. This can be used for evaluation and testing purposes.

Note: For shared service request lines see common groups in [Section 29.12.1](#).

GxSRACT (x = 0 - 3)
Service Request Software Activation Trigger, Group x

$$(x * 0400_H + 05C8_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	AS SR3	AS SR2	AS SR1	AS SR0	0	0	0	0	AG SR3	AG SR2	AG SR1	AG SR0
r	r	r	r	w	w	w	w	r	r	r	r	w	w	w	w

Field	Bits	Type	Description
AGSRy (y = 0 - 3)	y	w	Activate Group Service Request Node y 0 _B No action 1 _B Activate the associated service request line
0	[7:4]	r	Reserved, write 0, read as 0
ASSRy (y = 0 - 3)	8 + y	w	Activate Shared Service Request Node y 0 _B No action 1 _B Activate the associated service request line
0	[31:12]	r	Reserved, write 0, read as 0

Versatile Analog-to-Digital Converter (VADC)
29.12 Implementation into the TC26x B-Step

This section describes the actual implementation of the ADC module into the TC26x B-Step, i.e. the incorporation into the microcontroller system.

The following information is listed:

- [“Product-Specific Configuration” on Page 29-147](#)
- [“Summary of Registers and Locations” on Page 29-149](#)
- [“Analog Module Connections in the TC26x B-Step” on Page 29-154](#)
- [“Digital Module Connections in the TC26x B-Step” on Page 29-157](#)

29.12.1 Product-Specific Configuration

The functional description describes the features and operating modes of the A/D Converters in a general way. This section summarizes the configuration that is available in this product (TC26x B-Step).

Each converter group is equipped with a separate analog converter module and a dedicated analog input multiplexer.

Table 29-8 General Converter Configuration in the TC26x B-Step

Converter Group	Input Channels	Channels w. Altern. Reference	Common Service Req. Group	Associated Standard Reference
G0	0 ... 13	9	C0	V_{AREF1} , V_{AGND1}
G1	0 ... 13	12	C1	V_{AREF1} , V_{AGND1}
G2	0 ... 15	12	C0	V_{AREF1} , V_{AGND1}
G3	0 ... 11	12	C1	V_{AREF1} , V_{AGND1}

Versatile Analog-to-Digital Converter (VADC)
Synchronization Groups in the TC26x B-Step

The converter kernels in the TC26x B-Step can be connected to synchronization groups to achieve parallel conversion of several input channels.

Table 29-9 summarizes which kernels can be synchronized for parallel conversions.

Table 29-9 Synchronization Groups in the TC26x B-Step

ADC Kernel	Synchr. Group	Master selected by control input C1x ¹⁾			
		C10 ²⁾	C11	C12	C13
ADC00	A	ADC00	ADC01	ADC02	ADC03
ADC01	A	ADC01	ADC00	ADC02	ADC03
ADC02	A	ADC02	ADC00	ADC01	ADC03
ADC03	A	ADC03	ADC00	ADC01	ADC02

1) The control input is selected by bitfield STSEL in register **GxSYNCTR (x = 0 - 3)**.

Select the corresponding ready inputs accordingly by bits EVALRx.

2) Control input C10 always selects the own control signals of the corresponding ADC kernel. This selection is meant for the synchronization master or for stand-alone operation.

Versatile Analog-to-Digital Converter (VADC)
29.12.2 Summary of Registers and Locations

The Versatile ADC is built from a series of converter blocks that are controlled in an identical way. This makes programming versatile and scalable. The corresponding registers, therefore, have an individual offset assigned (see [Table 29-11](#)). The exact register location is obtained by adding the respective register offset to the base address (see [Table 29-10](#)) of the corresponding group.

Due to the regular group structure, several registers appear within each group. Other registers are provided for each channel. This is indicated in the register overview table by placeholders:

- $X###_H$ means: $x \times 0400_H + 0###_H$, for $x = 0 - 3$
- $###Y_H$ means: $###0_H + y \times 0004_H$, for $y = 0 - N$ (depends on register type)

Table 29-10 Registers Address Space

Module	Base Address	End Address	Note
VADC	F002 0000 _H	F002 3FFF _H	

Table 29-11 Registers Overview

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Page Num.
			Read	Write	
ID	Module Identification Register	0008 _H	U, SV	BE	29-14
CLC	Clock Control Register	0000 _H	U, SV	SV E, P	29-15
OCS	OCDS Control and Status Register	0028 _H	U, SV	SV P	29-16
ACCEN0	Access Enable Register 0	003C _H	U, SV	SV SE	29-18
KRST0	Kernel Reset Register 0	0034 _H	U, SV	SV E, P	29-19
KRST1	Kernel Reset Register 1	0030 _H	U, SV	SV E, P	29-20
KRSTCLR	Kernel Reset Clear Register	002C _H	U, SV	SV E, P	29-20
GLOBCFG	Global Configuration Register	0080 _H	U, SV	U, SV P	29-23
ACCPROT0	Access Protection Register 0	0088 _H	U, SV	SV, SE, P	29-27

Versatile Analog-to-Digital Converter (VADC)
Table 29-11 Registers Overview (cont'd)

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Page Num.
			Read	Write	
ACCPROT1	Access Protection Register 1	008C _H	U, SV	SV, SE, P	29-28
GxARBCFG	Arbitration Configuration Register	X480 _H	U, SV	U, SV P	29-68
GxARBPR	Arbitration Priority Register	X484 _H	U, SV	U, SV P	29-71
GxCHASS	Channel Assignment Register, Group x	X488 _H	U, SV	U, SV P	29-25
GxRRASS	Result Assignment Register, Group x	X48C _H	U, SV	U, SV P	29-26
GxQCTRL0	Queue 0 Source Control Register, Group x	X500 _H	U, SV	U, SV P	29-38
GxQMR0	Queue 0 Mode Register, Group x	X504 _H	U, SV	U, SV P	29-40
GxQSR0	Queue 0 Status Register, Group x	X508 _H	U, SV	U, SV P	29-42
GxQINR0	Queue 0 Input Register, Group x	X510 _H	U, SV	U, SV P	29-44
GxQ0R0	Queue 0 Register 0, Group x	X50C _H	U, SV	U, SV P	29-46
GxQBUR0	Queue 0 Backup Register, Group x	X510 _H	U, SV	U, SV P	29-48
GxASCTRL	Autoscan Source Control Register, Group x	X520 _H	U, SV	U, SV P	29-54
GxASMR	Autoscan Source Mode Register, Group x	X524 _H	U, SV	U, SV P	29-56
GxASSEL	Autoscan Source Channel Select Register, Group x	X528 _H	U, SV	U, SV P	29-58
GxASPND	Autoscan Source Pending Register, Group x	X52C _H	U, SV	U, SV P	29-58
BRCTRL	Background Request Source Control Register	0200 _H	U, SV	U, SV P	29-60

Versatile Analog-to-Digital Converter (VADC)
Table 29-11 Registers Overview (cont'd)

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Page Num.
			Read	Write	
BRSMR	Background Request Source Mode Register	0204 _H	U, SV	U, SV P	29-62
BRSELx	Background Request Source Channel Select Register, Group x	018Y _H	U, SV	U, SV P	29-64
BRSPNDx	Background Request Source Channel Pending Register, Group x	01CY _H	U, SV	U, SV P	29-65
GxCHCTRY	Channel x Control Register	X60Y _H	U, SV	U, SV P	29-75
GxICLASS0	Input Class Register 0, Group x	X4A0 _H	U, SV	U, SV P	29-78
GxICLASS1	Input Class Register 1, Group x	X4A4 _H	U, SV	U, SV P	29-78
GLOBICLASS0	Input Class Register 0, Global	00A0 _H	U, SV	U, SV P	29-78
GLOBICLASS1	Input Class Register 1, Global	00A4 _H	U, SV	U, SV P	29-78
GxALIAS	Alias Register	X4B0 _H	U, SV	U, SV P	29-81
GxBOUND	Boundary Select Register, Group x	X4B8 _H	U, SV	U, SV P	29-83
GLOBBOUND	Global Boundary Select Register	00B8 _H	U, SV	U, SV P	29-83
GxBFL	Boundary Flag Register, Group x	X4C8 _H	U, SV	U, SV P	29-89
GxBFLS	Boundary Flag Software Register, Group x	X4CC _H	U, SV	U, SV P	29-90
GxBFLC	Boundary Flag Control Register, Group x	X4D0 _H	U, SV	U, SV P	29-91
GxBFLNP	Boundary Flag Node Pointer Register, Group x	X4D4 _H	U, SV	U, SV P	29-92
GxRCRy	Group x Result Control Register y	X68Y _H	U, SV	U, SV P	29-96

Versatile Analog-to-Digital Converter (VADC)
Table 29-11 Registers Overview (cont'd)

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Page Num.
			Read	Write	
GxRESy	Group x Result Register y	X70Y _H	U, SV	U, SV P	29-98
GxRESyDy	Group x Result Register y (debug view)	X78Y _H	U, SV	U, SV P	29-100
GLOBRCR	Global Result Control Register	0280 _H	U, SV	U, SV P	29-102
GLOBRES	Global Result Register	0300 _H	U, SV	U, SV P	29-103
GLOBRESD	Global Result Register (debug view)	0380 _H	U, SV	U, SV P	29-103
GxVFR	Valid Flag Register, Group x	X5F8 _H	U, SV	U, SV P	29-105
GxSYNCTR	Synchronization Control Register	X4C0 _H	U, SV	U, SV P	29-120
GLOBTF	Global Test Function Register	0160 _H	U, SV	U, SV P	29-126
GxEMUXCTR	External Multiplexer Control Register, Group x	X5F0 _H	U, SV	U, SV P	29-130
EMUXSEL	External Multiplexer Select Register	03F0 _H	U, SV	U, SV P	29-132
GxSEFLAG	Source Event Flag Register, Group x	X588 _H	U, SV	U, SV P	29-134
GxCEFLAG	Channel Event Flag Register, Group x	X580 _H	U, SV	U, SV P	29-134
GxREFLAG	Result Event Flag Register, Group x	X584 _H	U, SV	U, SV P	29-135
GxSEFCLR	Source Event Flag Clear Register, Group x	X598 _H	U, SV	U, SV P	29-136
GxCEFCLR	Channel Event Flag Clear Register, Group x	X590 _H	U, SV	U, SV P	29-136
GxREFCLR	Result Event Flag Clear Register, Group x	X594 _H	U, SV	U, SV P	29-137

Versatile Analog-to-Digital Converter (VADC)
Table 29-11 Registers Overview (cont'd)

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Page Num.
			Read	Write	
GLOBEFLAG	Global Event Flag Register	00E0 _H	U, SV	U, SV P	29-137
GxSEVNP	Source Event Node Pointer Register, Group x	X5C0 _H	U, SV	U, SV P	29-139
GxCEVNP0	Channel Event Node Pointer Register 0, Group x	X5A0 _H	U, SV	U, SV P	29-140
GxCEVNP1	Channel Event Node Pointer Register 1, Group x	X5A4 _H	U, SV	U, SV P	29-141
GxREVNP0	Result Event Node Pointer Register 0, Group x	X5B0 _H	U, SV	U, SV P	29-142
GxREVNP1	Result Event Node Pointer Register 1, Group x	X5B4 _H	U, SV	U, SV P	29-143
GLOBEVNP	Global Event Node Pointer Register	0140 _H	U, SV	U, SV P	29-144
GxSRACT	Service Request Software Activation Trigger, Group x	X5C8 _H	U, SV	U, SV P	29-146

Versatile Analog-to-Digital Converter (VADC)
29.12.3 Analog Module Connections in the TC26x B-Step

The VADC module accepts a number of analog input signals. The analog input multiplexers select the input channels to be converted from the signals available in this product.

Note: If an analog input channel is connected to an I/O port pin, make sure the output driver and/or the digital input path are disabled.

The exact number of analog input channels and the available connection to port pins depend on the employed product type (see also [Section 29.12.1](#)). A summary of channels enclosing all versions of the TC26x B-Step can be found below.

Input channels marked “PDD” provide a pull-down device for pull-down diagnostics.

Input channels marked “MD” can activate the pullup and pulldown devices for multiplexer diagnostics.

Input channels marked “AltRef” can be selected as an alternate reference voltage for conversions on channels of the same group or for conversions on channels of the same cluster.

Input channels marked “noAltref” cannot select an alternate reference voltage, but only the corresponding standard reference voltage.

Table 29-12 Analog Connections in the TC26x B-Step

Signal	Dir.	Source/Destin. ¹⁾	Description
V_{AREF}	I	VAREF1	positive analog reference
V_{AGND}	I	VAGND1	negative analog reference
G0CH0 (AltRef)	I	AN0 (X)	analog input channel 0 of group 0
G0CH1 (MD)	I	AN1 (X)	analog input channel 1 of group 0
G0CH2 (MD)	I	AN2 (X)	analog input channel 2 of group 0
G0CH3	I	AN3 (X)	analog input channel 3 of group 0
G0CH4 (noAltref)	I	AN4	analog input channel 4 of group 0
G0CH5 (noAltref)	I	AN5	analog input channel 5 of group 0
G0CH6 (noAltref)	I	AN6	analog input channel 6 of group 0
G0CH7 (PDD, noAltref)	I	AN7	analog input channel 7 of group 0
G0CH8 (AltRef)	I	AN8	analog input channel 8 of group 0
G0CH9 (MD)	I	-	analog input channel 9 of group 0
G0CH10 (MD)	I	AN10	analog input channel 10 of group 0
G0CH11	I	AN11	analog input channel 11 of group 0

Versatile Analog-to-Digital Converter (VADC)
Table 29-12 Analog Connections in the TC26x B-Step (cont'd)

Signal	Dir.	Source/Destin.¹⁾	Description
G0CH12	I	AN12	analog input channel 12 of group 0
G0CH13	I	AN13	analog input channel 13 of group 0
G1CH0 (AltRef)	I	AN16	analog input channel 0 of group 1
G1CH1 (MD)	I	AN17	analog input channel 1 of group 1
G1CH2 (MD)	I	AN18	analog input channel 2 of group 1
G1CH3 (PDD)	I	AN19	analog input channel 3 of group 1
G1CH4	I	AN20 (X)	analog input channel 4 of group 1
G1CH5	I	AN21 (X)	analog input channel 5 of group 1
G1CH6	I	-	analog input channel 6 of group 1
G1CH7	I	-	analog input channel 7 of group 1
G1CH8 (AltRef)	I	AN24, P40.0	analog input channel 8 of group 1
G1CH9 (MD)	I	AN25, P40.1	analog input channel 9 of group 1
G1CH10 (MD)	I	AN26, P40.2	analog input channel 10 of group 1
G1CH11	I	AN27, P40.3	analog input channel 11 of group 1
G1CH12	I	AN28	analog input channel 12 of group 1
G1CH13	I	AN29	analog input channel 13 of group 1
G2CH0 (AltRef)	I	AN32	analog input channel 0 of group 2
G2CH1 (MD)	I	AN33	analog input channel 1 of group 2
G2CH2 (MD)	I	-	analog input channel 2 of group 2
G2CH3 (PDD, noAltref)	I	AN35	analog input channel 3 of group 2
G2CH4	I	AN36, P40.6 (X)	analog input channel 4 of group 2
G2CH5	I	AN37, P40.7 (X)	analog input channel 5 of group 2
G2CH6	I	AN38, P40.8 (X)	analog input channel 6 of group 2
G2CH7	I	AN39, P40.9 (X)	analog input channel 7 of group 2
G2CH8 (AltRef)	I	-	analog input channel 8 of group 2
G2CH9 (MD)	I	-	analog input channel 9 of group 2
G2CH10 (MD)	I	AN44 (X)	analog input channel 10 of group 2
G2CH11	I	AN45 (X)	analog input channel 11 of group 2
G2CH12	I	AN46 (X)	analog input channel 12 of group 2
G2CH13	I	AN47 (X)	analog input channel 13 of group 2

Versatile Analog-to-Digital Converter (VADC)
Table 29-12 Analog Connections in the TC26x B-Step (cont'd)

Signal	Dir.	Source/Destin. ¹⁾	Description
G2CH14	I	AN48	analog input channel 14 of group 2
G2CH15	I	AN49	analog input channel 15 of group 2
G3CH0 (AltRef)	I	P00.12	analog input channel 0 of group 3
G3CH1 (MD)	I	P00.11	analog input channel 1 of group 3
G3CH2 (MD)	I	P00.10	analog input channel 2 of group 3
G3CH3	I	P00.9	analog input channel 3 of group 3
G3CH4	I	P00.8	analog input channel 4 of group 3
G3CH5	I	P00.7	analog input channel 5 of group 3
G3CH6	I	P00.6	analog input channel 6 of group 3
G3CH7	I	P00.5	analog input channel 7 of group 3
G3CH8 (AltRef)	I	P00.4	analog input channel 8 of group 3
G3CH9 (MD)	I	P00.3	analog input channel 9 of group 3
G3CH10 (MD)	I	P00.2	analog input channel 10 of group 3
G3CH11	I	P00.1	analog input channel 11 of group 3
G3CH12	I	-	analog input channel 12 of group 3
G3CH13	I	-	analog input channel 13 of group 3
G3CH14	I	-	analog input channel 14 of group 3
G3CH15	I	-	analog input channel 15 of group 3

1) Channels marked "(X)" in this columns are overlaid with input channels of the DSADC.

Versatile Analog-to-Digital Converter (VADC)
29.12.4 Digital Module Connections in the TC26x B-Step

The VADC module accepts a number of digital input signals and generates a number of output signals. This section summarizes the connection of these signals to other on-chip modules or to external resources via port pins.

Note: The control bitfields for triggers and gates select the corresponding multiplexer input. Values 0000_B ... 1111_B select inputs with suffix -A ... -P.

Table 29-13 Digital Connections in the TC26x B-Step

Signal	Dir.	Source/Destin.	Description
Gate Inputs for Each Group (x = 0 - 3, input line selected via bitfield GTSEL = [yyyy_B])			
GxREQGTA	I	GTM_adc _x _trig0	[0000 _B] GTM ADC trigger 0
GxREQGTB	I	GTM_adc _x _trig1	[0001 _B] GTM ADC trigger 1
GxREQGTC	I	CCU6061 TRIG0	[0010 _B] CCU6061 trigger output 0
GxREQGTD	I	CCU6061 TRIG1	[0011 _B] CCU6061 trigger output 1
GxREQGTE	I	CCU6061 TRIG2	[0100 _B] CCU6061 trigger output 2
GxREQGTF	I	-	[0101 _B] Gating input F, group x
GxREQGTG	I	-	[0110 _B] Gating input G, group x
GxREQGTH	I	-	[0111 _B] Gating input H, group x
GxREQGTI	I (s)	-	[1000 _B] Gating input I, group x
GxREQGTJ	I (s)	-	[1001 _B] Gating input J, group x
GxREQGTK	I (s)	-	[1010 _B] Gating input K, group x
GxREQGTL	I (s)	-	[1011 _B] Gating input L, group x
GyREQGTM	I	eru_pdout_y	[1100 _B] ERU pattern detection output y
GxREQGTN	I	-	[1101 _B] Gating input N, group x
GxREQGTO	I	-	[1110 _B] Gating input O, group x
GxREQGTP	I	-	[1111 _B] Gating input P, group x
GxREQGTzSEL	O	GxREQTRzP ¹⁾	Selected gating signal of the respective source (z = 0 - 1)
Gate Inputs for Global Background Source			
BGREQGTA	I	GTM_adc4_trig0	[0000 _B] GTM ADC trigger 0
BGREQGTB	I	GTM_adc4_trig1	[0001 _B] GTM ADC trigger 1
BGREQGTC	I	CCU6061 TRIG0	[0010 _B] CCU6061 trigger output 0
BGREQGTD	I	CCU6061 TRIG1	[0011 _B] CCU6061 trigger output 1

Versatile Analog-to-Digital Converter (VADC)
Table 29-13 Digital Connections in the TC26x B-Step (cont'd)

Signal	Dir.	Source/Destin.	Description
BGREQGTE	I	CCU6061 TRIG2	[0100 _B] CCU6061 trigger output 2
BGREQGTF	I	-	[0101 _B] Gating input F, background src.
BGREQGTG	I	-	[0110 _B] Gating input G, background src.
BGREQGTH	I	-	[0111 _B] Gating input H, background src.
BGREQGTI	I (s)	-	[1000 _B] Gating input I, background src.
BGREQGTJ	I (s)	-	[1001 _B] Gating input J, background src.
BGREQGTK	I (s)	-	[1010 _B] Gating input K, background src.
BGREQGTL	I (s)	-	[1011 _B] Gating input L, background src.
BGREQGTM	I	-	[1100 _B] Gating input M, background src.
BGREQGTN	I	-	[1101 _B] Gating input N, background src.
BGREQGTO	I	-	[1110 _B] Gating input O, background src.
BGREQGTP	I	-	[1111 _B] Gating input E, background src.
BGREQGTSEL	O	BGREQTRP ¹⁾	Selected gating signal

Trigger Inputs for Each Group
(x = 0 - 3, input line selected via bitfield TRSEL = [yyyy_B])

GxREQTRA	I	CCU60_SR3	[0000 _B] CCU60 service request output 3
GxREQTRB	I	CCU61_SR3	[0001 _B] CCU61 service request output 3
GxREQTRC	I	-	[0010 _B] Trigger input C, group x
GxREQTRD	I	-	[0011 _B] Trigger input D, group x
GxREQTRE	I	-	[0100 _B] Trigger input E, group x
GxREQTRF	I	-	[0101 _B] Trigger input F, group x
GxREQTRG	I	-	[0110 _B] Trigger input G, group x
GxREQTRH	I	eru_iout_x	[0111 _B] ERU interrupt output x
GxREQTRI	I (s)	-	[1000 _B] Trigger input I, group x
GxREQTRJ	I (s)	-	[1001 _B] Trigger input J, group x
GxREQTRK	I (s)	-	[1010 _B] Trigger input K, group x
GxREQTRL	I (s)	-	[1011 _B] Trigger input L, group x
GxREQTRM	I	vadc_gxsr1	[1100 _B] Service request 1, group x
GxREQTRN	I	vadc_c0sr1	[1101 _B] Service request 1, common grp. 0
GxREQTRO	I	vadc_c1sr1	[1110 _B] Service request 1, common grp. 1

Versatile Analog-to-Digital Converter (VADC)
Table 29-13 Digital Connections in the TC26x B-Step (cont'd)

Signal	Dir.	Source/Destin.	Description
GxREQTRzP	I	GxREQGTzSEL ¹⁾	[1111 _B] Extend triggers to selected gating input of the respective source (z = 0 - 1)
GxREQTRzSEL	O	-	Selected trigger signal of the respective source (z = 0 - 1)

Trigger Inputs for Global Background Source

BGREQTRA	I	CCU60_SR3	[0000 _B] CCU60 service request output 3
BGREQTRB	I	CCU61_SR3	[0001 _B] CCU61 service request output 3
BGREQTRC	I	-	[0010 _B] Trigger input C, background src.
BGREQTRD	I	-	[0011 _B] Trigger input D, background src.
BGREQTRE	I	-	[0100 _B] Trigger input E, background src.
BGREQTRF	I	-	[0101 _B] Trigger input F, background src.
BGREQTRG	I	eru_iout_3	[0110 _B] ERU interrupt output 3
BGREQTRH	I	eru_iout_4	[0111 _B] ERU interrupt output 4
BGREQTRI	I (s)	eru_iout_6	[1000 _B] ERU interrupt output 6
BGREQTRJ	I (s)	eru_iout_7	[1001 _B] ERU interrupt output 7
BGREQTRK	I (s)	-	[1010 _B] Trigger input K, background src.
BGREQTRL	I (s)	-	[1011 _B] Trigger input L, background src.
BGREQTRM	I	-	[1100 _B] Trigger input M, background src.
BGREQTRN	I	vadc_c0sr1	[1101 _B] Service request 1, common grp. 0
BGREQTRO	I	vadc_c1sr1	[1110 _B] Service request 1, common grp. 1
BGREQTRP	I	BGREQGTSEL ¹⁾	[1111 _B] Extend triggers to selected gating input of the background src.
BGREQTRSEL	O	-	Selected trigger signal of the background source

System-Internal Connections (x = 0 - 3)

otgb0[15:0]	O	OTGM	Alternate trigger buses for additional trace signals indicating the input signal sample phase (see OCS)
otgb1[15:0]	O	OTGM	
G0ARBCNT	O	GTM_TIM1_CH1	Outputs a (count) pulse for each arbiter round
G1ARBCNT	O	GTM_TIM2_CH0	
G2ARBCNT	O	GTM_TIM1_CH4	
G3ARBCNT	O	GTM_TIM2_CH1	

Versatile Analog-to-Digital Converter (VADC)
Table 29-13 Digital Connections in the TC26x B-Step (cont'd)

Signal	Dir.	Source/Destin.	Description
GxSR0	O	ICU	Service request 0 of group x
GxSR1	O	ICU	Service request 1 of group x
GxSR2	O	ICU	Service request 2 of group x
GxSR3	O	ICU	Service request 3 of group x
C0SR0	O	ICU GTM_TIM0_CH0 GTM_TIM1_CH4 GTM_TIM2_CH0	Service request 0 of common block 0
C0SR1	O	ICU GTM_TIM0_CH2 GTM_TIM1_CH6 GTM_TIM2_CH2	Service request 1 of common block 0
C0SR2	O	ICU GTM_TIM0_CH4 GTM_TIM1_CH0 GTM_TIM2_CH4	Service request 2 of common block 0
C0SR3	O	ICU GTM_TIM0_CH6 GTM_TIM1_CH2 GTM_TIM2_CH6	Service request 3 of common block 0
C1SR0	O	ICU GTM_TIM0_CH1 GTM_TIM1_CH5 GTM_TIM2_CH1	Service request 0 of common block 1
C1SR1	O	ICU GTM_TIM0_CH3 GTM_TIM1_CH7 GTM_TIM2_CH3	Service request 1 of common block 1
C1SR2	O	ICU GTM_TIM0_CH5 GTM_TIM1_CH1 GTM_TIM2_CH5	Service request 2 of common block 1
C1SR3	O	ICU GTM_TIM0_CH7 GTM_TIM1_CH3 GTM_TIM2_CH7	Service request 3 of common block 1

Versatile Analog-to-Digital Converter (VADC)
Table 29-13 Digital Connections in the TC26x B-Step (cont'd)

Signal	Dir.	Source/Destin.	Description
EMUX00	O	P02.6, P33.3	Control of external analog multiplexer interface 0
EMUX01	O	P02.7, P33.2	
EMUX02	O	P02.8, P33.1	
EMUX10	O	P00.6, P33.6	Control of external analog multiplexer interface 1
EMUX11	O	P00.7, P33.5	
EMUX12	O	P00.8, P33.4	
CBFLOUT0	O	-	Common boundary flag output 0
CBFLOUT1	O	-	Common boundary flag output 1
CBFLOUT2	O	-	Common boundary flag output 2
CBFLOUT3	O	-	Common boundary flag output 3
VADCG0BFL0	O	P33.4	Boundary flag 0 output of group 0
G0BFL0	O	GTM_ stat_in00_[0]	Boundary flag 0 level of group 0
G0BFSEL0	I	GTM_ mcs_trig00_[1]	Boundary flag 0 (group 0) source select
G0BFDAT0	I	GTM_ mcs_trig00_[0]	Boundary flag 0 (group 0) alternate data
VADCG0BFL1	O	P33.5	Boundary flag 1 output of group 0
G0BFL1	O	GTM_ stat_in00_[1]	Boundary flag 1 level of group 0
G0BFSEL1	I	GTM_ mcs_trig00_[3]	Boundary flag 1 (group 0) source select
G0BFDAT1	I	GTM_ mcs_trig00_[2]	Boundary flag 1 (group 0) alternate data
VADCG0BFL2	O	P33.6	Boundary flag 2 output of group 0
G0BFL2	O	-	Boundary flag 2 level of group 0
G0BFSEL2	I	0	Boundary flag 2 (group 0) source select
G0BFDAT2	I	0	Boundary flag 2 (group 0) alternate data
VADCG0BFL3	O	P33.7 CCU60_CTRAPC	Boundary flag 3 output of group 0
G0BFL3	O	-	Boundary flag 3 level of group 0
G0BFSEL3	I	0	Boundary flag 3 (group 0) source select

Versatile Analog-to-Digital Converter (VADC)
Table 29-13 Digital Connections in the TC26x B-Step (cont'd)

Signal	Dir.	Source/Destin.	Description
G0BFDAT3	I	0	Boundary flag 3 (group 0) alternate data
VADCG1BFL0	O	P33.0	Boundary flag 0 output of group 1
G1BFL0	O	GTM_ stat_in01_[0]	Boundary flag 0 level of group 1
G1BFSEL0	I	GTM_ mcs_trig10_[1]	Boundary flag 0 (group 1) source select
G1BFDAT0	I	GTM_ mcs_trig10_[0]	Boundary flag 0 (group 1) alternate data
VADCG1BFL1	O	P33.1	Boundary flag 1 output of group 1
G1BFL1	O	GTM_ stat_in01_[1]	Boundary flag 1 level of group 1
G1BFSEL1	I	GTM_ mcs_trig10_[3]	Boundary flag 1 (group 1) source select
G1BFDAT1	I	GTM_ mcs_trig10_[2]	Boundary flag 1 (group 1) alternate data
VADCG1BFL2	O	P33.2	Boundary flag 2 output of group 1
G1BFL2	O	-	Boundary flag 2 level of group 1
G1BFSEL2	I	0	Boundary flag 2 (group 1) source select
G1BFDAT2	I	0	Boundary flag 2 (group 1) alternate data
VADCG1BFL3	O	P33.3	Boundary flag 3 output of group 1
G1BFL3	O	-	Boundary flag 3 level of group 1
G1BFSEL3	I	0	Boundary flag 3 (group 1) source select
G1BFDAT3	I	0	Boundary flag 3 (group 1) alternate data
VADCG2BFL0	O	P00.4	Boundary flag 0 output of group 2
G2BFL0	O	GTM_ stat_in02_[0]	Boundary flag 0 level of group 2
G2BFSEL0	I	GTM_ mcs_trig01_[1]	Boundary flag 0 (group 2) source select
G2BFDAT0	I	GTM_ mcs_trig01_[0]	Boundary flag 0 (group 2) alternate data
VADCG2BFL1	O	P00.5	Boundary flag 1 output of group 2
G2BFL1	O	GTM_ stat_in02_[1]	Boundary flag 1 level of group 2

Versatile Analog-to-Digital Converter (VADC)
Table 29-13 Digital Connections in the TC26x B-Step (cont'd)

Signal	Dir.	Source/Destin.	Description
G2BFSEL1	I	GTM_ mcs_trig01_[3]	Boundary flag 1 (group 2) source select
G2BFDAT1	I	GTM_ mcs_trig01_[2]	Boundary flag 1 (group 2) alternate data
VADCG2BFL2	O	P00.6	Boundary flag 2 output of group 2
G2BFL2	O	-	Boundary flag 2 level of group 2
G2BFSEL2	I	0	Boundary flag 2 (group 2) source select
G2BFDAT2	I	0	Boundary flag 2 (group 2) alternate data
VADCG2BFL3	O	P00.7	Boundary flag 3 output of group 2
G2BFL3	O	-	Boundary flag 3 level of group 2
G2BFSEL3	I	0	Boundary flag 3 (group 2) source select
G2BFDAT3	I	0	Boundary flag 3 (group 2) alternate data
VADCG3BFL0	O	P10.0, P10.6	Boundary flag 0 output of group 3
G3BFL0	O	-	Boundary flag 0 level of group 3
G3BFSEL0	I	GTM_ mcs_trig11_[1]	Boundary flag 0 (group 3) source select
G3BFDAT0	I	GTM_ mcs_trig11_[0]	Boundary flag 0 (group 3) alternate data
VADCG3BFL1	O	P10.1, P10.7	Boundary flag 1 output of group 3
G3BFL1	O	-	Boundary flag 1 level of group 3
G3BFSEL1	I	GTM_ mcs_trig11_[3]	Boundary flag 1 (group 3) source select
G3BFDAT1	I	GTM_ mcs_trig11_[2]	Boundary flag 1 (group 3) alternate data
VADCG3BFL2	O	P10.2	Boundary flag 2 output of group 3
G3BFL2	O	-	Boundary flag 2 level of group 3
G3BFSEL2	I	0	Boundary flag 2 (group 3) source select
G3BFDAT2	I	0	Boundary flag 2 (group 3) alternate data
VADCG3BFL3	O	P10.3	Boundary flag 3 output of group 3
G3BFL3	O	-	Boundary flag 3 level of group 3
G3BFSEL3	I	0	Boundary flag 3 (group 3) source select
G3BFDAT3	I	0	Boundary flag 3 (group 3) alternate data

Versatile Analog-to-Digital Converter (VADC)

- 1) Internal signal connection.

Versatile Analog-to-Digital Converter (VADC)

29.13 Use Case Example for VADC

This section provides a code example for the VADC module to give an overview about the core functionality. The code example realizes an event triggered analog to digital conversion. The result is stored as a 12-bit digital value and triggers a result service request.

The VADC contains several ADC groups (G0 ... G3). Each group is connected to up to 8 analog input channels via a multiplexer. The following code example initializes one of the groups and uses one static allocated analog channel as input.

The code example realizes an analog-to-digital conversion by using the background source. The conversion can either run continuously in autoscan mode or can be triggered by an event, like in the following example. This is useful if the conversion needs to be executed at certain times. The result from the most recent conversion is accessible in the global result register **GLOBRES**. A result service request is activated after each conversion. The initialization is done in the following order:

1. Load the global VADC module registers
2. Enable the analog/digital converter 0 (G0)
3. Select channel 0 of G0 as analog input
4. Initialize the conversion result service request
5. Start first conversion

Versatile Analog-to-Digital Converter (VADC)
Step Description to Initialize the VADC Module:

(Line 1) reset ENDINIT to get access to ENDINIT protected registers. (see also ENDINIT protection in chapter SCU).

(Line 2) enable control of the module, in clock control register **CLC**.

(Line 3) read back (dummy variable has to be defined). The reading process ensures that the write process from line 2 is done.

(Line 4) set ENDINIT to lock the protected register again.

(Line 5) set group 0 to normal operation mode. This activates the converter of group 0. (see also **Section 29.4.1** and **GxARBCFG (x = 0 - 3)**).

(Line 6) load the global configuration register GLOBCFG with the divider factor for the analog internal clock (see also **Figure 29-7**). Set DIVWC[15] to enable write access and initiate the start-up calibration by setting SUCAL[31]. The calibration is necessary to get a precise conversion.

(Line 7) This line enables the arbitration slot 2, that's the background scan source slot which the examples uses. (see also Arbitration Priority Register **GxARBPR (x = 0 - 3)**)

(Line 8) This line sets RESTBS[20], so each conversion result from group 0 input channel 0 is now stored in the global result register GLOBRES. (see also GxCHCTry)

(Line 9) This line selects input channel 0 of group 0 as part of the background scan sequence. (see also **BRSELx (x = 0 - 3)**).

(Line 10) By setting ENGT[0..1]=01 a conversion request can be issued for every channel which was enabled in line 10. here only channel 0. (see also **BRSMR**)

(Line 11) This line enables the group 0 service request in the service request control register SRC_VADCG0SR0 and sets the interrupt priority to VADC0INT (1...255)

(Line 12) The interruptHandlerInstall function gets called (this function has to be written first, see interrupt handler example in chapter IR for more details). This function installs the interrupt service routine (ISR) entry address in the interrupt vector array with the priority VADC0INT.

(Line 13) This command enables the result interrupt. It occurs after a result event.

(Line 14) wait until start-up calibration (start in line 6) is done.

(Line 15) setting LDEV=1 generates a load event which starts a conversion.

Note: The conversion result ISR function prototype would be: void VADC_SCAN_irq (void); here could be your ISR code;

Versatile Analog-to-Digital Converter (VADC)
Basic Initialization of the VADC:

```
//=====> Load global module registers
(1) SCU_vResetENDINIT (0); // Access to ENDINIT-prot. reg.
(2) VADC_CLC = 0x0000; // Enable module clock and ctrl.
(3) dummy = VADC_CLC; // Read back ensures write oper.
(4) SCU_vSetENDINIT (0); // Lock ENDINIT-protected reg.
//=====> Enable converter for group 0
(5) VADC_G0ARBCFG = 0x3; // ANONC = 11, analog converter ON
(6) VADC_GLOBCFG = (1<<31) /* SUCAL = 1, start-up calib. */ \
| (1<<15) /* DIVWC = 1, enable write */ \
| 0x9; // DIVA = 9 (clock prescaler)
(7) VADC_G0ARBPR = (1<<26); // AREN2 = 1, enable arb. slot 2
// (= background source)
(8) VADC_G0CHCTR0 = (1<<20); // RESTBS = 1, global result reg.
(9) VADC_BRSSSEL0 = 0x1; // Select CH0 of group 0 for scan
(10) VADC_BRSMR = 0x1; // ENGT = 10B, enable conv. req.
//=====> Init and install service request
(11) SRC_VADCCG0SR0 = (1<<10) /* Enable SR node 0, group 0 */ \
| VADC0INT; // Set prio to <VADC0INT>(1..255)
(12) interruptHandlerInstall (VADC0INT, & VADC_SCAN_irq);
(13) VADC_GLOBRCR = (1<<31); // SRGEN = 1, result service requ.
//=====> Wait for completion of startup cal.
(14) while((VADC_G0ARBCFG.U & 0x30000000) != 0x20000000);
// CALS = 1, CAL = 0: calibr. done
//=====> Start a conversion
(15) VADC_BRSMR |= (1<<9); // LDEV = 1, generate a load event
```

Note: Before a service request can occur, the service request system must be globally enabled. The Interrupt Control Register (ICR) holds the global interrupt enable bit (ICR.IE) which enables the CPU service request system. Most compilers support the attribute (or similar) to set this bit (see Architecture Manual for more details):
__enable();

30 Delta-Sigma Analog-to-Digital Converter (DSADC)

The Delta-Sigma Analog-to-Digital Converter module (DSADC) of the TC26x B-Step provides a series of analog input channels connected to on-chip modulators using the Delta/Sigma (DS) conversion principle.

The on-chip demodulator channels convert these inputs to discrete digital values.

The number of inputs and DSADC channels depends on the chosen product type (refer to [“Implementation into the TC26x B-Step” on Page 30-85](#)).

Each converter channel can operate independent of the others, controlled by a dedicated set of registers. The results of each channel can be stored in a dedicated channel-specific result register.

The on-chip filter stages generate digital results from the modulator signal. The on-chip modulators accept differential or single-ended input signals.

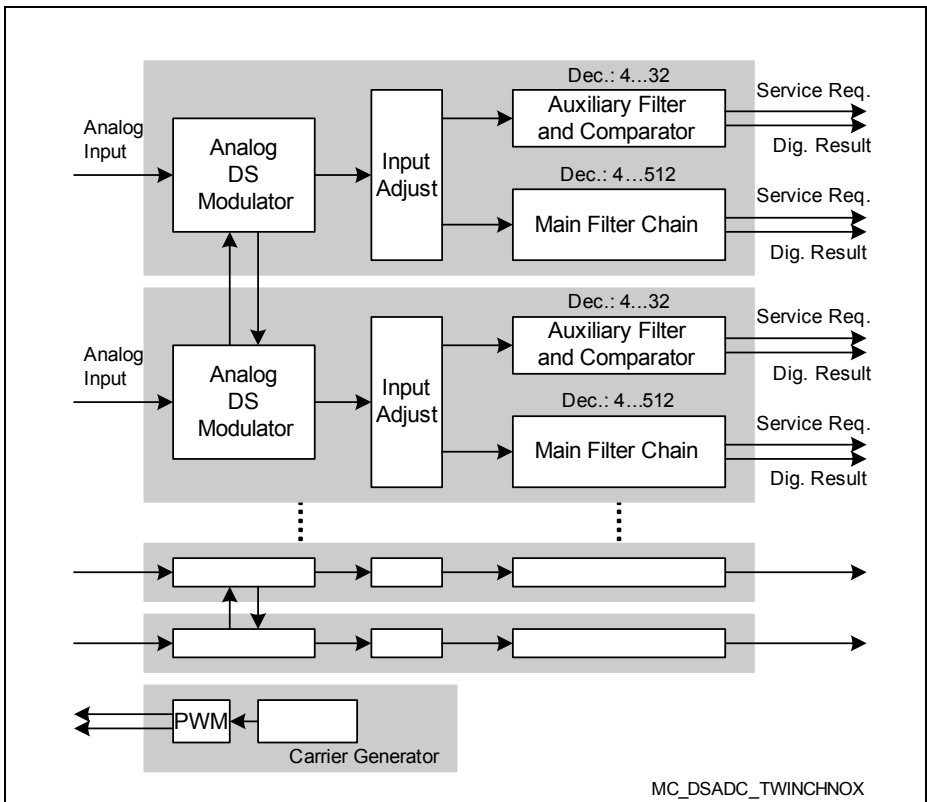


Figure 30-1 DSADC Module Overview

Delta-Sigma Analog-to-Digital Converter (DSADC)

You will find the following major sections within this chapter:

- **“Introduction and Basic Structure” on Page 30-4**
- **“Configuration of General Functions” on Page 30-6**
- **“Input Channel Configuration” on Page 30-21**
- **“Main Filter Chain” on Page 30-48**
- **“Auxiliary Filter” on Page 30-58**
- **“Filter Configuration and Control” on Page 30-61**
- **“Conversion Result Handling” on Page 30-70**
- **“Service Request Generation” on Page 30-72**
- **“Resolver Support” on Page 30-75**
- **“Time-Stamp Support” on Page 30-83**
- **“Implementation into the TC26x B-Step” on Page 30-85** (including a register summary)

Features

The following features describe the functionality of a DS Converter:

- Selectable connections for each input line (differential or single-ended inputs)
 - Input multiplexers to select input pins, switching controllable by external triggers
 - Alternate connection to supply voltage, half-scale voltage or reference ground
 - Common mode hold voltage selectable for each individual pin during idle phases
- Differential input amplifier
 - Input voltage range 0...5 V
 - Common mode voltage range selectable
 - Programmable gain (1/2/4/8/16)
 - DC Offset <5 mVDC, gain error < ±1%
 - Input impedance 100 kΩ
 - Calibration for offset and gain
- On-chip 3rd-order modulator
 - Sample frequency 10...20 MHz
 - High-performance and low-power modes
 - External run control
- Main demodulator (concatenated hardware filter stages)
 - Configurable CIC filter with decimation rates of 4...128
 - FIR filter with 8 coefficients (9-bit) with decimation rate 2
 - FIR filter with 28 coefficients (9-bit) with decimation rate 2
 - Optional high-pass filter for DC compensation ($f_{-3dB} = 10^{-5} \times f_d$)
 - Integration stage with external gating
 - Automatic offset correction
 - Time-stamp support
 - Pass Band 10...100 kHz, output sampling rate $f_d = 30...300$ kHz, Pass band ripple $f_d: \pm 1\%$,

Delta-Sigma Analog-to-Digital Converter (DSADC)

Stop band attenuation: $0.5 \dots 1 \times f_d$: >40 dB / $1 \dots 1.5 \times f_d$: >45 dB /
 $1.5 \dots 2 \times f_d$: > 50 dB / $2 \dots 2.5 \times f_d$: >55 dB / $2.5 \dots \text{OSR}/2 \times f_d$: > 60 dB

- Parallel auxiliary demodulator for limit checking
 - Configurable CIC filter with decimation rates of 4...32
 - Two-level boundary comparator
- Built-in support for resolver applications
 - Carrier signal generator
 - Delay compensation for position signals
 - Carrier cancelation by integrator unit
- DMA/Service-requests
 - Result events on new valid results with external gating
 - Alarm events from limit checking

Table 30-1 Abbreviations used in DSADC chapter

Abbreviation	Meaning
ADC	Analog to Digital Converter
CIC	Cyclic Integrating Comb (filter)
DMA	Direct Memory Access (controller)
DNL	Differential Non-Linearity (error)
DSADC	Delta-Sigma (conversion principle) Analog to Digital Converter
FIR	Finite Impulse Response (filter)
INL	Integral Non-Linearity (error)
LSB _n	Least Significant Bit: finest granularity of the analog value in digital format, represented by one least significant bit of the conversion result with n bits resolution (measurement range divided in 2 ⁿ equally distributed steps)
OSR	Oversampling Ratio
PWM	Pulse Width Modulation
SNR	Signal-Noise Ratio

Delta-Sigma Analog-to-Digital Converter (DSADC)**30.1 Introduction and Basic Structure**

The Delta-Sigma Analog to Digital Converter module of the TC26x B-Step provides several channels with a main and an auxiliary demodulator including configurable filters for decimation (see [Figure 30-2](#)).

The on-chip buffered 3rd order modulator converts an analog input signal to a datastream.

The main chain of digital filters builds the demodulator which produces result values at a configurable output rate. The elements of the filter chain can be selected according to the requirements of the application. The filter chain configuration determines the attenuation and delay properties of the filter. The decimation at a configurable rate reduces the input sampling rate to a lower result data rate.

The configurable CIC filter provides the basic filtering and decimation with a selectable decimation rate.

Two FIR filters, each with a decimation rate of 2, allow effective signal shaping by attenuating the upper frequencies of the signal spectrum.

The high-pass filter provides offset compensation by removing the DC component of the input signal.

The integrator accumulates a configurable amount of result values. The number of samples is programmable or can be controlled by a hardware signal. This function supports resolver applications to get the baseband signal for the motor position calculation. Furthermore, the integrator can be also used in shunt current measurement applications.

A smaller but quicker parallel auxiliary filter with a comparator supports limit checking, e.g. for overcurrent detection. Two limit values can be defined to restrict the generation of service requests to result values within a configurable area. This saves CPU performance and/or DMA bandwidth.

A carrier signal can be generated to support resolver applications.

The on-chip carrier signal generator produces a selectable output signal (sine, triangle, rectangle) which can be used to drive a resolver. Synchronization of each input signal to the carrier signal ensures correct integration of the resolver input signals.

Service requests can be generated to trigger DMA transfers or to request CPU service.

The basic module clock f_{DSD} is connected to the system clock signal f_{SPB} .

Delta-Sigma Analog-to-Digital Converter (DSADC)

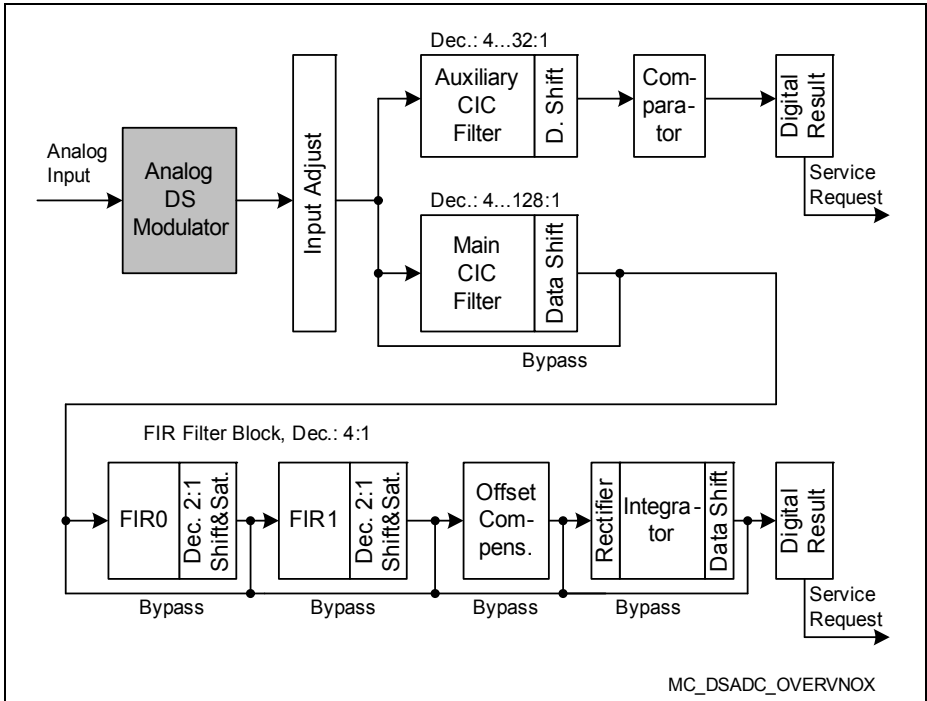


Figure 30-2 DSADC Structure Overview

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.2 Configuration of General Functions

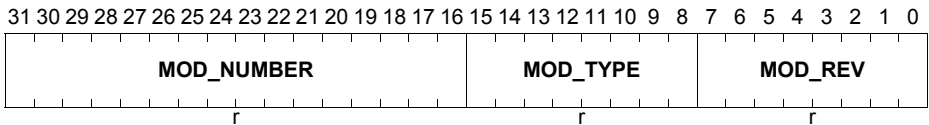
Several parameters can be configured to adapt the functionality of the DSADC to the requirements of the actual application (see [Section 30.2.4](#)).

30.2.1 Module Identification

The module identification register indicates the version of the DSADC module that is used in the TC26x B-Step.

ID

Module Identification Register (0008_H) **Reset Value: 00C6 C0XX_H**



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Indicates the revision number of the implementation. This information depends on the design step.
MOD_TYPE	[15:8]	r	Module Type This internal marker is fixed to C0 _H .
MOD_NUMBER	[31:16]	r	Module Number Indicates the module identification number (00C6 _H = DSADC)

Delta-Sigma Analog-to-Digital Converter (DSADC)
30.2.2 System Registers

A set of standardized registers provides general access to the module and controls basic system functions.

The Clock Control Register **CLC** allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application.

Register **CLC** controls the module clock signal and the reactivity to the sleep signal.

CLC
Clock Control Register (0000_H) Reset Value: 0000 0003_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	E DIS	0	DIS S	DIS R
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module. 0 _B On request: enable the module clock 1 _B Off request: stop the module clock
DISS	1	r	Module Disable Status Bit 0 _B Module clock is enabled 1 _B Off: module is not clocked
0	2	r	Reserved, write 0, read as 0
EDIS	3	rw	Sleep Mode Enable Control Used to control module's reaction to sleep mode. 0 _B Sleep mode request is enabled and functional 1 _B Module disregards the sleep mode control signal
0	[31:4]	r	Reserved, write 0, read as 0

The OCDS control and status register OCS controls the module's behavior in suspend mode (used for debugging).

Delta-Sigma Analog-to-Digital Converter (DSADC)

The OCDS Control and Status (OCS) register is cleared by Debug Reset.

The OCS register can only be written when the OCDS is enabled.

If OCDS is being disabled, the OCS register value will not change.

When OCDS is disabled the OCS suspend control is ineffective.

Write access is 32 bit wide only and requires Supervisor Mode.

OCS
OCDS Control and Status Register (0028_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	SUS STA	SUS _P	SUS				0	0	0	0	0	0	0	0
r	r	rh	w	rw				r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
0	[23:0]	r	Reserved, write 0, read as 0
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0000 _B Will not suspend 0001 _B Hard suspend: Clock is switched off immediately. 0010 _B Soft suspend channel 0 0100 _B Soft suspend channel 2 0101 _B Soft suspend channel 3 Others Reserved <i>Note: In soft suspend mode, the respective channel is stopped after the next result has been stored.</i>
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[31:30]	r	Reserved, write 0, read as 0

Delta-Sigma Analog-to-Digital Converter (DSADC)

Note: Register OCS is cleared by a Debug Reset. While OCDS is disabled, OCS is reset with each system reset. A system reset has no effect while OCDS is enabled. Write access requires supervisor mode.

Delta-Sigma Analog-to-Digital Converter (DSADC)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on-chip bus master TAG IDs 00 0000_B to 01 1111_B (see On-Chip Bus chapter for the mapping of product TAG ID <-> master peripheral). Register ACCEN0 provides one enable bit for each possible TAG ID encoding.

Register ACCEN0 itself is protected by the Safety Endinit feature.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 00 0000_B, EN1 -> TAG ID 00 0001_B, ..., EN31 -> TAG ID 01 1111_B (TAG IDs 1X XXXX_B are not used).

ACCEN0
Access Enable Register 0
(003C_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B No Write access 1 _B Write access will be executed

1) The Access Enable functionality controls only write transactions to registers CLC, OCS, KRSTx, and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

Delta-Sigma Analog-to-Digital Converter (DSADC)

Individual Module Reset

The Kernel Reset Registers **KRST0/KRST1** are used to reset the related module kernel. To reset a module kernel it is necessary to set the RST bits in both Kernel Reset Registers. The RST bits will be cleared by the BPI with the end of the BPI kernel reset sequence.

Note: A module kernel reset does not reset registers that do not belong to the module kernel: CLC, OCS, ACCEN0 and, of course, the KRST registers.*

Kernel Reset Registers 0 and 1 each include bit RST. To trigger a module reset, both RST bits must be set. They will be cleared automatically with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set in the same clock cycle the RST bit is cleared. This bit indicates that a kernel reset was processed. Bit RSTSTAT can be cleared by setting bit CLR in register **KRSTCLR**.

Note: During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

KRST0

Kernel Reset Register 0 (0034_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	RST STA T	RST
r	r	r	r	r	r	r	r	r	r	r	r	r	r	rh	rwh

Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>Request a kernel reset. The reset is executed if the reset bits of both kernel reset registers are set.</p> <p>0_B No action</p> <p>1_B A kernel reset was requested</p> <p>RST is cleared after the kernel reset was executed.</p>

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
RSTSTAT	1	rh	Kernel Reset Status Indicates an executed kernel reset. RSTSTAT is set after the execution of a kernel reset in the same clock cycle in which the reset bits are cleared. 0_B No kernel reset was executed 1_B Kernel reset was executed Clear RSTSTAT by setting bit CLR in register KRSTCLR.
0	[31:2]	r	Reserved, write 0, read as 0

KRST1
Kernel Reset Register 1
(0030_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	RST
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	rwh

Field	Bits	Type	Description
RST	0	rwh	Kernel Reset Request a kernel reset. The reset is executed if the reset bits of both kernel reset registers are set. 0_B No action 1_B A kernel reset was requested RST is cleared after the kernel reset was executed.
0	[31:1]	r	Reserved, write 0

Delta-Sigma Analog-to-Digital Converter (DSADC)

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (RSTSTAT).

KRSTCLR
Kernel Reset Status Clear Register (002C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	CLR
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	w

Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved, write 0, read as 0

Delta-Sigma Analog-to-Digital Converter (DSADC)
30.2.3 Register Access Control

Several protection schemes are provided to prevent unintended write access to control bitfields of the DSADC.

- The registers of the DSADC are protected by the general access control mechanism that is configured by register **ACCEN0**.
- A specific register access control scheme provides a versatile protection scheme against unintended corruption of register contents. Register ACCPROT allows the restriction of write accesses for several groups of registers. The registers to be protected can be selected by the user. The table below lists the registers that belong to each register group.
Register ACCPROT itself is protected by the Safety Endinit feature.
- Groups of bitfields within a register may also be protected by an associated write control bit. This write control bit (xxWC) must be written with 1 along with the write access to the intended bitfield(s).

ACCPROT

Access Protection Register (0090_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RG 11	RG 10	0	0	0	0	0	0	0	0	0	RG 04	RG 03	RG 02	RG 01	RG 00
rw	rw	r	r	r	r	r	r	r	r	r	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RG0x (x = 0 - 4)	x	rw	Register Group x, Channels 0, 2, 3 0 _B Full access to register group x 1 _B Write access to registers of group x is blocked
0	[13:5]	r	Reserved, write 0, read as 0
RG10, RG11	14, 15	rw	Register Group 0/1, General Control 0 _B Full access to register group x 1 _B Write access to registers of group x is blocked
0	[31:16]	r	Reserved, write 0, read as 0

Delta-Sigma Analog-to-Digital Converter (DSADC)

Table 30-2 Register Protection Groups

Control Bits	Registers	Notes
RG00	FCFGM0 ... FCFGM3, FCFGC0 ... FCFGC3, FCFGA0 ... FCFGA3	Filter control
RG01	IWCTR0 ... IWCTR3	Integrator control
RG02	DICFG0 ... DICFG3	Demodulator control
RG03	RECTCFG0 ... RECTCFG3, CGSYNC0 ... CGSYNC3	Demodulator control
RG04	MODCFG0 ... MODCFG3	Modulator control
RG10	GLOBCFG, GLOBRC GLOBVCMH0, GLOBVCMH2	Global control
RG11	CGCFG	Carrier generator control

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.2.4 Global Configuration Registers

The global configuration register GLOBCFG selects the source for the internal clock signal which can be used to drive the modulators.

It also selects internal measurement modes.

The global run control register GLOBRC controls the general operation of the available channels.

GLOBCFG

Global Configuration Register (0080_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	PS WC	ICT	0	LO SUP	IBSEL			
r	r	r	r	r	r	r	r	w	rw	r	rw	rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	IRM 0	0	0	0	0	0	0	0	0	MCSEL		
r	r	r	r	rw	r	r	r	r	r	r	r	r	rw		

Field	Bits	Type	Description
MCSEL	[2:0]	rw	Modulator Clock Select Selects the source for the on-chip clock source f_{CLK} for the modulator clock. Only one bit may be set at any given time. 000 _B Internal clock off, no source selected 001 _B f_{DSD} 010 _B f_{ERAY} 100 _B f_{OSCO} All other combinations are reserved.
0	[10:3]	r	Reserved, write 0, read as 0
IRM0	11	rw	Internal Resistance Measurement Control 0 _B Normal operation 1 _B Enable the testmode for the input resistors on channel 0, channel 2 and channel 3 (IRM0)
0	[15:12]	r	Reserved, write 0, read as 0

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
IBSEL	[19:16]	rw	Bias Current Select¹⁾ Selects the internal bias current for the input buffer. 0000 _B Very low bias current ... 1111 _B Very high bias current
LOSUP	20	rw	Low Power Supply Voltage Select Adjusts the analog circuitry to the supply voltage used in the application system. 0 _B 5 V power supply is connected 1 _B 3.3 V power supply is connected <i>Note: Make sure to keep LOSUP = 0 in the case of a 5 V supply.</i>
0	21	r	Reserved, write 0, read as 0
ICT	22	rw	Internal Channel Test 0 _B Normal operation 1 _B Channel test mode active <i>Note: This is a test mode only. Make sure to keep ICT = 0 during normal operation.</i>
PSWC	23	w	Write Control for Power Supply Parameters 0 _B No write access to power supply parameters 1 _B Bitfields ICT, LOSUP, IBSEL can be written
0	[31:24]	r	Reserved, write 0, read as 0

1) This is a preset value. Do not change this value when writing to GLOBCFG, unless there is an explicit recommendation.

GLOBRC
Global Run Control Register
(0088_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	M3 RUN	M2 RUN	0	M0 RUN
r	r	r	r	r	r	r	r	r	r	r	r	rw	rw	r	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	CH3 RUN	CH2 RUN	0	CH0 RUN
r	r	r	r	r	r	r	r	r	r	r	r	rw	rw	r	rw

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
CHxRUN (x = 0, 2, 3)	x	rw	Channel x Run Control Each bit (when set) enables the corresponding demodulator channel. 0 _B Stop channel x 1 _B Demodulator channel x is enabled and runs When CHxRUN is set, all filter blocks are cleared.
0	1, [15:4]	r	Reserved, write 0, read as 0
MxRUN (x = 0, 2, 3)	16 + x	rw	Modulator x Run Control Each bit (when set) enables the corresponding modulator. 0 _B Stop on-chip modulator x 1 _B Modulator x is enabled and can run ¹⁾
0	17, [31:20]	r	Reserved, write 0, read as 0

1) Additional run control via the selected gate signal is possible by the automatic power control feature (bit APC in register **MODCFGx** (x = 0, 2, 3)).

Delta-Sigma Analog-to-Digital Converter (DSADC)

The Initial Configuration Registers IGCFG and ICCFGx¹⁾ control specific setup features of the on-chip modulators. Use the default values unless other values are recommended by additional documentation. The write enable bit helps to prevent unintended write access after writing the intended configuration.

IGCFG
Initial Global Config. Register (00D0_H) Reset Value: 8000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR EN	0	0	0	0	0	GLOBSP									
rw	r	r	r	r	r	rw									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	DITRIM		
r	r	r	r	r	r	r	r	r	r	r	r	r	rw		

Field	Bits	Type	Description
DITRIM	[2:0]	rw	Trimming Value for the Dithering Function This trim value is used for all modulators of the device. 000 _B Minimum dithering intensity 001 _B Low dithering intensity 011 _B Medium dithering intensity 111 _B High dithering intensity Other combinations are reserved.
0	[15:3]	r	Reserved, write 0, read as 0
GLOBSP	[25:16]	rw	Global Setup Parameters for the MultiADC Do not change this bitfield, i.e. keep SPWC = 0 when writing to this register.
0	[30:26]	r	Reserved, write 0, read as 0
WREN	31	rw	Write Enable 0 _B No write access to this register 1 _B Register IGCFG can be written

1) Due to the pair-coupled structure of the on-chip twin-modulators, only registers ICCFGx with an even index are available.

For example, ICCFG0 controls the modulators of channel 0 and channel 1, ICCFG1 has no function.

Delta-Sigma Analog-to-Digital Converter (DSADC)

ICCFGx (x = 0, 2, 3)
Initial Channel Config. Reg. x (x * 0100_H + 01D0_H) **Reset Value: 8000 0003_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WR EN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
rw	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	TWINS P						0	0	0	I REN	0	0	DI1 EN	DI0 EN
r	r				rw			r	r	r	rw	r	r	rw	rw

Field	Bits	Type	Description
DI0, DI1	0, 1	rw	Dithering Function Enable Controls the dithering function for each modulator of the respective twin separately. 0 _B Disable dithering 1 _B Dithering is enabled (default)
0	[3:2]	r	Reserved, write 0, read as 0
I REN	4	rw	Integrator Reset Enable Controls the influence of an overload condition on resetting the modulator's integrator stages. 0 _B No integrator reset 1 _B Integrators are reset in case of an overload condition.
0	[7:5]	r	Reserved, write 0, read as 0
TWINS P	[13:8]	rw	Setup Parameters for this Twin Modulator Do not change this bitfield, i.e. keep SPWC = 0 when writing to this register.
0	[30:14]	r	Reserved, write 0, read as 0
WREN	31	rw	Write Enable 0 _B No write access to this register 1 _B Register ICCFGx can be written

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.3 Input Channel Configuration

The input data for a channel is obtained from the on-chip modulator.

The on-chip modulator can directly convert a differential or single-ended input signal to an input data stream. The performance can be increased by connecting two modulators to a pair. In this case, one of the two channels operates in 4th order mode while the other one operates in 2nd order mode.

The on-chip modulator runs on an internal clock.

The figure below summarizes the data path through the on-chip modulator.

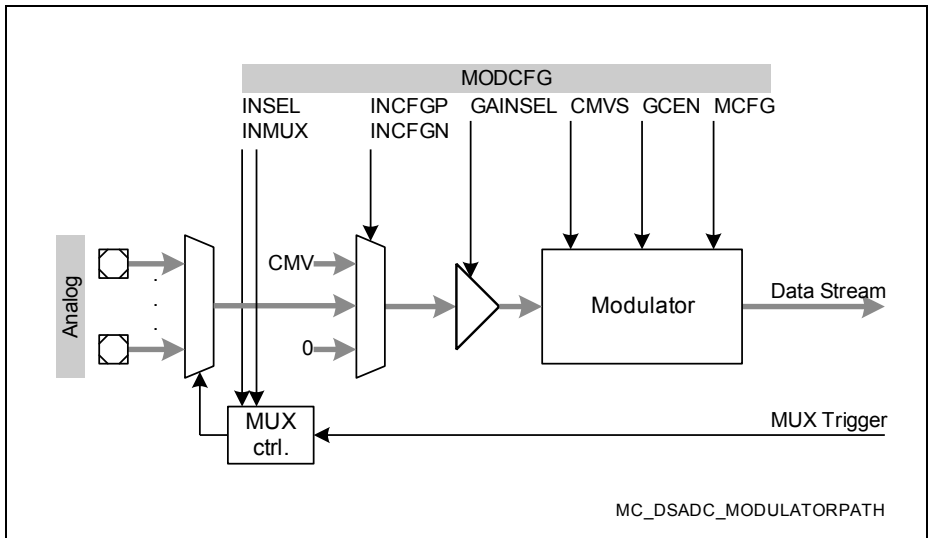


Figure 30-3 Analog Input Path

Delta-Sigma Analog-to-Digital Converter (DSADC)

The modulator clock can be generated in different ways:

- The modulator clock is derived on-chip from the internal clock f_{CLK} (module clock, ERAY PLL clock, oscillator clock).
- The internal clock (f_{CLK}) is also used for the on-chip modulators and for the carrier generator. To ensure a stable modulator clock, it can optionally be derived from the FlexRay clock or directly from oscillator OSC0.
- The used modulator clock also drives the on-chip carrier generator. This enables synchronous operation of carrier generator and integrator.

A trigger signal can be input from a selectable input. These inputs are connected to on-chip peripherals or to pins (see [Section 30.11.4](#)). This trigger signal can be used for different purposes:

- Integration trigger:
The external signal defines the integration window, i.e. the timespan during which result values are integrated.
- Timestamp trigger:
The external signal requests the actualization of the timestamp register.
- Input multiplexer trigger:
The external signal requests the switching of the analog input multiplexer to the next lower input or to the defined start value, respectively.
- Modulator run gate:
The on-chip modulator can be controlled by an external gate signal while it is enabled. In this case it can be disabled temporarily to save power.
- Service request gate:
Service requests for the main filter chain can be restricted to the high or low times of the selected trigger signal.

Note: To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

The figure below summarizes these three signal paths and indicates the source of the corresponding control information.

Delta-Sigma Analog-to-Digital Converter (DSADC)

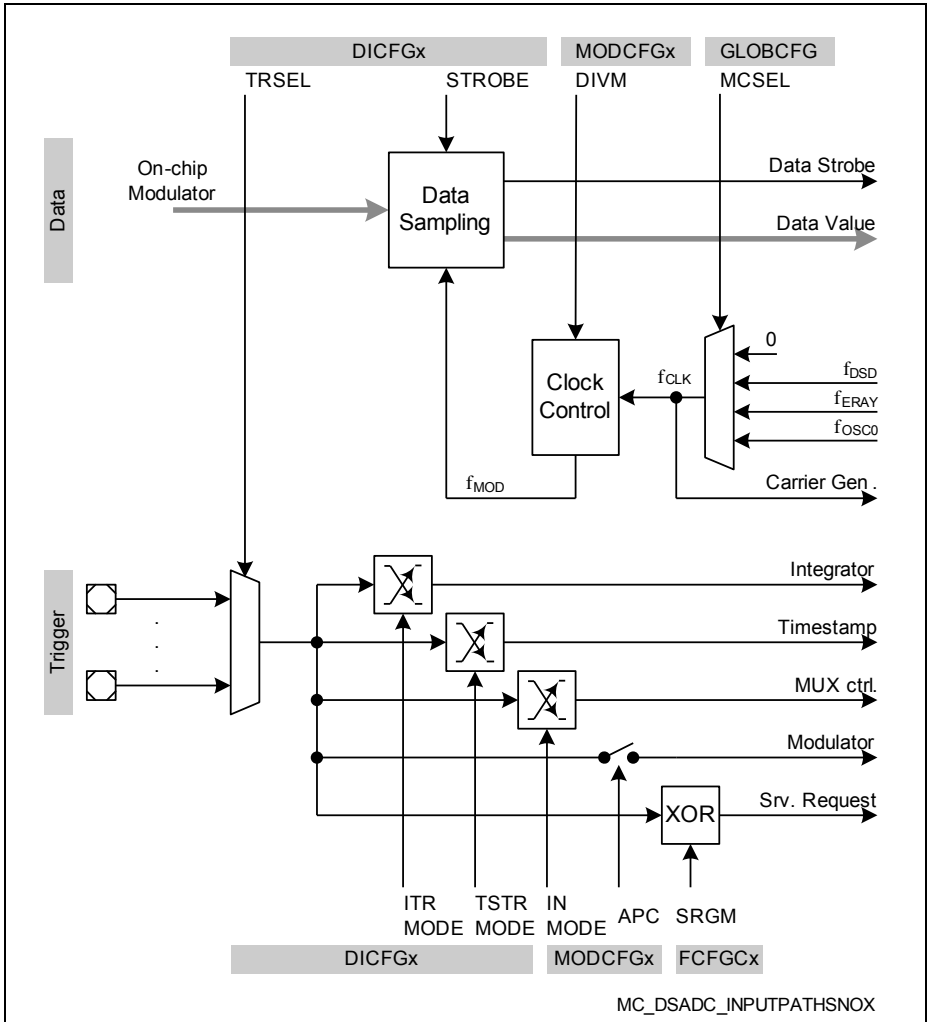


Figure 30-4 Input Path Summary

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.3.1 Modulator Clock Selection and Generation

The modulator clock signal is generated on-chip by a programmable prescaler. The on-chip clock signal is enabled by bitfield MCSEL.

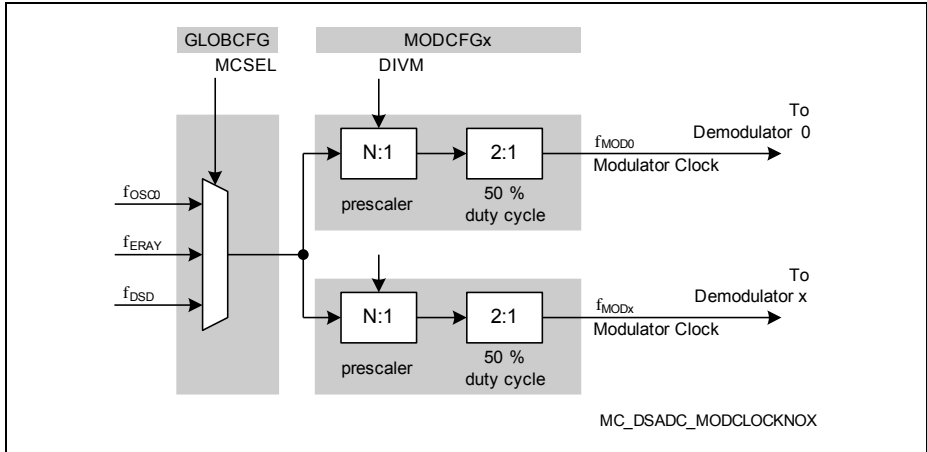


Figure 30-5 Modulator Clock Configuration

The selected clock source (f_{CLK}) is synchronized to the module clock. A configurable edge detector selects the clock signal edges that generate the data strobes which read the next input data and trigger the demodulator (see [Figure 30-6](#)).

Bitfield CSRC in register **DICFGx** ($x = 0, 2, 3$) selects the clock source for each channel. Bitfield STROBE selects the clocking mode, i.e. the clock edges that put a new input data sample into the filter chain.

Delta-Sigma Analog-to-Digital Converter (DSADC)

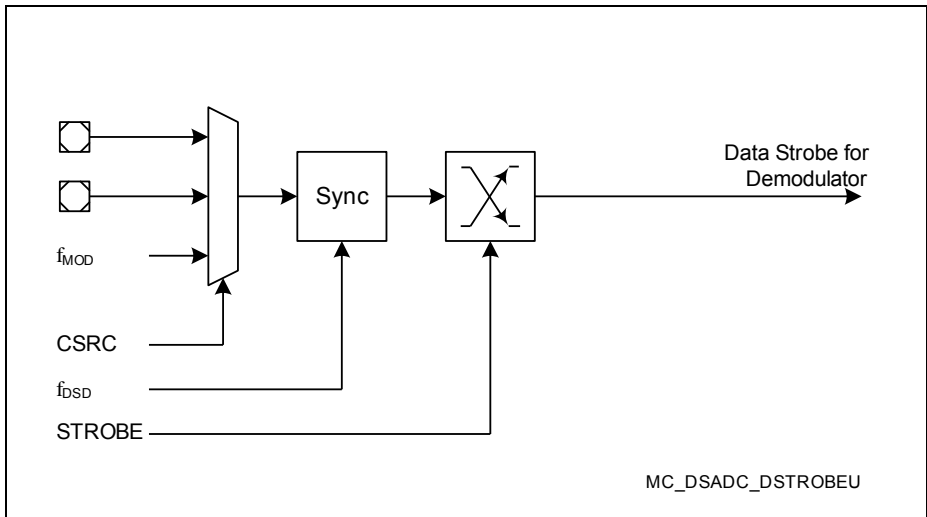


Figure 30-6 Demodulator Data Strobe Selection

Note: The modulator frequency can be selected for each channel separately. To avoid errors caused by cross-coupling, it is recommended to run the on-chip modulator at the same modulator frequency. Using frequencies with a ratio of 2:1 is a compromise.

30.3.2 Input Data Selection

The data stream is generated by the on-chip modulator and is converted to the CIC filter's input format.

Delta-Sigma Analog-to-Digital Converter (DSADC)

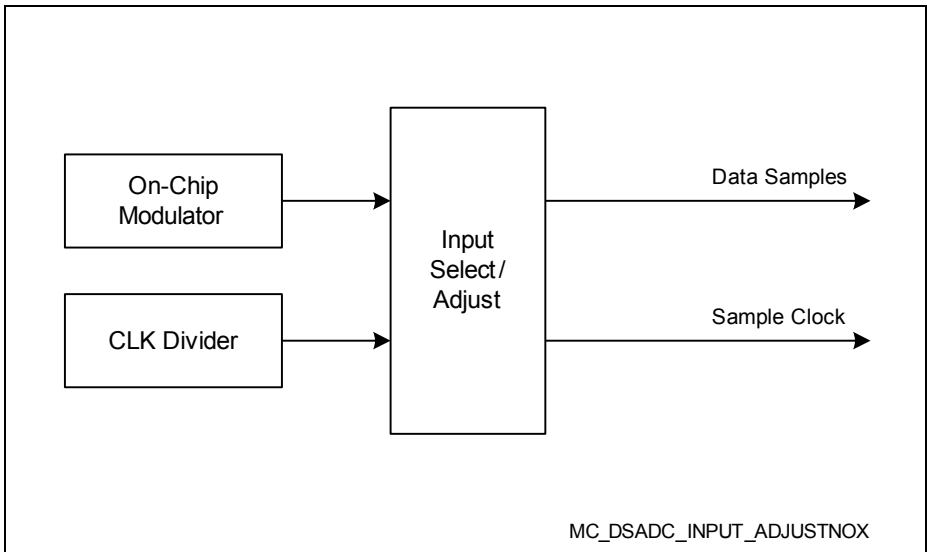


Figure 30-7 Input Control Unit

All options are configured by the demodulator input configuration register **DICFGx (x = 0, 2, 3)**.

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.3.3 On-Chip Modulator

The on-chip modulator operates at a frequency range of 10 MHz to 20 MHz. This sample frequency is derived from the actual module frequency (see [“Modulator Clock Selection and Generation” on Page 30-24](#)).

The analog input signal is fed to the sample stage via input resistors and a differential input buffer. This buffer can also operate in single-ended mode with the input signal fed to one input and the other input internally grounded.

Analog input multiplexers connect the input buffer to different pins. Section [“Product-Specific Configuration” on Page 30-85](#) details the available inputs. The input multiplexers are controlled by register [MODCFGx \(x = 0, 2, 3\)](#).

The gain factor of the input stage can be programmed to 1, 2, 4, 8, 16. Three common mode voltages can be selected.

Note: The buffer's feedback path mirrors the selected common mode voltage permanently to the analog signal source through the input resistors.

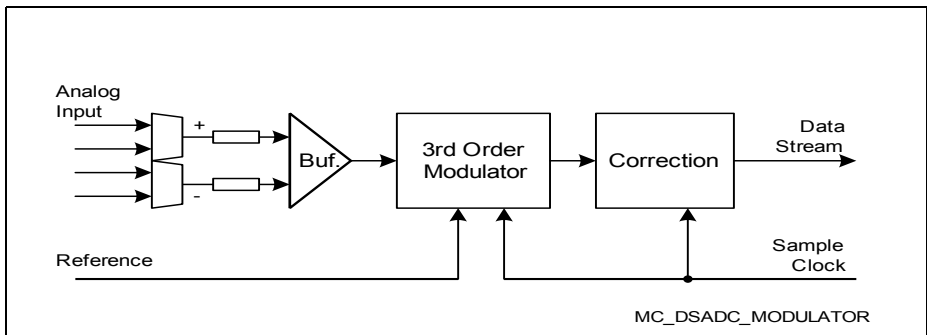


Figure 30-8 Structure of the On-Chip Modulator

Offset correction and gain correction improve the performance of the modulator. Offset measurement mode can be enabled (both inputs to V_{CM} via bitfields INCFGP and INCFGN in register [MODCFGx \(x = 0, 2, 3\)](#)) during the initialization phase and also during operation, e.g. to compensate temperature drift. The measured offset value can be written to register OFFMx for automatic offset correction. Gain correction is accomplished by using a factory-provided compensation value (see [Section 30.3.7](#)).

Operating Modes

The on-chip modulator can run in standard performance mode or in high performance mode. High performance mode achieves the maximum specified SNR, while standard performance mode reduces the power consumption by approx. 2 mA per channel.

Delta-Sigma Analog-to-Digital Converter (DSADC)

The on-chip passive analog anti-alias filter can be enabled to precondition the input signals. The analog anti-alias filter has a typical 3-dB cutoff frequency of approx. 500 kHz.

Both functions are controlled by bitfield MCFG in register **MODCFGx (x = 0, 2, 3)**.

Power Reduction via External Run Control

To reduce the amount of power consumed by the modulator, the modulator can be controlled (enabled/disabled) by an external gate signal (see bit APC in register **MODCFGx (x = 0, 2, 3)**).

Whenever the modulator is enabled, either initially during initialization or repeatedly via automatic power control, it needs a settling time of approx. 1 μ s before delivering proper data.

Make sure to include this settling time in the on-off cycles of the gate signal.

Internal Dithering

Static or low-frequency input signals can generate so-called idle tones which could degrade the SNR. The internal dithering avoids this phenomenon. Internal dithering is enabled by default, see bits DIIEN in register **ICCFGx (x = 0, 2, 3)**.

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.3.4 Input Path Control

The input for the DSADC is fed through several stages before being evaluated and filtered. Registers MODCFGx and DICFGx select the available options for these signal stages.

The following features can be configured:

- Signal input pin selection
- Configuration of input buffer and on-chip modulator
- Modulator clock source and/or frequency
- Trigger input pin selection

Signal Input Pin Selection

Some modulator inputs provide analog input multiplexers which connect them to several input pins. These input multiplexers can be controlled directly by software or can be switched triggered by an external control signal. Software selects the desired position via bitfield INSEL, bitfield INMUX indicates the actual setting of the input multiplexer. Bitfield INMODE defines the condition for a trigger event, bitfield INMAC selects the way in which the multiplexer is controlled:

- Software control:
Control bitfield INMUX follows bitfield INSEL, software directly selects the intended input pins.
- Preset mode:
The (software-written) value in bitfield INSEL is copied to bitfield INMUX upon a trigger event. Software can preselect the next intended multiplexer setting which then becomes active when the next trigger event occurs.
- Single-step mode:
Bitfield INMUX is decremented upon each trigger event. If $INMUX = 00_B$ when the trigger occurs, it is loaded from bitfield INSEL instead. In this mode, a predefined sequence of input pins can be scanned automatically.

The trigger signal itself is selected by bitfield TRSEL in register **DICFGx (x = 0, 2, 3)**.

Note: Not all channels provide input multiplexers. Also, the width of the input multiplexers may differ from channel to channel.

*Refer to **“Product-Specific Configuration” on Page 30-85** for a description of the available input paths.*

Delta-Sigma Analog-to-Digital Converter (DSADC)

The modulator configuration register selects the operation mode of the on-chip modulator:

- Input line clamping
- Gain factor of input signal path
- Modulator operating mode
- Divider factor for modulator clock

MODCFGx (x = 0, 2, 3)
Modulator Configuration Register x
 $(x * 0100_H + 0100_H)$
Reset Value: 0800 0005_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
M WC	0	APC	GC EN	MCFG	CMVS	D WC	0	0	0	DIVM					
w	r	rw	rw	rw	rw	w	r	r	r	rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INC WC	IN MAC	INMODE	INMUX	INSEL	GAINSEL			INCFGN	INCFGP						
w	rw	rw	rh	rw	rw			rw	rw						

Field	Bits	Type	Description
INCFGP	[1:0]	rw	Configuration of Positive Input Line Defines the internal connection of the positive buffer input. 00 _B Input pin 01 _B Reference voltage V_{AREF} 10 _B Common mode voltage V_{CM} 11 _B Reference ground
INCFGN	[3:2]	rw	Configuration of Negative Input Line Defines the internal connection of the negative buffer input. 00 _B Input pin 01 _B Reference voltage V_{AREF} 10 _B Common mode voltage V_{CM} 11 _B Reference ground

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
GAINSEL	[7:4]	rw	Gain Select of Analog Input Path 0000 _B Gain factor 1 0001 _B Gain factor 2 0010 _B Gain factor 4 0011 _B Gain factor 8 0100 _B Gain factor 16 Other combinations are reserved.
INSEL	[9:8]	rw	Input Pin Selection Defines the initial or permanent setting for the input multiplexer (bitfield INMUX) depending on the selected operating mode (bitfield INMODE).
INMUX	[11:10]	rh	Input Multiplexer Setting Indicates the current setting of the input multiplexer connecting the input pins to the buffer inputs. 00 _B Input pin position A 01 _B Input pin position B 10 _B Input pin position C 11 _B Input pin position D The available input pins are listed in Section 30.11.3
INMODE	[13:12]	rw	Input Multiplexer Control Mode Defines the condition for a trigger event to control the input multiplexer. 00 _B Software control (INMUX follows INSEL) 01 _B Trigger event upon a falling edge 10 _B Trigger event upon a rising edge 11 _B Trigger event upon any edge Bitfield INMAC selects the action upon a trigger event.
INMAC	14	rw	Input Multiplexer Action Control Defines the mechanism by which the input multiplexer is controlled. 0 _B Preset mode (load INMUX upon a trigger) 1 _B Single-step mode (decrement INMUX upon a trigger, wrap around to <INSEL>)
INCWC	15	w	Write Control for Input Parameters 0 _B No write access to input parameters 1 _B Bitfields INCFGP, INCFGN, GAINSEL, INSEL, INMODE, INMAC can be written

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
DIVM	[19:16]	rw	Divider Factor for Modulator Clock Defines the operation frequency for the modulator, derived from the selected internal clock source. ¹⁾ $0_H \quad f_{MOD} = f_{CLK} / 2$ $1_H \quad f_{MOD} = f_{CLK} / 4$ $2_H \quad f_{MOD} = f_{CLK} / 6$... $F_H \quad f_{MOD} = f_{CLK} / 32$
0	[22:20]	r	Reserved, write 0, read as 0
DWC	23	w	Write Control for Divider Factor 0_B No write access to divider factor 1_B Bitfield DIVM can be written
CMVS	[25:24]	rw	Common Mode Voltage Selection Defines the common mode voltage for the input buffers of a twin-modulator ²⁾ . $00_B \quad V_{CM} = V_{AREF} / 3.03$ (1.65 V/1.09 V for $V_{AREF} = 5.0 \text{ V}/3.3 \text{ V}$) $01_B \quad V_{CM} = V_{AREF} / 2.27$ (2.2 V/1.45 V for $V_{AREF} = 5.0 \text{ V}/3.3 \text{ V}$) $10_B \quad V_{CM} = V_{AREF} / 2.0$ (2.5 V/1.65 V for $V_{AREF} = 5.0 \text{ V}/3.3 \text{ V}$) 11_B Reserved <i>Note: For most applications factor 2.0 (CMVS = 10_B) will be the optimum (see Section 30.3.5).</i>
MCFG	[27:26]	rw	Modulator Configuration Selects additional modulator features (see “Operating Modes” on Page 30-27). 00_B High-performance mode active 01_B High-performance mode active and anti-alias filter enabled 10_B Standard-performance mode active 11_B Standard-performance mode active and anti-alias filter enabled
GCEN	28	rw	Gain Calibration Enable 0_B Normal operation 1_B On-chip gain calibration mode enabled <i>Note: See also Section 30.3.7.</i>

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
APC	29	rw	Automatic Power Control 0_B Off: Modulator active while its associated bit MxRUN is set 1_B On: Modulator active while MxRUN is set and the gate signal (selected trigger) is active high
0	30	r	Reserved, write 0, read as 0
MWC	31	w	Write Control for Mode Selection 0_B No write access to mode selection 1_B Bitfields CMVS, MCFG, GCEN, APC can be written

- 1) The limit values for f_{MOD} must not be exceeded when selecting the module input frequency and the prescaler setting. f_{MOD} is the clock signal selected by MCSEL.
- 2) The common mode voltage is selected for all modulators by bitfield CMVS in register MODCFG0.

Delta-Sigma Analog-to-Digital Converter (DSADC)

The demodulator input configuration register selects input signal sources for each channel:

- Source of data stream
- Trigger signal source and mode
- Sample clock source
- Data strobe generation mode

DICFGx (x = 0, 2, 3)
Demodulator Input Configuration Register x
(x * 0100_H + 0108_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SC WC	0	0	0	0	0	0	0	STROBE			CSRC				
w	r	r	r	r	r	r	r	rw			rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TR WC	TRSEL		TSTR MODE		ITR MODE		DS WC	0	0	0	DSRC				
w	rw		rw		rw		w	r	r	r	rw				

Field	Bits	Type	Description
DSRC	[3:0]	rw	Input Data Source Select 0000 _B On-chip modulator, standalone (3rd order) Other combinations are reserved.
0	[6:4]	r	Reserved, write 0, read as 0
DSWC	7	w	Write Control for Data Selection 0 _B No write access to data parameters 1 _B Bitfield DSRC can be written
ITRMODE	[9:8]	rw	Integrator Trigger Mode¹⁾ 00 _B No integration trigger, integrator bypassed, INTEN = 0 all the time 01 _B Trigger event upon a falling edge 10 _B Trigger event upon a rising edge 11 _B No trigger, integrator active all the time, INTEN = 1 all the time <i>Note: To ensure proper operation, ensure that bitfield ITRMODE is 00_B before selecting any other trigger mode.</i>

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
TSTRMODE	[11:10]	rw	Timestamp Trigger Mode²⁾ 00 _B No timestamp trigger 01 _B Trigger event upon a falling edge 10 _B Trigger event upon a rising edge 11 _B Trigger event upon each edge
TRSEL	[14:12]	rw	Trigger Select³⁾ Selects an input for the trigger signal used for the following features (see also Figure 30-4): integrator control, timestamp, multiplexer control, modulator control (APC), service request gating. The connected trigger input signals are listed in Section 30.11.4
TRWC	15	w	Write Control for Trigger Parameters 0 _B No write access to trigger parameters 1 _B Bitfields TRSEL, TSTRMODE, ITRMODE can be written
CSRC	[19:16]	rw	Sample Clock Source Select 0000 _B Internal clock Other combinations are reserved.
STROBE	[23:20]	rw	Data Strobe Generation Mode 0000 _B No data strobe 0001 _B Direct clock, a sample trigger is generated at each rising clock edge Other combinations are reserved.
0	[30:24]	r	Reserved, write 0, read as 0
SCWC	31	w	Write Control for Strobe/Clock Selection 0 _B No write access to strobe/clock parameters 1 _B Bitfields STROBE, CSRC can be written

1) The integration trigger mode controls bit INTEN in register **IWCTR_x** (**x = 0, 2, 3**) and hence the operation of the integrator:

Bit INTEN is set when the selected trigger signal transition occurs (INTEN remains 1 while ITRMODE = 11_B). Bit INTEN is cleared when the inverse trigger signal transition occurs or when the repeat counter expires (INTEN remains 0 while ITRMODE = 00_B). Refer to [“Integrator Stage” on Page 30-54](#) for more information.

2) The timestamp trigger mode controls capturing the timestamp information to register **TSTMP_x** (**x = 0, 2, 3**).

3) To avoid unintended triggers, select the trigger source first before enabling the corresponding function.

Delta-Sigma Analog-to-Digital Converter (DSADC)
30.3.5 Common Mode Voltage

The common mode voltage V_{CM} is derived from the analog reference voltage V_{AREF} . The relational factor, i.e. the V_{CM} level, is selected by bitfield CMVS in register **MODCFGx (x = 0, 2, 3)**.

The optimum choice depends on the application. It is recommended to select $V_{CM} = V_{AREF}/2.27$ to achieve best results for SNR/SFDR. For passive signal sources, the maximum signal amplitude is limited then.

Passive Signal Sources

For passive signal sources the common mode voltage defines the center of the signal range. To avoid non-linearities at the top of the input voltage range, a lower common mode voltage can be selected. This depends on the range of the actual input signal.

For $V_{DDM} = 3.3\text{ V}$, the modulator is supplied directly by the external voltage (bit LOSUP must be set, see **Section 30.2.4**).

For $V_{DDM} = 5.0\text{ V}$, the modulator is supplied by an on-chip regulator.

Active Signal Sources

If the differential signal source itself generates the common mode voltage, there might be a difference between the external and the internal common mode voltage. This voltage difference leads to currents through the external and internal circuitry. An asymmetry of this circuitry can generate an additional gain error.

To limit this effect, select the common mode voltage according to the signal source. The maximum input voltage range depends on the selected common mode voltage.

For single-ended signals the common mode voltage generates an additional error that can be compensated by software (see **“Correction for Single-Ended Operation” on Page 30-45**).

30.3.6 Common Mode Hold Voltage

The input channel's common mode voltage is connected to an input pin while this pin is used for the conversions. It is disconnected from the common mode voltage when

- the modulator is inactive, e.g. when using the automatic power control feature
- the input path is disconnected during calibration phases
- the pin is not currently selected via the analog input multiplexer

Pins that are intended to be used for the DSADC can be connected to the common mode hold voltage V_{CMH} under these circumstances. The connection to the common mode hold voltage ensures that the sensor input does not float away while the respective input line is disconnected.

Registers GLOBVCMH0/GLOBVCMH1/GLOBVCMH2 select the connection to V_{CMH} for each of the possible input pins. Two nibbles are assigned to each channel (positive and

Delta-Sigma Analog-to-Digital Converter (DSADC)

negative inputs) and contain the control bits for each possible input of the respective channel.

Note: Enabling the common mode hold voltage source (GLOBVCMH2.ON = 1) eliminates crosscoupling between inputs of analog input multiplexers.

Bitfield GLOBVCMH2.VCMHS selects the level of the applied common mode hold voltage. The correct setting depends on the available supply voltage and the intended input voltage.

Note: The available channels provide different numbers of input pins. For a summary, refer to “Product-Specific Configuration” on Page 30-85. Only those bits in GLOBVCMHn are valid that correspond to an existing input pin.

A pin is connected to the common mode hold voltage when:

- The common mode hold voltage is selected for this pin (GLOBVCMHn.INxPVCy = 1 or GLOBVCMHn.INxNVCy = 1)
- *AND*
 - The respective pin is not currently connected to the channel (MODCFGx.INMUX)
 - *OR* the respective modulator is disabled (in powerdown)

A pin is disconnected from the common mode hold voltage in other cases.

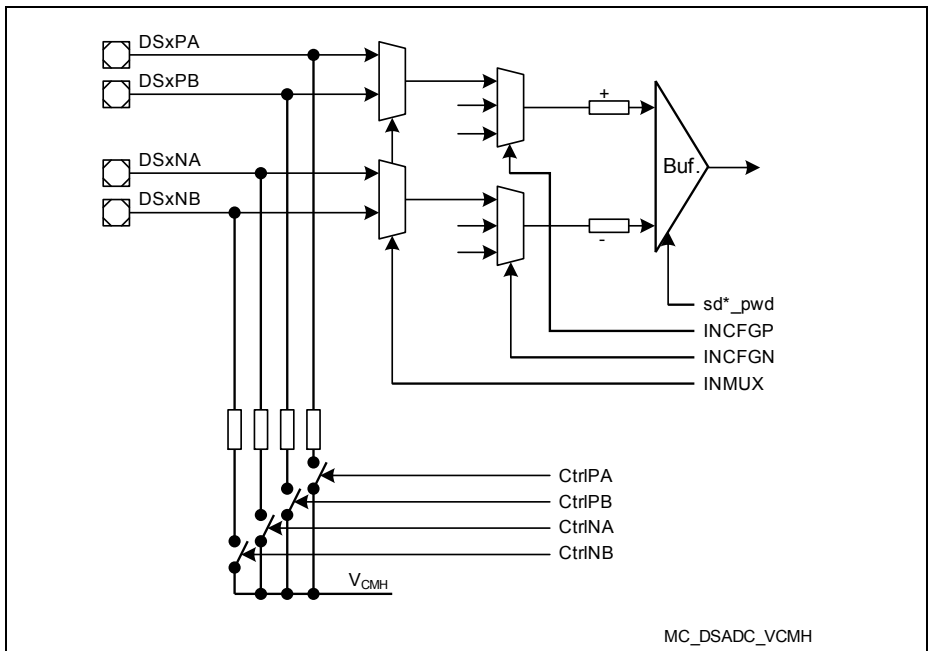


Figure 30-9 Control of VCMH Switches

Delta-Sigma Analog-to-Digital Converter (DSADC)

The common mode hold voltage registers select which input pins are connected to the common mode hold voltage while they are not used for conversions, and which common mode hold voltage is generated:

GLOBVCMH0
Common Mode Hold Voltage Register 0
(00B0_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IN3N VC3	IN3N VC2	IN3N VC1	IN3N VC0	IN3P VC3	IN3P VC2	IN3P VC1	IN3P VC0	0	0	0	IN2N VC0	0	0	0	IN2P VC0
rw	rw	rw	rw	rw	rw	rw	rw	r	r	r	rw	r	r	r	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	IN0N VC1	IN0N VC0	0	0	IN0P VC1	IN0P VC0
r	r	r	r	r	r	r	r	r	r	rw	rw	r	r	rw	rw

Field	Bits	Type	Description
IN0PVC0, IN0PVC1	0, 1	rw	Voltage Control of Positive Inputs 0-1 of CH0 Defines the connection of the respective positive input y to the common mode hold voltage. y indicates the input of the analog multiplexers. 0 _B No connection to common mode hold voltage 1 _B This pin is connected to the common mode hold voltage V_{CMH} while it is not selected for conversions
0	[3:2]	r	Reserved, write 0, read as 0
IN0NVC0, IN0NVC1	4, 5	rw	Voltage Control of Negative Inputs 0-1 of CH0 Defines the connection of the respective negative input y to the common mode hold voltage. y indicates the input of the analog multiplexers. 0 _B No connection to common mode hold voltage 1 _B This pin is connected to the common mode hold voltage V_{CMH} while it is not selected for conversions
0	[15:6]	r	Reserved, write 0, read as 0
IN2PVC0	16	rw	Voltage Control of Positive Input 0 of CH2 Functionality according to bitfield IN0PVC0.

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
0	[19:17]	r	Reserved, write 0, read as 0
IN2NVC0	20	rw	Voltage Control of Negative Input 0 of CH2 Functionality according to bitfield IN0NVC0.
0	[23:21]	r	Reserved, write 0, read as 0
IN3PVC0, IN3PVC1, IN3PVC2, IN3PVC3	24, 25, 26, 27	rw	Voltage Control of Positive Inputs 0-3 of CH3 Functionality according to bitfield IN0PVC0.
IN3NVC0, IN3NVC1, IN3NVC2, IN3NVC3	28, 29, 30, 31	rw	Voltage Control of Negative Inputs 0-3 of CH3 Functionality according to bitfield IN0NVC0.

GLOBVCMH2
Common Mode Hold Voltage Register 2

 (00B8_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VCMHS	VHON	0	0	0	0	0	0	0	0	0	0	0	0	0	0
rw	rw	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
0	[28:0]	r	Reserved, write 0, read as 0
VHON	29	rw	Common Mode Hold Voltage On 0_B V_{CMH} is generated as long as one of bits INxPVCy or INxNVCy is set 1_B V_{CMH} is generated all the time

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
VCMHS	[31:30]	rw	<p>Common Mode Hold Voltage Selection</p> <p>00_B $V_{CMH} = V_{DDM} / 2.0$ (2.5 V/1.65 V for $V_{DDM} = 5.0$ V/3.3 V)</p> <p>01_B $V_{CMH} = V_{DDM} / 2.27$ (2.2 V/1.45 V for $V_{DDM} = 5.0$ V/3.3 V)</p> <p>10_B $V_{CMH} = V_{DDM} / 3.03$ (1.65 V/1.09 V for $V_{DDM} = 5.0$ V/3.3 V)</p> <p>11_B Reserved</p> <p><i>Note: Select the common mode hold voltage according to the selected standard common mode voltage (MODCFGx (x = 0, 2, 3).CMVS).</i></p>

Delta-Sigma Analog-to-Digital Converter (DSADC)
30.3.7 Calibration Support

The performance of the DSADC can be improved by applying some calibration techniques. Several possibilities are provided:

- Offset compensation can automatically remove the DC component of an AC signal. This is done by the offset compensation filter.
- Offset calibration compensates shifted result values by using results from zero-input measurements.
- Gain calibration uses a calibration factor which is obtained from actual measurements and stored device-specific correction values. The application can multiply the result values by this factor.
- Gain adjustment compensates the variation of the input resistors with stored device-specific correction values.

Note: To compensate temperature effects it is recommended to repeat the calibration sequence when the device temperature has changed by approximately 20 °C.

Offset Calibration Support

The input structures of a channel can shift the input voltage which causes an offset error. This can be compensated by determining the shift by measuring a zero-input. This is done by connecting both input lines to the same signal, e.g. the internal common mode voltage:

- Select the same voltage for both differential input lines (MODCFGx.INCFGP = MODCFGx.INCFGN)
 - For differential operation, select common mode voltage (INCFGP/N = 10_B)
 - For single-ended operation, select reference ground (INCFGP/N = 11_B)
- Calculate the average value from a set of measurements (exclude values that differ more than 10% from the mean value)
- Store the calculated average value in register **OFFMx (x = 0, 2, 3)**

Note: Disable the offset compensation filter for this measurement (FCFGMx.OCEN = 0).

Gain Calibration Support

The overall accuracy can be improved by correcting the specific gain error of a channel via software. This is done by multiplying the result values by a specific gain calibration factor.

Gain calibration is based on a high-precision squarewave generator that generates a reference signal ($V_{\text{AREF}} / 10$) for gain factor measurements. Alternatively, the reference voltage V_{AREF} can be directly selected for a full-scale calibration with an analog gain factor 1.

The digital gain calibration factor (DGCF) for a channel (CHNr) is calculated from the device-specific square wave amplitudes stored in the user Flash area (ASQ) and the actual square wave amplitude value AM measured in the application:

Delta-Sigma Analog-to-Digital Converter (DSADC)

DGCF = (ASQ / 20) / AM [mV] (ASQ is given in steps of 50 μ V).

The actual digital gain correction factor is determined by executing the following sequence:

- Select the intended operating configuration: modulator frequency, decimation rate, data shifter setting, etc.
- Configure the intended (analog) gain factor (MODCFGx.GAINSEL)
Note: Select factors 1...8 (500 mV \times factor 16 leads to overload)
- Select an internal voltage (e.g. MODCFGx.INCFGP = MODCFGx.INCFGN = 01_B, V_{AREF})
- Enable gain calibration mode (MODCFGx.GCEN = 1)
This disconnects the input pins¹⁾ and connects the channel input to an internal square wave signal source with an amplitude of nominally $V_{AREF} / 10$ (exact voltage stored in UCB_IFX) and a frequency of $f_{MOD} / 8192$.
- Determine the actual amplitude by converting the high level and the low level of the square wave signal. Result = AM.
 - Read a sequence of result values.²⁾
 - Ignore the values close to the signal transitions to exclude the overshoots/undershoots caused by the Gibbs phenomenon (see [Figure 30-10](#)).
 - Average the high values and the low values to eliminate noise from this measurement, the difference is the measured amplitude value AM.
- Compute the digital gain calibration factor by dividing the device-specific amplitude by the converted amplitude value:
DGCF [mV/LSB] = (ASQ / 20) / AM.

Note: ASQ is determined with $V_{AREF} = 5.00$ V. If a different reference voltage is used in the application, the full scale value must be scaled accordingly ($V_{AREF} / 5.00$ V).

The result value range for a full-scale input (e.g. ± 5.0 V for an analog gain factor of 1) is, therefore, $AFS = \pm(V_{AREF} / DGCF) = \pm(5\,000 \times AM \times 20 / ASQ)^3$.

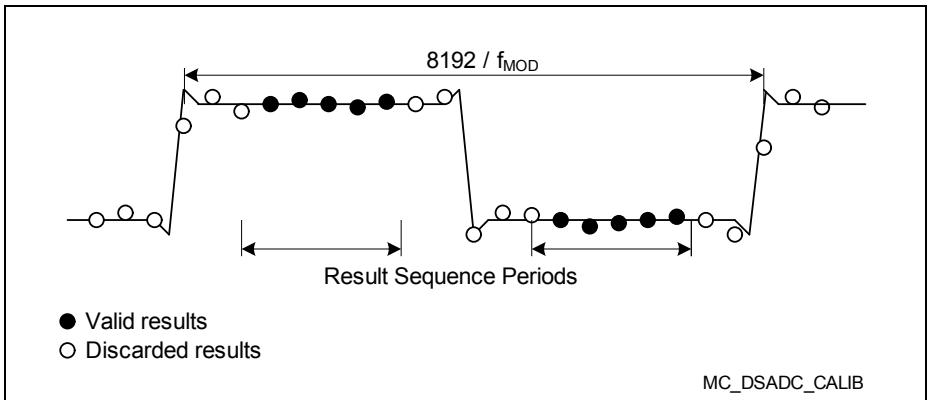
Note: Gain calibration can be executed for both modulators of a twin at the same time. It is recommended to use the same modulator frequency in this case.

1) Selecting a pin instead of the internal voltage (INCFGP/INCFGN = 00_B) will connect the square wave signal to the respective pin. In an application this usually must be avoided.

2) The number of result values per period depends on the configured decimation rate. For example, at a decimation rate of 64, 128 values (8192 / 64) cover one period of the square wave signal.

3) Assuming a reference voltage of $V_{AREF} = 5.00$ V. AFS = full-scale amplitude value.

Delta-Sigma Analog-to-Digital Converter (DSADC)


Figure 30-10 Gain Calibration Measurement
Alternate Gain Calibration

For an analog gain factor of 1, the actual digital gain correction factor is alternatively determined by executing the following sequence:

- Select the intended operating configuration: modulator frequency, decimation rate, data shifter setting, etc.
- Select the internal reference voltages as inputs
(MODCFGx.INCFGP = 01_B: V_{AREF}, MODCFGx.INCFGN = 11_B: V_{AGND})
- Determine the actual full-scale value by converting the reference voltage. Average a sequence of values to eliminate noise from this measurement. Result = AM.¹⁾
- Compute the digital gain calibration factor by dividing the full-scale amplitude by the converted full-scale value:
DGCF [mV/LSB] = V_{AREF} / AM = 5 000 / AM.²⁾

The gain calibration sequence must be executed after each startup.

Using the Calibration Values

The offset calibration value is written into **OFFMx (x = 0, 2, 3)** and is subtracted automatically.

With the gain calibration factor the input amplitude A_{IN} can now be computed from the digital result value A_{IN} as:

$$A_{IN} = A_{IN} \times DGCF.$$

The full-range result value is AFS = ±(V_{AREF} / DGCF).

1) Offset calibration must be done before executing this measurement.

2) Assuming a reference voltage of V_{AREF} = 5.00 V.

Delta-Sigma Analog-to-Digital Converter (DSADC)
Gain Adjustment

The input resistors of a channel can cause an additional gain error due to the variation of their values. The actual value of the input resistors is measured during production test. The device-specific 16-bit gain adjustment values (RIN) are stored in the user Flash area (UCB_IFX) at offset $84_H + 8 \times \text{CHNr}$. The respective value of R_{IN} ($R_{IN} [\text{kOhm}] = \text{RIN} / 100$) can then be used together with the sensor impedance to calculate more accurate results.

The actual input resistance depends on the chip temperature. Therefore, RIN calibration values in the user Flash are stored for two temperatures (HOT and COLD). To increase the precision for this parameter, use these values in the following way:

- Compute the linear interpolation factor FLIN:

$$\text{FLIN} = (\text{RIN_COLD} - \text{RIN_HOT}) / (T_{\text{HOT}} - T_{\text{COLD}})$$
- Compute the temperature deviation DT:

$$\text{DT} = T_{\text{HOT}} - T_{\text{ACTUAL}}$$
- Compute the actual resistance value R_{IN} :

$$R_{IN} = (\text{RIN_HOT} + \text{DT} \times \text{FLIN} \times 0.1 + \text{DT}^2 \times \text{FLIN} \times 0.005) / 100 [\text{kOhm}]$$

The temperature during the calibration sequence (T_{ACTUAL}) can be measured via the on-chip die temperature sensor (DTS) or can be estimated according to a known operating condition of the application.

Note: These values are generated at $T_{\text{COLD}} = -40^\circ\text{C}$ and $T_{\text{HOT}} = +125^\circ\text{C}$.

Delta-Sigma Analog-to-Digital Converter (DSADC)

Correction for Single-Ended Operation

The built-in calibration support is optimized for differential operation. Single-ended operation incurs an asymmetry caused by the external circuitry (only attached to one input pin, see R_{EXT} in [Figure 30-11](#)). An additional compensation factor can be calculated from the different resistances present in the analog input path:

R_{FL} = Resistance of the feedback loop

R_{CM} = Resistance of the common mode voltage path

R_{AIN} = Resistance of the analog input line

R_{EXT} = Resistance of the external circuitry.

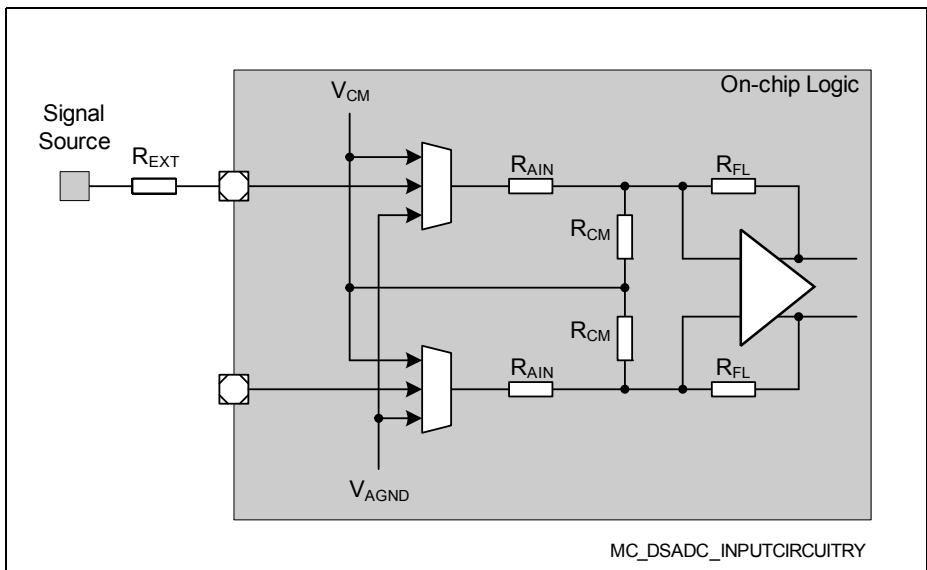


Figure 30-11 Overview of the Input Circuitry of a Channel

Note: This description assumes the sensor signal to be connected to the positive input and the negative input to be grounded internally.

General Algorithm for Resistances of $R_{EXT} \leq 10$ kOhm:

- Calculate auxiliary resistance values:

$$\text{Equivalent input path resistance: } R_{Aequ} = R_{AIN} + R_{EXT} / 2$$

$$\text{Equivalent common mode resistance: } R_{Cequ} = R_{AIN} \times 0.110553$$

- Calculate auxiliary common mode voltage values:

$$\text{Equivalent for low input: } V_{CML} = V_{CM} \times R_{AIN} / (R_{AIN} + R_{Cequ}) = V_{CM} \times 0.900452$$

$$\text{Equivalent for high input: } V_{CMH} = V_{CM} + (V_{AREF} / 2 - V_{CM}) \times R_{Cequ} / (R_{Cequ} + R_{Aequ})$$

Delta-Sigma Analog-to-Digital Converter (DSADC)

- Determine the additional offset value based on the full-scale value

$$\text{OFFS} = \text{AFS} \times (R_{\text{EXT}} / R_{\text{Aequ}}) \times (V_{\text{CML}} / V_{\text{AREF}})$$
 This additional offset value can be added to the value in register OFFMx.
- Determine the additional gain factor based on the various resistances

$$\text{GFS} = R_{\text{Aequ}} / \{R_{\text{AIN}} \times [1 - (R_{\text{EXT}} / R_{\text{Aequ}}) \times (0.5 - V_{\text{CMH}}/V_{\text{AREF}} + V_{\text{CML}}/V_{\text{AREF}})]\}$$
 This additional gain factor can be multiplied with the existing factor DGCF

Note: Using on-chip paths (e.g. the internal grounding of an input line) may incur an inaccuracy due to voltage drops over the on-chip paths.

Example for $R_{\text{AIN}} = 131 \text{ k}\Omega$, $R_{\text{EXT}} = 10 \text{ k}\Omega$, $V_{\text{CM}} = V_{\text{AREF}} / 2.27$, $V_{\text{AREF}} = 5.0 \text{ V}$:

- $R_{\text{Aequ}} = R_{\text{AIN}} + R_{\text{EXT}} / 2 = 131 \text{ k}\Omega + 10 \text{ k}\Omega / 2 = 136 \text{ k}\Omega$
- $R_{\text{Cequ}} = R_{\text{AIN}} \times 0.110553 = 131 \text{ k}\Omega \times 0.110553 = 14.48 \text{ k}\Omega$
- $V_{\text{CML}} = V_{\text{CM}} \times 0.900452 = V_{\text{AREF}} \times 0.396675$
- $V_{\text{CMH}} = V_{\text{CM}} + (V_{\text{AREF}} / 2 - V_{\text{CM}}) \times R_{\text{Cequ}} / (R_{\text{Cequ}} + R_{\text{Aequ}})$
 $= V_{\text{AREF}} / 2.27 + (V_{\text{AREF}} / 2 - V_{\text{AREF}} / 2.27) \times R_{\text{Cequ}} / (R_{\text{Cequ}} + R_{\text{Aequ}})$
 $= 2.202643 \text{ V} + 0.297357 \text{ V} \times 14.48 \text{ k}\Omega / 150.48 \text{ k}\Omega$
 $= 2.231256 \text{ V}$
- $\text{OFFS} = \text{AFS} \times (R_{\text{EXT}} / R_{\text{Aequ}}) \times (V_{\text{CML}} / V_{\text{AREF}})$
 $= \text{AFS} \times 0.029167$
- $\text{GFS} = R_{\text{Aequ}} / \{R_{\text{AIN}} \times [1 - (R_{\text{EXT}} / R_{\text{Aequ}}) \times (0.5 - V_{\text{CMH}}/V_{\text{AREF}} + V_{\text{CML}}/V_{\text{AREF}})]\}$
 $= 136 \text{ k}\Omega / \{131 \text{ k}\Omega \times [1 - 0.073529 \times (0.5 - 0.446251 + 0.396675)]\}$
 $= 136 \text{ k}\Omega / (131 \text{ k}\Omega \times 0.966881)$
 $= 1.073729$

Simplified Algorithm for Resistances of $R_{\text{EXT}} \leq 1 \text{ k}\Omega$:

- Determine the additional offset value based on the full-scale value

$$\text{OFFS} = \text{AFS} \times [R_{\text{EXT}} / (2 \times R_{\text{AIN}} + R_{\text{EXT}})]$$
 This additional offset value can be added to the value in register OFFMx.
- Determine the additional gain factor based on the various resistances

$$\text{GFS} = (R_{\text{AIN}} + R_{\text{EXT}}/2) / [R_{\text{AIN}} \times (1 - R_{\text{EXT}} / 2 \times R_{\text{AIN}})]$$
 This additional gain factor can be multiplied with the existing factor DGCF

Example for $R_{\text{AIN}} = 131 \text{ k}\Omega$, $R_{\text{EXT}} = 1 \text{ k}\Omega$, $V_{\text{CM}} = V_{\text{AREF}} / 2.27$, $V_{\text{AREF}} = 5.0 \text{ V}$:

- $\text{OFFS} = \text{AFS} \times [R_{\text{EXT}} / (2 \times R_{\text{AIN}} + R_{\text{EXT}})]$
 $= \text{AFS} \times 0.003802$
- $\text{GFS} = (R_{\text{AIN}} + R_{\text{EXT}}/2) / [R_{\text{AIN}} \times (1 - R_{\text{EXT}} / 2 \times R_{\text{AIN}})]$
 $= 131.5 \text{ k}\Omega / (131 \text{ k}\Omega \times 0.996183)$
 $= 1.007266$

Delta-Sigma Analog-to-Digital Converter (DSADC)
Range Adaptation

The scaling factor to normalize the result values according to a given format (e.g. $\pm 32\,767$ or 1Q15) is then the relation of the format's limit value and the full scale result (e.g. the scaling factor is $32\,767 / AFS$).

The above calculations result in a gain calibration factor giving the input voltage in millivolts: DGCF [mV/LSB]. If another unit is required by the application, the range adaptation factor can be adjusted accordingly.

Stored Calibration Values

The device-specific calibration values are stored in the User Configuration Block (UCB_IFX) within the on-chip Flash memory. The area assigned to the DSADC begins at offset 80_H and provides records of 2 words per channel. The channel-specific records, therefore, can be read from offset $80_H + 8 \times CHNr$. The table below shows the structure of a record.

Table 30-3 Structure of DSADC Calibration Records

	Halfword 3	Halfword 2	Halfword 1	Halfword 0
Location CH0	86_H	84_H	82_H	80_H
Location CH1	$8E_H$	$8C_H$	$8A_H$	88_H
:	:	:	:	:
Stored value	RIN_HOT	RIN_COLD	Reserved	ASQ

The parameters are stored in the following format:

Resistance = $RIN_* \times 10 \text{ Ohm}$, i.e. $32C8_H = 130 \text{ kOhm}$.

Amplitude = $ASQ \times 50 \mu\text{V}$, i.e. $2710_H = 500 \text{ mV}$.

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.4 Main Filter Chain

The result data words are generated by feeding the input data stream through a chain of filter elements and decimating it by a selectable ratio.

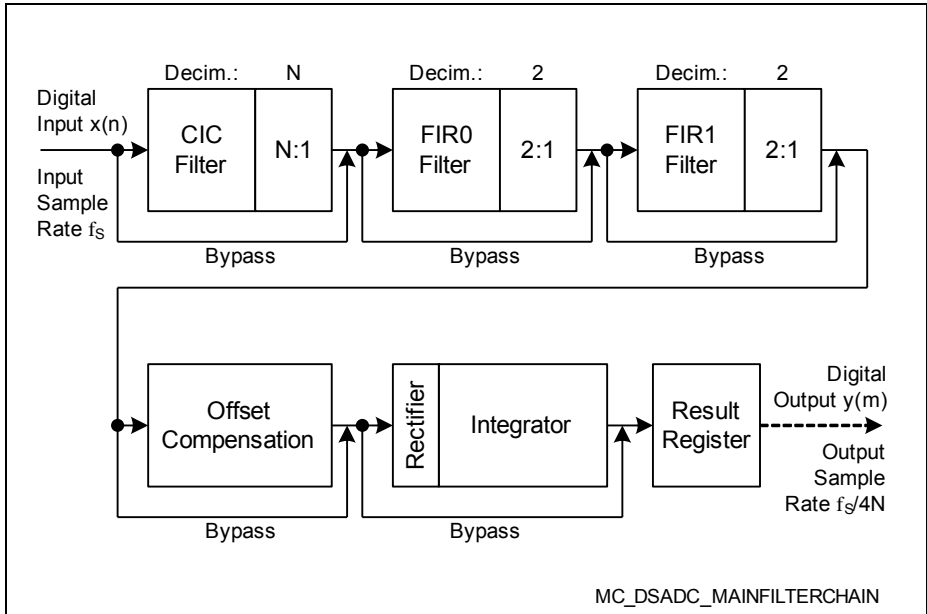


Figure 30-12 Structure of the Main Filter Chain

The elements of the filter can be bypassed, i.e. the filter chain is configurable and its behavior can be adapted to the requirements of the actual application. This comprises the frequency attenuation as well as the total decimation rate.

Note: Only configure or reconfigure filter parameters while the channel is inactive. After the configuration, start the channel by setting the corresponding bit CHxRUN.

30.4.1 CIC Filter

The Cyclic Integrating Comb filter (CIC filter, a.k.a. SINC filter) is a simple but very efficient low-pass filter. Up to three comb filter stages can be cascaded to improve the frequency characteristics. The number of active filter stages can be selected (CIC1, CIC2, CIC3) to find the optimum tradeoff between delay and frequency characteristics.

In addition, a combined mode can be selected (CICF) which represents a compromise between a 2-stage and a 3-stage filter.

Delta-Sigma Analog-to-Digital Converter (DSADC)

To synchronize filters of different channels with fine granularity, the decimation counter starts from an arbitrary start value. This start value can be different from the decimation factor and is loaded only once when the counter is started.

The decimation counter is also restarted (i.e. loaded with the start value) when an integration starts (see [Section 30.4.4](#)).

The decimation factor can be selected in a wide range from 4 to 128.

Table 30-4 Data Shifter Position¹⁾ Dep. on Filter Mode and Decimation

Decimation	CIC1	CIC2	CIC3	CICF
4	17:2	19:4	21:6	20:5
5	18:3 *	20:5 *	22:7 *	21:6 *
6	18:3	21:6 *	23:8 *	22:7 *
7, 8	18:3	21:6	24:9 *	22:7
9, 10	19:4 *	22:7 *	25:10 *	23:8 *
11	19:4	22:7	26:11 *	23:8
12	19:4	23:8 *	26:11	24:9 *
13 - 16	19:4	23:8	27:12 *	24:9
17 - 20	20:5 *	24:9 *	28:13 *	25:10 *
21 - 22	20:5	24:9	29:14 *	25:10
23 - 25	20:5	25:10 *	29:14	26:11 *
26 - 32	20:5	25:10	30:15 *	26:11
33 - 40	21:6 *	26:11 *	31:16 *	27:12 *
41 - 45	21:6	26:11	32:17 *	27:12
46 - 50	21:6	27:12 *	32:17	28:13 *
51 - 64	21:6	27:12	33:18 *	28:13
65 - 80	22:7 *	28:13 *	34:19 *	29:14 *
81 - 90	22:7	28:13	35:20 *	29:14
91 - 101	22:7	29:14 *	35:20	30:15 *
102 - 128	22:7	29:14	36:21 *	30:15

1) * indicates a change. Input data width is 16 bits.

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.4.2 FIR Filters

The FIR filter further attenuates the higher frequency bands that pass the CIC filter. This improves the overall frequency response of the main filter chain.

The FIR filter is realized as two subsequent FIR blocks. Each FIR filter block adds a decimation factor of 2, so the total decimation factor (i.e. the oversampling rate, including the CIC filter) is $4 \times N$.

A shift unit at the output of each block (see [Figure 30-13](#)) selects 16 bits from the filter block's output data. This selected portion (usually the upper 16 bits) is used for further processing. To reduce the clipping in case of known limited data values, the selected data portion can be moved down (1 bit after FIR0, up to 3 bits after FIR1). This effectively increases the gain. Bitfields FSH and DSH in register **FCFGMx (x = 0, 2, 3)** select the position of the data portions.

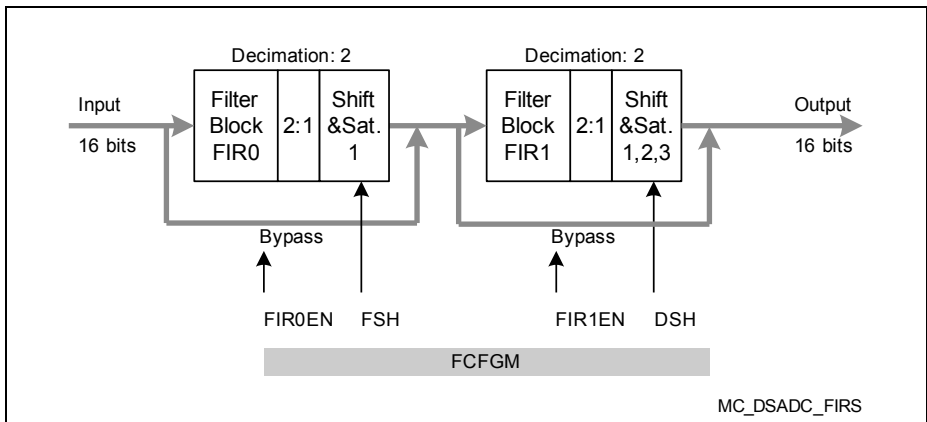


Figure 30-13 FIR Filter Blocks

If an FIR filter block is bypassed, the associated data shifter is also bypassed and its shift control has no effect.

The FIR filters can be described as:

$$y_1(m_1) = \sum_{i=0}^{N_1-1} a_1(i) x(2m_1 - i) \quad y(m) = \sum_{i=0}^{N_2-1} a_2(i) y_1(2m - i)$$

[Figure 30-14](#) shows the FIR filter structure. The first stage FIR has $N_1 = 8$ coefficients, and the second stage has $N_2 = 28$ coefficients.

Delta-Sigma Analog-to-Digital Converter (DSADC)

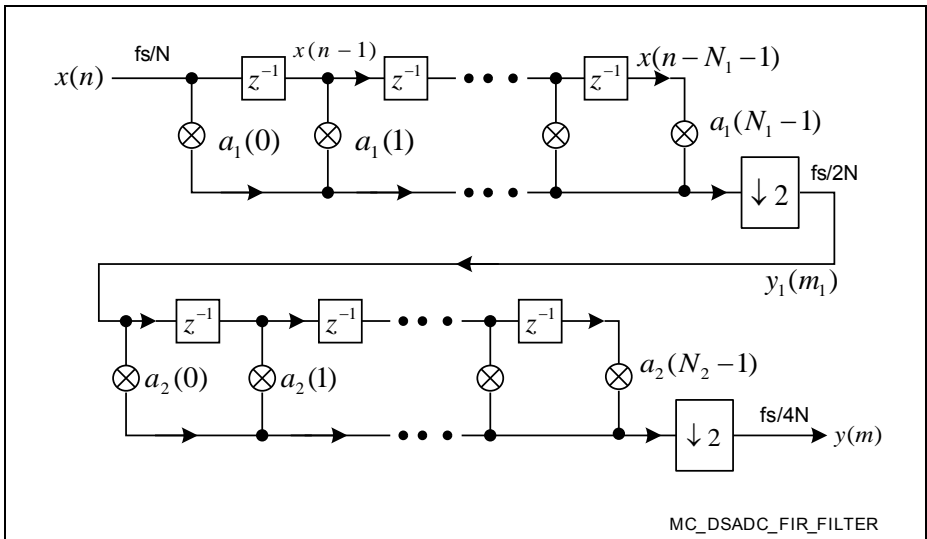


Figure 30-14 FIR Filter Structure

The coefficients of both FIR filters are fixed (see [Table 30-5](#) and [Table 30-6](#)). The filters are symmetric and have linear phase.

Table 30-5 Coefficients of FIR0

a0	a1	a2	a3	a4	a5	a6	a7
-17	-25	83	256	256	83	-25	-17

Table 30-6 Coefficients of FIR1

a0, a27	a1, a26	a2, a25	a3, a24	a4, a23	a5, a22	a6, a21	a7, a20	a8, a19	a9, a18	a10, a17	a11, a16	a12, a15	a13, a14
-3	-2	2	9	-1	-14	-8	19	25	-13	-55	-12	126	256

The resolution of the CIC filter does not represent the resolution of the Delta-Sigma converter. The resolution or the effective number of bits (ENOB) of the Delta-Sigma converter depends on the Signal to Noise Ratio (SNR) of the Delta-Sigma converter including modulator and decimation filter (CIC filter and FIR filter).

The maximum filter output value of a filter depends on the sum of absolute filter coefficients (762 for FIR0, 1090 for FIR1) and the input data width (15-bit + sign), resulting in 24 969 216 for FIR0 and 35 717 120 for FIR1. Therefore, the required data

Delta-Sigma Analog-to-Digital Converter (DSADC)

width of the filters is 26 bits for FIR0 and 27 bits for FIR1. To handle both filters with the same MAC unit, the data width shall be 27 bits + sign.

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.4.3 Offset Compensation

The offset compensation stage is a high-pass filter that removes the DC component of the input signal, in particular in case of a differential input signal.

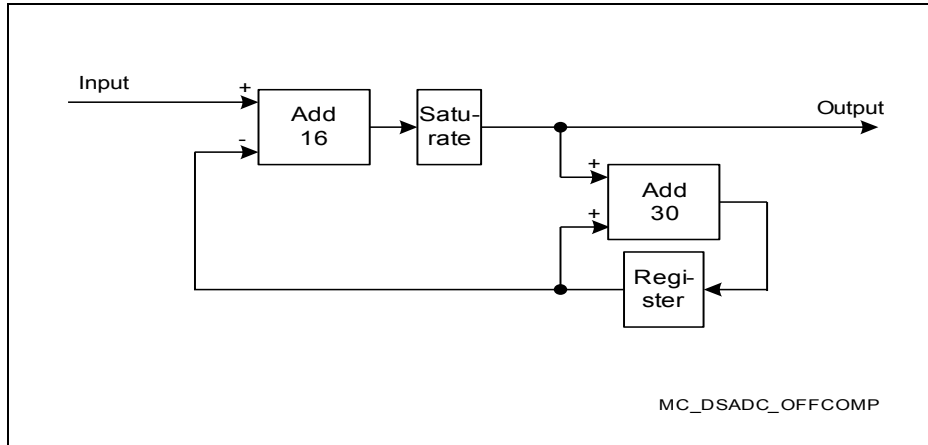


Figure 30-15 Offset Compensation

*Note: When using the offset compensation stage, clear the offset register (**OFFM_x** (**x = 0, 2, 3**)) to avoid over-compensation.*

When running the offset calibration, disable the offset compensation stage to avoid wrong measurements.

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.4.4 Integrator Stage

The integrator integrates the result values generated during the defined integration window by adding a configurable number of values to build the final result value. The integration window can be started triggered by an internal or external signal.

A configurable number of values can automatically be discarded after the trigger before the integration window is started. This positions the integration window exactly into a timeframe where the filter is stable or where the signal to be measured is free of system-generated noise.

Integration can be used to measure currents through shunt resistors at defined positions in the signal waveform. It also can remove the carrier signal component in resolver applications. In this case, the values to be integrated can be rectified to yield the maximum amplitude of the receiver signal. The delay between the carrier signal (generated by the on-chip carrier generator) and the received position signals can be compensated automatically. Also refer to [Section 30.9](#).

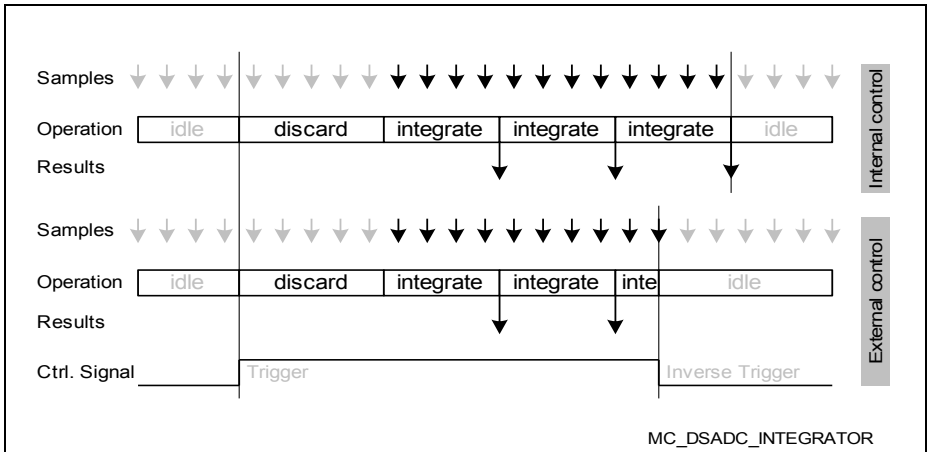


Figure 30-16 Integrator Operation

Note: The integrator can add up to 64 values, while enabled. Its output, therefore, is shifted by 6 bits (1d 64). This leads to reduced result magnitudes when less than 64 values are accumulated.

Delta-Sigma Analog-to-Digital Converter (DSADC)

Starting the Integration Window

The integration window starts when bit INTEN becomes 1:

- **Software-Controlled Mode:**
Select software-controlled integration mode by setting bitfield ITRMODE = 11_B
- **Trigger-Controlled Mode:**
Select trigger-controlled integration mode (ITRMODE = 01_B or 10_B)
and generate the selected trigger edge at the configured trigger input

The external integration trigger signal is selected by bitfield TRSEL in register **DICFGx** (**x = 0, 2, 3**). Bitfield ITRMODE selects the transition of the selected signal to generate the trigger event. Select the trigger source first before enabling it.

Also, the decimation counter of the CIC filter is restarted, i.e. loaded with its start value (see [Section 30.4.1](#)).

When the integration window is started (INTEN becomes 1) the integration counter starts counting. After <NVALDIS> values the counter is reset and the integrator is started (if NVALDIS is zero the integration starts immediately).

Executing Integration Cycles

During an integration cycle <NVALINT+1> input values are integrated. After that, the integration result is stored in the result register and the integrator and the counter are cleared. If bit INTEN = 1 the next integration cycle is started.

Stopping the Integration Window

The integration window stops when bit INTEN becomes 0:

- **Software-Controlled Mode:**
Disable software-controlled integration mode by setting bitfield ITRMODE = 00_B
- **Trigger-Controlled Mode:**
 - Internally controlled end-of-integration (IWS = 0, see upper part of [Figure 30-16](#)):
After <REPVAL+1> integration cycles INTEN is cleared and the integration window closes, after (<NVALINT+1> × <REPVAL+1>) input values
 - Externally controlled end-of-integration (IWS = 1, see lower part of [Figure 30-16](#)):
Generate the selected inverse trigger edge at the configured trigger input (ITRMODE = 01_B or 10_B)
 - Integration may also be stopped by software by setting bitfield ITRMODE = 00_B

Note: When the integration window is closed by an external signal (INTEN cleared by inverse trigger), the integrator and the counter are cleared also.

Delta-Sigma Analog-to-Digital Converter (DSADC)

IWCTR_x (x = 0, 2, 3)
Integration Window Control Register x

$$(x * 0100_H + 0120_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	NVALINT					IWS	0	NVALDIS						
r	r	rw					rw	r	rw						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REPVAL				REPCNT				INT EN	0	NVALCNT					
rw				rh				rh	r	rh					

Field	Bits	Type	Description
NVALCNT	[5:0]	rh	Number of Values Counted Counts the number of values until integration is started (NVALDIS) or completed (NVALINT). NVALCNT counts up from zero.
0	6	r	Reserved, write 0, read as 0
INTEN	7	rh	Integration Enable¹⁾ 0 _B Integration stopped. INTEN is cleared at the end of the integration window, i.e. upon the inverse trigger event transition of the external trigger signal. 1 _B Integration enabled. INTEN is set upon the defined trigger event.
REPCNT	[11:8]	rh	Integration Cycle Counter Counts the number of integration cycles if activated (IWS = 0). This number is selected via bitfield REPVAL.
REPVAL	[15:12]	rw	Number of Integration Cycles Defines the number of integration cycles to be counted by REPCNT if activated (IWS = 0). The number of cycles is <REPVAL+1>.
NVALDIS	[21:16]	rw	Number of Values Discarded Start the integration cycle after NVALDIS values
0	22	r	Reserved, write 0, read as 0

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
IWS	23	rw	Integration Window Size 0_B Internal control: stop integrator after <REPVAL+1> integration cycles 1_B External control: stop integrator when bit INTEN becomes 0 <i>Note: IWS is only relevant in trigger-controlled mode.</i>
NVALINT	[29:24]	rw	Number of Values Integrated Stop the integration cycle after <NVALINT+1> values
0	[31:30]	r	Reserved, write 0, read as 0

1) For the control of bit INTEN, see also bitfield ITRMODE in register **DICFGx (x = 0, 2, 3)**.

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.5 Auxiliary Filter

The parallel auxiliary filter uses a CIC filter for decimation, similar to the one used in the main filter chain. The decimation rate is restricted to 32. This also reduces the filter delay, so the auxiliary filter can be used to supervise the input signal and detect abnormal input values earlier than the main filter chain.

The subsequent comparator provides automatic limit checking by comparing each result to two configurable reference values. The comparator can generate a separate service request.

The two values are defined in the boundary select register and determine the valid result value band if limit checking is enabled.

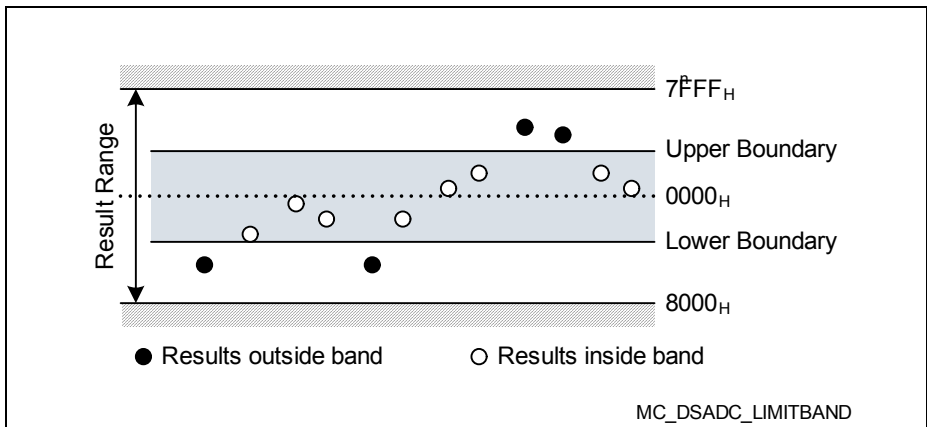


Figure 30-17 Result Monitoring through Limit Checking

A result value is considered inside the defined band when both of the following conditions are true:

- the value is less than or equal to the selected upper boundary
- the value is greater than or equal to the selected lower boundary

The result range can also be divided into two areas:

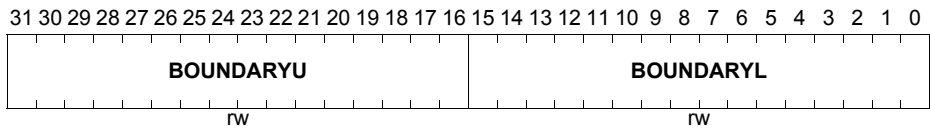
To select the lower part as valid band, set the lower boundary to the minimum value (8000_H) and set the upper boundary to the highest intended value.

To select the upper part as valid band, set the upper boundary to the maximum value ($7FFF_H$) and set the lower boundary to the lowest intended value.

The auxiliary filter can generate two types of output:

- Service requests, optionally restricted by the comparators
- Range signals, indicating when the results are above the upper limit (SAULx) or below the lower limit (SBLx)

Delta-Sigma Analog-to-Digital Converter (DSADC)

BOUNDSELx (x = 0, 2, 3)
Boundary Select Register x ($x * 0100_H + 0128_H$) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
BOUNDARYL	[15:0]	rw	Lower Boundary Value for Limit Checking This (two's complement) value is compared to the results of the parallel filter.
BOUNDARYU	[31:16]	rw	Upper Boundary Value for Limit Checking This (two's complement) value is compared to the results of the parallel filter.

An alarm event can be generated when a new conversion result becomes available. Alarm events can be restricted to result values that are inside or outside a user-defined band (see **Figure 30-17**). This feature supports automatic range monitoring and minimizes the CPU load by issuing service requests only under certain conditions. For example, an input value can be monitored and an alarm indicates a certain threshold.

Note: Limit checking uses the parallel auxiliary filter at a low decimation rate and, therefore, the alarm is generated earlier than the threshold values are seen at the output of the regular filter chain.

Alarm events can also be suppressed completely (see **FCFGAx (x = 0, 2, 3)**).

The range signals are generated independent of service requests.

Signal SAULx is active while the results are above the upper limit, signal SBLLx is active while the results are below the lower limit.

Delta-Sigma Analog-to-Digital Converter (DSADC)

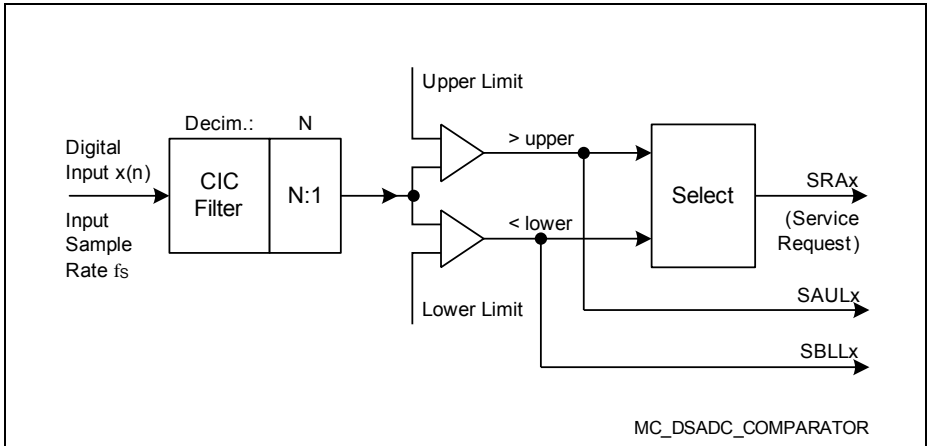


Figure 30-18 Comparator Structure

Delta-Sigma Analog-to-Digital Converter (DSADC)
30.6 Filter Configuration and Control

Both filter chains (main and auxiliary) can be configured for the intended application. The following pages describe the available bitfields in the control registers, [Section 30.6.2](#) provides recommended settings to obtain best results.

Note: The main filter chain, of course, provides more configuration options than the auxiliary filter.

30.6.1 Filter Configuration Options

Both filter paths can be configured according to the requirements of the application. The stages of the main filter chain can be selected.

FCFGMx (x = 0, 2, 3)
Filter Configuration Register x, Main Filter Chain

$$(x * 0100_H + 0110_H)$$

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	FSH	DSH	OC EN	FIR1 EN	FIR0 EN	
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	

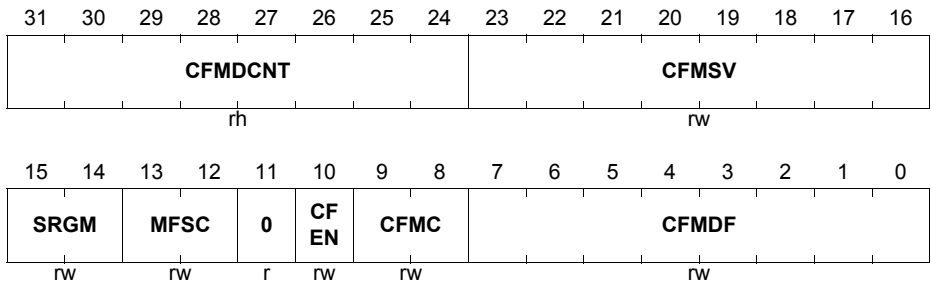
Field	Bits	Type	Description
FIR0EN	0	rw	FIR Filter 0 Enable 0 _B FIR filter 0 disabled and bypassed 1 _B Enable FIR filter 0
FIR1EN	1	rw	FIR Filter 1 Enable 0 _B FIR filter 1 disabled and bypassed 1 _B Enable FIR filter 1
OCEN	2	rw	Offset Compensation Filter Enable 0 _B Offset compensation filter disabled and byp. 1 _B Enable offset compensation filter

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
DSH	[4:3]	rw	Data Shift Control Selects the displacement caused by the data shifter at the FIR filter output (see Section 30.4.2). 00 _B No shift, use full range 01 _B Shift by 1 10 _B Shift by 2 11 _B Shift by 3
FSH	5	rw	FIR Shift Control Selects the displacement caused by the data shifter inbetween the FIR filter blocks (see Section 30.4.2). 0 _B No shift, use full range 1 _B Shift by 1
0	[31:6]	r	Reserved, write 0, read as 0

FCFGCx (x = 0, 2, 3)
Filter Configuration Register x, Main CIC Filter

$$(x * 0100_H + 0114_H)$$

Reset Value: 0000 0000_H


Field	Bits	Type	Description
CFMDF	[7:0]	rw	CIC Filter (Main Chain) Decimation Factor The decimation factor of the Main CIC filter is CFMDF + 1. Valid values are 03 _H to 7F _H (4 to 128).
CFMC	[9:8]	rw	CIC Filter (Main Chain) Configuration 00 _B CIC1 01 _B CIC2 10 _B CIC3 11 _B CICF

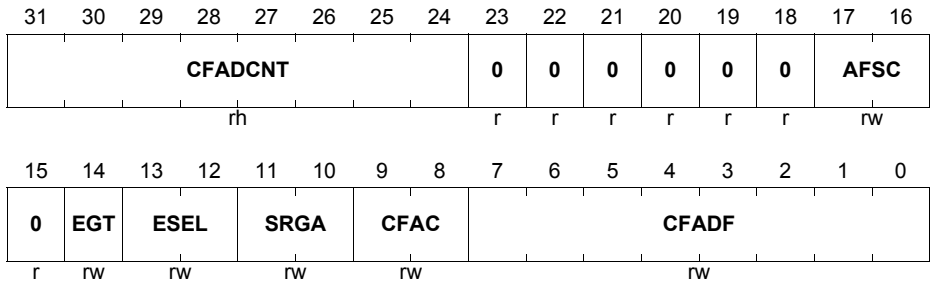
Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
CFEN	10	rw	CIC Filter Enable 0 _B CIC filter disabled and bypassed 1 _B Enable CIC filter
0	11	r	Reserved, write 0, read as 0
MFSC	[13:12]	rw	Main Filter Shift Control Selects the displacement caused by the data shifter at the main CIC filter output. 00 _B No shift, use full range 01 _B Shift by 1 10 _B Shift by 2 11 _B Shift by 3
SRGM	[15:14]	rw	Service Request Generation Main Chain 00 _B Never, service requests disabled 01 _B While gate (selected trigger signal) is high 10 _B While gate (selected trigger signal) is low 11 _B Always, for each new result value
CFMSV	[23:16]	rw	CIC Filter (Main Chain) Start Value The decimation counter begins counting at value CFMSV, when started or restarted. Valid values are 03 _H to CFMDF (4 to selected decimation factor).
CFMDCNT	[31:24]	rh	CIC Filter (Main Chain) Decimation Counter The decimation counter counts the filter cycles until an output is generated, i.e. the oversampling rate. CFMDCNT counts down from the respective start value.

Delta-Sigma Analog-to-Digital Converter (DSADC)

FCFGAx (x = 0, 2, 3)
Filter Configuration Register x, Auxiliary Filter

 (x * 0100_H + 0118_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
CFADF	[7:0]	rw	CIC Filter (Auxiliary) Decimation Factor The decimation factor of the Auxiliary CIC filter is CFADF + 1. Valid values are 03 _H to 1F _H (4 to 32).
CFAC	[9:8]	rw	CIC Filter (Auxiliary) Configuration 00 _B CIC1 01 _B CIC2 10 _B CIC3 11 _B CICF
SRGA	[11:10]	rw	Service Request Generation Auxiliary Filter 00 _B Never, service requests disabled 01 _B Auxiliary filter: As selected by bitfields ESEL and EGT (if integrator enabled) 10 _B Alternate source: Capturing of a sign delay value to register CGSYNCx (x = 0, 2, 3) 11 _B Reserved
ESEL	[13:12]	rw	Event Select Defines when an event for the auxiliary filter is generated (see also bit EGT). 00 _B Always, for each new result value 01 _B If result is inside the boundary band 10 _B If result is outside the boundary band 11 _B Reserved

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
EGT	14	rw	Event Gating Defines if events for the auxiliary filter are coupled to the integration window. 0 _B Separate: generate events according to ESEL 1 _B Coupled: generate events only when the integrator is enabled and after the discard phase defined by bitfield NVALDIS ¹⁾
0	15	r	Reserved, write 0, read as 0
AFSC	[17:16]	rw	Auxiliary Filter Shift Control Selects the displacement caused by the data shifter at the auxiliary CIC filter output. 00 _B No shift, use full range 01 _B Shift by 1 10 _B Shift by 2 11 _B Shift by 3
0	[23:18]	r	Reserved, write 0, read as 0
CFADCNT	[31:24]	rh	CIC Filter (Auxiliary) Decimation Counter The decimation counter counts the filter cycles until an output is generated, i.e. the oversampling rate.

1) While the integrator is bypassed, event gating is disabled, i.e. service requests are generated according to bitfield ESEL. The event gating suppresses service requests, result values are still stored in register RESAx.

30.6.2 Recommended Settings

The digital filters, in particular the main filter chain, provide a number of configuration options to control their operation. However, not all combinations achieve the performance the filters can deliver. The following recommendations describe the dependence of the performance on the configuration settings as well as some standard operating modes.

The digital result values that represent the input signal depend on the configuration of the filter chain, i.e. on the decimation factor, the involved filter elements, and the settings of the data shifters.

Modulator Values

The full-scale values produced by the on-chip modulator are subject to a certain variation due to the involved analog circuitry and also depend on the operating mode:

- $\pm 3 \text{ 600}_D$ as an uncalibrated average value
- $\pm 4 \text{ 096}_D$ in case of an overload condition ($I_{REN} = 1$)

The example calculations below use a full-scale value of $\pm 3 \text{ 800}_D$ ($0ED8_H/F128_H$) to avoid filter overflows.

Note: The modulator output value variation is compensated by the calibration sequence.

Gain Factor of the CIC Filter

The gain factor of the CIC filter is 1.0 for decimation rates of 2^N .

The automatic data shifter after the CIC filter compensates the variation of the result magnitude caused by different decimation factors. This compensation, however, can only be done by powers of 2, while the decimation rate itself can be selected arbitrarily. The gain of the CIC filter, therefore, ranges from 1.0 (decimation rate is 2^N , e.g. 32 or 64) down to approximately 0.5 (decimation rate is $2^N + 1$, e.g. 33 or 65).

The influence of this varying gain on the result magnitude can be computed by using the following approach:

- Compute the factor according to the selected filter configuration and decimation rate (For CIC3 and a decimation rate of 45, the factor is $45^3 = 91 \text{ 125}$)¹⁾
- Determine the next higher power of 2 (e.g. $2^{17} = 131 \text{ 072} > 91 \text{ 125}$)
- Compute the relation of the actual factor and the next power of 2 (e.g. $91 \text{ 125} / 131 \text{ 072} = 0.695228577$)

1) Formula for CIC1: $1 \times N$, for CIC2: N^2 , for CIC3: N^3 , for CICF: $2 \times N^2$.

Delta-Sigma Analog-to-Digital Converter (DSADC)

Gain Factor of FIR Filters and Data Shifters

The gain factor of the FIR filters is defined by their coefficients (see [Section 30.4.2](#)).

For FIR0 the gain factor is:

$$2 \times (-17 - 25 + 83 + 256) / 1\,024 \\ = 0.580078125$$

For FIR1 the gain factor is:

$$2 \times (-3 - 2 + 2 + 9 - 1 - 14 - 8 + 19 + 25 - 13 - 55 - 12 + 126 + 256) / 1\,024 \\ = 0.642578125$$

Shifting the data values by one bit equals a gain factor of 2.0. Therefore, the resulting gain factor is 2^S , where S is the selected value of the respective data shifter.

Avoiding Clipping

To avoid overflow and clipping of values within the filter chain, the magnitude of the result values must not exceed $\pm 2^{15}$ at any stage. This is the basis for selecting the correct settings for the data shifters. See also the example in [Table 30-7](#).

Total Gain Factor of the Digital Filter Chain

Multiplying the gain factors of the individual enabled elements of the filter chain results in the total gain factor:

$$G_{TOT} = G_{CIC} \times G_{SC} \times G_{FIR0} \times G_{SF0} \times G_{FIR1} \times G_{SF1}$$

$$G_{TOT} = G_{CIC} \times 2^{MFSC} \times 0.580078125 \times 2^{FSH} \times 0.642578125 \times 2^{DSH}$$

Note: When defining the settings for the data shifters, try to generate the maximum possible data values without exceeding the limits of the 16-bit data format at each stage. This minimizes errors induced by quantization.

Table 30-7 Example Setup Main Filter Chain, On-chip Modulator

Bitfield	Possible Values	Recommendation ¹⁾	Notes
MODCFGx.GAINSEL	1, 2, 4, 8, 16	1 (0000 _B)	Analog gain factor (valid for this example)
f_{MOD}	10 ... 20 MHz	20 MHz	Modulator sample frequency
FCFGCx.CFMC	CIC1, CIC2, CIC3, CICF	CIC3 (10 _B)	CIC filter grade
FCFGCx.CFMDF	4 ... 128	50 (31 _H)	CIC filter decimation rate (OSR) Gain factor: 125 000 / 131 072 = 0.953674316

Delta-Sigma Analog-to-Digital Converter (DSADC)
Table 30-7 Example Setup Main Filter Chain, On-chip Modulator (cont'd)

Bitfield	Possible Values	Recommendation ¹⁾	Notes
FCFGCx.MFSC	0, 1, 2, 3	3 (11 _B)	Data shifter steps after CIC filter Gain factor: $2^3 = 8.0$ (Magnitude = 28 992: OK)
FCFGMx.FSH	0, 1	0	Data shifter steps after FIR0 Gain factor: 0.580078125 (Magnitude = 16 817: OK; FSH = 1 would lead to 33 635!!)
FCFGMx.DSH	0, 1, 2, 3	1 (01 _B)	Data shifter steps after FIR1 Gain factor: $2^1 = 2.0 \times 0.642578125$ $= 1.28515625$ (Magnitude = 21 613: OK)
Total gain factor	-	-	0.953674316×8.0 $\times 0.580078125 \times 1.28515625$ $= 5.687645168$
RESMx.RESULT	-	-	$\pm 3\ 800_D \times 5.687645168$ $= \pm 21\ 613\ (546D_H)^{2)}$

1) Bitfield values in parentheses.

2) These values are given after offset correction and refer to the nominal modulator output values. The final result values are adjusted by the complete calibration sequence.

Dithering Control

The dithering features reduces the idle tones caused by low-frequency input signals. After reset, dithering is enabled, but the dithering trim value is 000_B. To optimize the effect, set the trim value to 011_B.

Delta-Sigma Analog-to-Digital Converter (DSADC)
30.6.3 Group Delay

Data that enter a digital filter need a certain number of filter clocks before they appear at the filter's output and can be used by the system. The effective group delay depends on the configuration of the filter chain. **Table 30-8** summarizes the delays incurred by the elements of the filter chain. "N" indicates the selected oversampling rate (decimation factor) of the CIC filter. Delays are listed in modulator clock cycles.

Table 30-8 Group Delay Summary

Filter Chain Element	Delay [t_{MOD}]	Notes
On-chip modulator	7	Delay of analog frontend
Input select/adjust unit	1	Synchronization
CIC1	$1 \times (N-1) / 2$	
CIC2	$2 \times (N-1) / 2$	
CIC3	$3 \times (N-1) / 2$	
CICF	$(2 \times N) - 1$	
FIR0	$(3.5 \times N) + C0$	$C0 = 3$ To be added to CIC delay
FIR1	$(27 \times N) + C1$	$C1 = N + 5$ To be added to CIC / FIR0 delay
OC	0	
Integrator	X	Depending on the selected number of discarded and integrated values
Offset correction	0	

Example

$f_{MOD} = 16.66$ MHz, $t_{MOD} = 60$ ns, OSR(CIC3) = 32, FIR active, OC off, no integration.

Group delay:

$$[7 + 1 + (3 \times 31 / 2) + (3.5 \times 32 + 3) + (28 \times 32 + 5)] \times t_{MOD} =$$

$$(7 + 1 + 46.5 + 115 + 901) \times t_{MOD} = 1\,070.5 \times t_{MOD} = 64.23 \mu\text{s}$$

At the given oversampling rate of $32 \times 2 \times 2 = 128$ this equals a delay of $8.36 \times t_D$ (data output rate cycles).

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.7 Conversion Result Handling

The DSADC preprocesses the conversion result data before storing them for retrieval by the CPU or a DMA channel.

Conversion result handling comprises the following functions:

- Filtering and Post-Processing
- Storage of Conversion Results
- Result Event Generation

The result data words are generated by feeding the input data stream through a chain of filter elements and decimating it by a selectable ratio. The selectable integrator can further reduce the output data rate.

The elements of the main filter chain after the initial CIC filter can be bypassed, i.e. the filter chain is configurable and its behavior can be adapted to the requirements of the actual application.

A programmable offset is subtracted automatically from each result value before being stored in the result register. This offset value is stored in the corresponding register OFFMx. It may be an arbitrary value defined by the application.

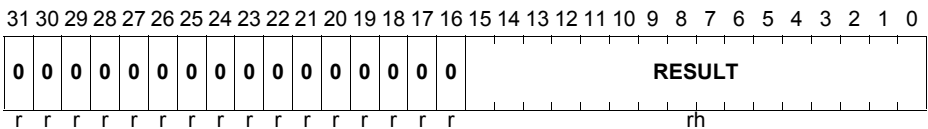
The offset value can also be obtained by using the offset measurement mode (both inputs to V_{CM} via bitfields INCFGP and INCFGN in register MODCFGx (x = 0, 2, 3)).

The auxiliary filter is fixed. The result values of the auxiliary filter are also available for the application, although in many cases the comparator output signal will be sufficient.

The conversion result values are stored in result register RESMx and RESAx, respectively.

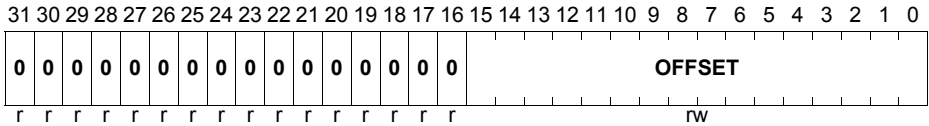
RESMx (x = 0, 2, 3)

Result Register x Main Filter (x * 0100_H + 0130_H) **Reset Value: 0000 0000_H**

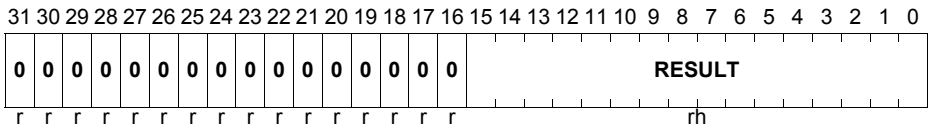


Field	Bits	Type	Description
RESULT	[15:0]	rh	Result of most recent conversion Signed value (two's complement)
0	[31:16]	r	Reserved, write 0, read as 0

Delta-Sigma Analog-to-Digital Converter (DSADC)

OFFMx (x = 0, 2, 3)
Offset Register x Main Filter ($x * 0100_H + 0138_H$) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
OFFSET	[15:0]	rw	Offset Value This signed value is subtracted from each result before being written to the corresponding result register RESMx.
0	[31:16]	r	Reserved, write 0, read as 0

RESAx (x = 0, 2, 3)
Result Register x Auxiliary Filter ($x * 0100_H + 0140_H$) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
RESULT	[15:0]	rh	Result of most recent conversion Signed value (two's complement)
0	[31:16]	r	Reserved, write 0, read as 0

Result Representation

Due to the differential input stage, the results are signed values stored in a 16-bit two's complement format. The CIC filter's output value depends on the selected filter parameters and decimation factor. An automatic data shifter extracts the most significant bits from this filter result.

 Delta-Sigma Analog-to-Digital Converter (DSADC)

30.8 Service Request Generation

The DSADC can activate service request output signals to issue an interrupt, to trigger a DMA channel, or to trigger other on-chip modules.

Service request connections can be found in [Table 30-13 “Digital Connections in the TC26x B-Step” on Page 30-90](#).

Several events can be assigned to each service request output. Service requests can be generated by several types of events:

- **Result events:** indicate a new valid result in a result register. Usually, this triggers a read action by the CPU (or DMA). Result events are generated at the output rate of the configured filter chain.
Can be issued via SRAX and SRMx.
- **Alarm events:** indicate that a conversion result value is within a programmable value range. This offloads the CPU/DMA from background tasks, i.e. a service request is only activated if the specified conversion result range is met or exceeded.
Can be issued via SRAX.
- **Special events:** indicate specific circumstances of previously configured functions.
 - Timestamp trigger event can generate a service request.
Can be issued via SRMx.
 - Capture event for sign delay measurement can generate a service request.
Can be issued via SRAX.

Each event is indicated by a dedicated flag that can be cleared by software. If a service request is enabled for a certain event, the service request is generated for each event, independent of the status of the corresponding event indication flag. This ensures efficient DMA handling of DSADC events (the event can generate a service request without the need to clear the indication flag).

Note: For service request signals to occur, they must be activated via bitfields SRGM and/or SRGA.

Gating Requests of Auxiliary Channel

Result/Alarm events on service request SRAX can be suppressed while the integrator is enabled and is controlled by an external signal. When bit EGT in register **FCFGAx (x = 0, 2, 3)** is set, result events (ESEL = 00_B) or alarm events (ESEL = 01_B or ESEL = 10_B) only generate service requests while the integrator is active after the programmed discard phase.

This prevents alarm requests from being generated while the signal is not evaluated via the main filter chain.

Note: While the integrator is disabled (ITRMODE = 00_B), bit EGT has no effect on the generation of auxiliary service requests.

Delta-Sigma Analog-to-Digital Converter (DSADC)

The following registers provide a set of bits for each available channel.
The number of available channels depends on the chosen device type.

EVFLAG
Event Flag Register
(00E0_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	AL EV3	AL EV2	0	AL EV0
r	r	r	r	r	r	r	r	r	r	r	r	rwh	rwh	r	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	RES EV3	RES EV2	0	RES EV0
r	r	r	r	r	r	r	r	r	r	r	r	rwh	rwh	r	rwh

Field	Bits	Type	Description
RESEV_x (x = 0, 2, 3)	x	rwh	Result Event 0 _B No result event 1 _B A new result has been stored in register RESM _x
0	1, [15:4]	r	Reserved, write 0, read as 0
ALEV_x (x = 0, 2, 3)	x+16	rwh	Alarm Event 0 _B No alarm event 1 _B An alarm event has occurred
0	17, [31:20]	r	Reserved, write 0, read as 0

Delta-Sigma Analog-to-Digital Converter (DSADC)

EVFLAGCLR
Event Flag Clear Register
(00E4_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	AL EC3	AL EC2	0	AL EC0
r	r	r	r	r	r	r	r	r	r	r	r	w	w	r	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	RES EC3	RES EC2	0	RES EC0
r	r	r	r	r	r	r	r	r	r	r	r	w	w	r	w

Field	Bits	Type	Description
RESECx (x = 0, 2, 3)	x	w	Result Event Clear 0 _B No action 1 _B Clear bit RESEVx
0	1, [15:4]	r	Reserved, write 0, read as 0
ALECx (x = 0, 2, 3)	x+16	w	Alarm Event Clear 0 _B No action 1 _B Clear bit ALEVx
0	17, [31:20]	r	Reserved, write 0, read as 0

*Note: Software can set flags RESEVx and ALEVx and trigger the corresponding event by writing 1 to the respective bit. Writing 0 has no effect.
Software can clear these flags by writing 1 to bit RESECx and ALECx, respectively.*

Delta-Sigma Analog-to-Digital Converter (DSADC)

30.9 Resolver Support

Resolver applications determine the rotation angle by evaluating the signals from two orthogonally placed coils. These coils are excited by the magnetic field of a third coil. The DSADC can read the two return signals using two input channels and can also generate the excitation sine signal (carrier). It also provides synchronization logic to compensate the delay between the generated carrier signal and the received position signals. The integrator stage converts the carrier-based return signals to position-based values (carrier cancellation).

30.9.1 Carrier Signal Generation

The carrier signal generator (CG) is supplied with the selected internal clock signal (f_{CLK}) and outputs a PWM signal that induces a sine signal in the excitation coil of the resolver. Alternatively, it can generate PWM patterns that resemble triangle or square signals (see [Figure 30-20](#)). The polarity of the carrier signal can be selected.

A carrier signal period consists of 32 steps. Each step equals a PWM period of 32 cycles.

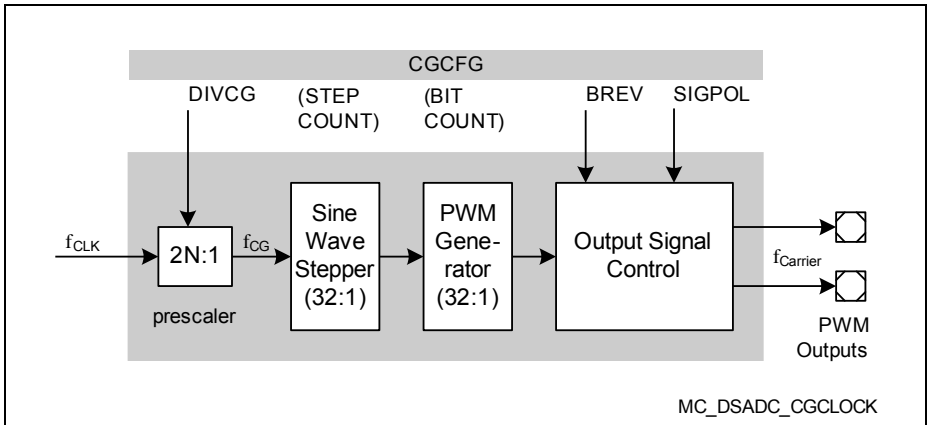


Figure 30-19 Carrier Generator Block Diagram

Bit-reverse generation mode increases the frequency spectrum to yield a smoother induced sine signal. This is done by distributing the 0 and 1 bits over the 32 cycles of a PWM period.

The generated pattern is actually a cosine signal, i.e. it starts at the maximum output value. This is advantageous if the output pin is pulled high before the carrier signal is generated. In case of a pull-down the inverted output signal should be selected.

Delta-Sigma Analog-to-Digital Converter (DSADC)

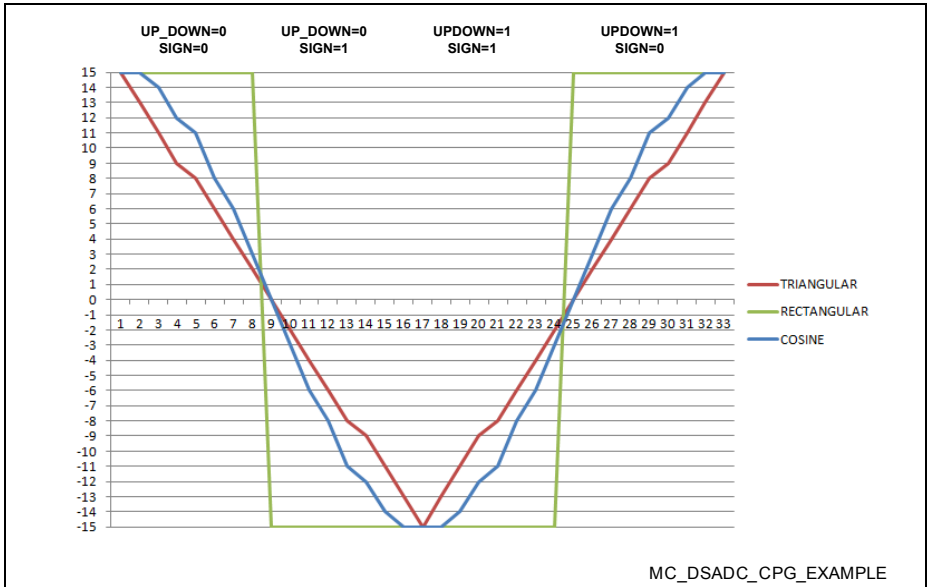


Figure 30-20 Example Pattern/Waveform Outputs

CGCFG

Carrier Generator Configuration Register

(00A0_H)

Reset Value: 0710 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SGN CG	STE PD	STE PS	STEPCOUNT				0	0	0	BITCOUNT				
r	rh	rh	rh	rh				r	r	r	rh				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RUN	0	0	0	0	0	0	0	DIVCG			SIG POL	B REV	CG MOD		
rh	r	r	r	r	r	r	r	rw			rw	rw	rw		

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
CGMOD	[1:0]	rw	Carrier Generator Operating Mode 00 _B Stopped 01 _B Square wave 10 _B Triangle 11 _B Sine wave Stopping the carrier generator (CGMOD = 00 _B) terminates the PWM output after completion of the current period (indicated by bit RUN = 0).
BREV	2	rw	Bit-Reverse PWM Generation 0 _B Normal mode 1 _B Bit-reverse mode
SIGPOL	3	rw	Signal Polarity 0 _B Normal: carrier signal begins with +1 1 _B Inverted: carrier signal begins with -1
DIVCG	[7:4]	rw	Divider Factor for the PWM Pattern Signal Generator Defines the input frequency of the carrier signal generator, derived from the selected internal clock source. 0 _H $f_{CG} = f_{CLK} / 2$ 1 _H $f_{CG} = f_{CLK} / 4$ 2 _H $f_{CG} = f_{CLK} / 6$... F _H $f_{CG} = f_{CLK} / 32$ <i>Note: The frequency of the carrier signal itself is $f_{CG} / 1024$.</i>
0	[14:8]	r	Reserved, write 0, read as 0
RUN	15	rh	Run Indicator 0 _B Stopped (cleared at the end of a period) 1 _B Running
BITCOUNT	[20:16]	rh	Bit Counter Counts the 32 cycles generated for each step
0	[23:21]	r	Reserved, write 0, read as 0
STEPCOUNT	[27:24]	rh	Step Counter Counts the ±16 steps generated for each carrier signal period

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
STEPS	28	rh	Step Counter Sign Indicates the sign of the step counter value 0 _B Step counter value is positive 1 _B Step counter value is negative
STEPD	29	rh	Step Counter Direction 0 _B Step counter is counting up 1 _B Step counter is counting down
SGNCG	30	rh	Sign Signal from Carrier Generator 0 _B Positive values 1 _B Negative values
0	31	r	Reserved, write 0, read as 0

30.9.2 Return Signal Synchronization

In a resolver, the received return signals are induced by the carrier signal and their amplitudes are modulated with the sine and cosine magnitudes corresponding to the current resolver position. These amplitudes are determined by integrating the return signals over a carrier signal period.

To properly integrate their magnitude, the return signals must be rectified. For this purpose the carrier generator provides the sign information of the generated carrier signal (SGNCG in register **CGCFG**).

Alternatively, an external carrier signal generator can be used. If this generator delivers a sign signal, this can be input to a pin and is then used as external carrier sign signal. If no sign signal is available, the carrier signal itself can be converted by the next adjacent input channel and its sign signal is then used as alternate carrier sign signal.

*Note: The resulting sign signal is called SGNCS in **Figure 30-21**.*

The rectification of the received signals must be delayed to compensate the round trip delay through the system (driver, resolver coils, cables, etc.). For the rectification, the received values are multiplied with the delayed carrier sign signal (SGND in register **RECTCFGx (x = 0, 2, 3)**). This synchronization is done for each channel separately, to achieve the maximum possible amplitudes for each signal.

Note: The rectification unit is part of the integrator stage. Therefore, it is only active while the integrator is active.

The delay is realized with the sign delay counter SDCOUNT. SDCOUNT is cleared and started upon a falling edge of the selected sign signal (SGNCS in **Figure 30-21**), i.e. at the begin of the positive halfwave of the carrier signal. After counting SDPOS results from the filter chain, also the rectification signal (SGND) is cleared, indicating positive values from now on. After counting SDNEG values, the rectification signal is set, indicating negative values (see **Figure 30-21**).

The compare values SDPOS and SDNEG are stored by the application software. SDPOS is the delay value that accounts for the resolver signal's round trip delay. This delay is constantly measured by capturing the current counter value into bitfield SDCAP when the first positive result (after negative results) is received in the respective channel. Software can read these value and compute a delay value e.g. by averaging a series of measured values to compensate noise. The delay for the negative halfwave (SGND = 0) is determined by adding the duration of a carrier signal halfwave. This value is written to bitfield SDNEG.

A new captured value is indicated by setting the flag SDCVAL. This flag is cleared when reading register CGSYNCx.

Capturing a new value can trigger a service request. The service request line of the auxiliary channel is used for this purpose. This alternate request source is selected by bitfield SRGA in register **FCFGAx (x = 0, 2, 3)**.

Delta-Sigma Analog-to-Digital Converter (DSADC)

RECTCFGx (x = 0, 2, 3)
Rectification Configuration Register x

 (x * 0100_H + 01A8_H)

 Reset Value: 8000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SGND	SGNCS	0	0	0	0	0	0	0	0	0	0	0	0	0	0
rh	rh	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDCV	VAL	0	0	0	0	0	0	0	0	SSRC	0	0	0	RFEN	
rh	r	r	r	r	r	r	r	r	r	rw	r	r	r	r	rw

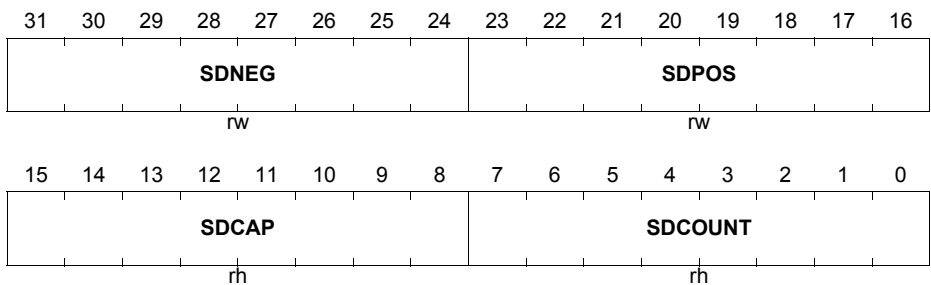
Field	Bits	Type	Description
RFEN	0	rw	Rectification Enable General control of the rectifier circuit. 0 _B No rectification, data not altered 1 _B Data are rectified according to SGND <i>Note: Rectification is only active while the integrator is active.</i>
0	[3:1]	r	Reserved, write 0, read as 0
SSRC	[5:4]	rw	Sign Source Selects the sign signal that is to be delayed. 00 _B On-chip carrier generator 01 _B Sign of result of next channel 10 _B External sign signal A 11 _B External sign signal B
0	[14:6]	r	Reserved, write 0, read as 0
SDCV	15	rh	Sign Delay Capture Valid Flag Indicates a new value in bitfield SDCAP. 0 _B No new result available 1 _B Bitfield SDCAP has been updated with a new captured value and has not yet been read
0	[29:16]	r	Reserved, write 0, read as 0
SGNCS	30	rh	Selected Carrier Sign Signal 0 _B Positive values 1 _B Negative values

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
SGND	31	rh	Sign Signal Delayed 0_B Positive values 1_B Negative values

CGSYNCx (x = 0, 2, 3)
Carrier Generator Synchronization Register x

$$(x * 0100_H + 01A0_H)$$

Reset Value: 0000 0000_H


Field	Bits	Type	Description
SDCOUNT	[7:0]	rh	Sign Delay Counter Counts the result values from the filter chain to delay the carrier sign signal
SDCAP	[15:8]	rh	Sign Delay Capture Value Indicates the result values counted between the begin of the positive halfwave of the carrier signal and the first received positive value.
SDPOS	[23:16]	rw	Sign Delay Value for Positive Halfwave Defines the content of SDCOUNT to generate a low delayed sign signal SGND (positive values).
SDNEG	[31:24]	rw	Sign Delay Value for Negative Halfwave Defines the content of SDCOUNT to generate a high delayed sign signal SGND (negative values).

Delta-Sigma Analog-to-Digital Converter (DSADC)

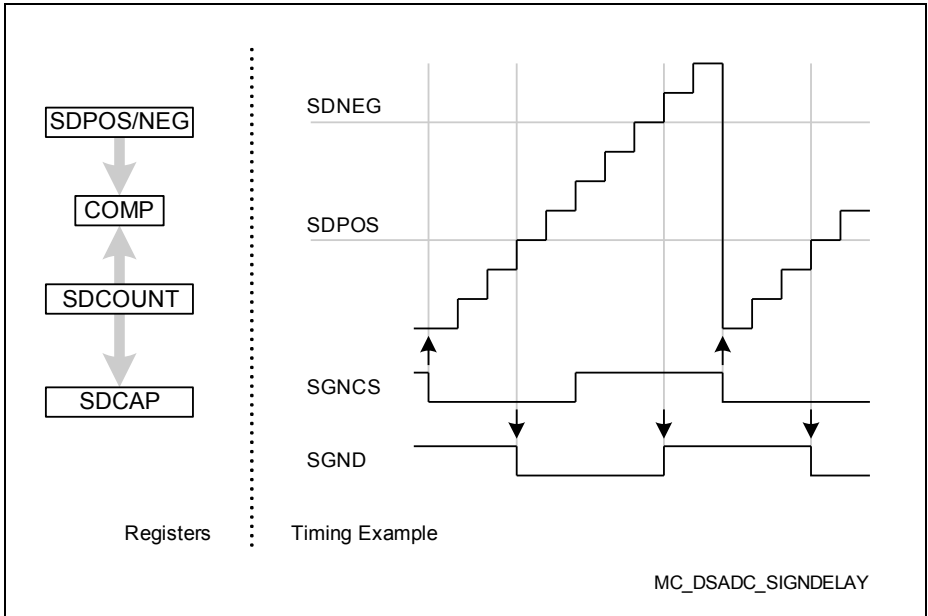


Figure 30-21 Sign Delay Example

Note: Whenever a new result value becomes available from the filter chain, the rectification counter (SDCOUNT) is updated and the rectified value is forwarded to the integrator.

Delta-Sigma Analog-to-Digital Converter (DSADC)
30.10 Time-Stamp Support

Some applications need to determine the result value at certain points of time inbetween two regular output values. The interpolation algorithm needs to determine the position of the required point of time in relation to the regular results.

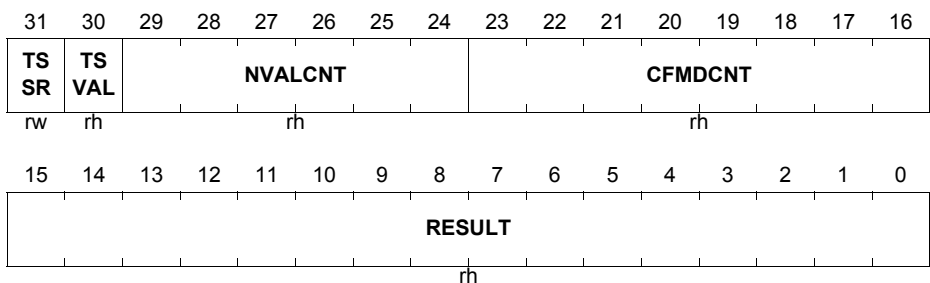
This interpolation is supported by providing a timestamp that marks the delay since the last regular output value. This timestamp is composed of:

- the last regular result value
- the current decimation counter value
- the current integrator counter value

All timestamp information is combined into one register, so the complete timestamp can easily be stored away by a single DMA transfer. The timestamp information is captured into register TSTMPx upon a hardware trigger. For this purpose, the trigger signal is used, which is selected by bitfield TRSEL in register **DICFGx** (**x = 0, 2, 3**). Bitfield TSTRMODE selects the edge(s) that capture timestamp information.

The valid flag TSVAL is set when timestamp information is stored and is cleared when register TSTMP is read. This indicates to the application software if or not a timestamp trigger has occurred before the corresponding result value was generated.

When timestamp information is stored, a service request can optionally be generated. This allows application software to operate on all result data, even if multiple timestamp triggers occur before a regular result service request.

TSTMPx (x = 0, 2, 3)
Time-Stamp Register x ($x * 0100_H + 0150_H$) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
RESULT	[15:0]	rh	Result of most recent conversion This value is copied from register RESMx (x = 0, 2, 3).

Delta-Sigma Analog-to-Digital Converter (DSADC)

Field	Bits	Type	Description
CFMDCNT	[23:16]	rh	CIC Filter (Main Chain) Decimation Counter This value is copied from register FCFGCx (x = 0, 2, 3) .
NVALCNT	[29:24]	rh	Number of Values Counted This value is copied from register IWCTR_x (x = 0, 2, 3) .
TSVAL	30	rh	Timestamp Valid Indicates valid timestamp information. 0 _B No timestamp trigger occurred since last read access 1 _B Timestamp information has been stored after a timestamp trigger
TSSR	31	rw	Timestamp Service Request 0 _B Just store the timestamp, no service request 1 _B Generate a service request on SRM _x after storing timestamp information ¹⁾

1) Service request generation for the main chain must be enabled via bitfield FCFG_x.SRGM.

 Delta-Sigma Analog-to-Digital Converter (DSADC)

30.11 Implementation into the TC26x B-Step

This section describes the actual implementation of the ADC module into the TC26x B-Step, i.e. the incorporation into the microcontroller system.

The following information is listed:

- [“Product-Specific Configuration” on Page 30-85](#)
- [“Summary of Registers and Locations” on Page 30-86](#)
- [“Analog Module Connections in the TC26x B-Step” on Page 30-89](#)
- [“Digital Module Connections in the TC26x B-Step” on Page 30-90](#)

30.11.1 Product-Specific Configuration

The functional description describes the features and operating modes of the DSADC in a general way. This section summarizes the configuration that is available in this product (TC26x B-Step).

The DSADC features a number of twin-modulators that handle 2 input channels each.

Table 30-9 General Converter Configuration in the TC26x B-Step

Channel	Analog Inputs ¹⁾	Reference Voltages	Notes
0	2	V_{AREF1} , V_{AGND1}	2:1 analog multiplexer
1	--	--	Not available
2	1	V_{AREF1} , V_{AGND1}	
3	4	V_{AREF1} , V_{AGND1}	4:1 analog multiplexer

1) Differential input pin pairs.

Delta-Sigma Analog-to-Digital Converter (DSADC)
30.11.2 Summary of Registers and Locations

The DSADC is built from a series of channels that are controlled in an identical way. This makes programming versatile and scalable. The corresponding registers, therefore, have an individual offset assigned (see [Table 30-11](#)). The exact register location is obtained by adding the respective register offset to the base address (see [Table 30-10](#)) of the corresponding channel.

Due to the regular structure, several registers appear within each channel. This is indicated in the register overview table by placeholders:

- $0X##_H$ means: $x \times 0100_H + 01##_H$, for $x = 0, 2, 3$

Table 30-10 Registers Address Space

Module	Base Address	End Address	Note
DSADC	F002 4000 _H	F002 4FFF _H	

Table 30-11 Registers Overview

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Page Num.
			Read	Write	
ID	Module Identification Register	0008 _H	U, SV	BE	30-6
CLC	Clock Control Register	0000 _H	U, SV	SV E, P	30-7
OCS	OCDS Control and Status Register	0028 _H	U, SV	SV P	30-8
ACCEN0	Access Enable Register 0	003C _H	U, SV	SV SE	30-10
KRST0	Kernel Reset Register 0	0034 _H	U, SV	SV E, P	30-11
KRST1	Kernel Reset Register 1	0030 _H	U, SV	SV E, P	30-12
KRSTCLR	Kernel Reset Clear Register	002C _H	U, SV	SV E, P	30-13
ACCPROT	Access Protection Register	0090 _H	U, SV	SV, SE, P	30-14
GLOBCFG	Global Configuration Register	0080 _H	U, SV	U, SV P	30-16
GLOBRC	Global Run Control Register	0088 _H	U, SV	U, SV P	30-17

Delta-Sigma Analog-to-Digital Converter (DSADC)
Table 30-11 Registers Overview (cont'd)

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Page Num.
			Read	Write	
IGCFG	Initial Global Configuration Register	00D0 _H	U, SV	U, SV P	30-19
ICCFGx (x = 0, 2, 3)	Initial Channel Configuration Register x	0XD0 _H	U, SV	U, SV P	30-20
GLOBVCMH0	Common Mode Hold Voltage Register 0	0XB0 _H	U, SV	U, SV P	30-38
GLOBVCMH2	Common Mode Hold Voltage Register 2	0XB8 _H	U, SV	U, SV P	30-39
MODCFGx (x = 0, 2, 3)	Modulator Configuration Register x	0X00 _H	U, SV	U, SV P	30-30
DICFGx (x = 0, 2, 3)	Demodulator Input Configuration Register x	0X08 _H	U, SV	U, SV P	30-34
BOUNDSELx (x = 0, 2, 3)	Global Boundary Select Register	0X28 _H	U, SV	U, SV P	30-59
IWCTRx (x = 0, 2, 3)	Integration Window Control Register	0X20 _H	U, SV	U, SV P	30-56
FCFGMx (x = 0, 2, 3)	Filter Configuration Register Main Filter Chain	0X10 _H	U, SV	U, SV P	30-61
FCFGCx (x = 0, 2, 3)	Filter Configuration Register Main CIC Filter	0X14 _H	U, SV	U, SV P	30-62
FCFGAx (x = 0, 2, 3)	Filter Configuration Register Auxiliary Filter	0X18 _H	U, SV	U, SV P	30-64
RESMx (x = 0, 2, 3)	Result Register x Main Filter	0X30 _H	U, SV	U, SV P	30-70
OFFMx (x = 0, 2, 3)	Offset Register x Main Filter	0X38 _H	U, SV	U, SV P	30-71
RESAx (x = 0, 2, 3)	Result Register x Auxiliary Filter	0X40 _H	U, SV	U, SV P	30-71
EVFLAG	Event Flag Register	0XE0 _H	U, SV	U, SV P	30-73
EVFLAGCLR	Event Flag Clear Register	0XE4 _H	U, SV	U, SV P	30-74

Delta-Sigma Analog-to-Digital Converter (DSADC)
Table 30-11 Registers Overview (cont'd)

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Page Num.
			Read	Write	
CGCFG	Carrier Generator Configuration Register	00A0 _H	U, SV	U, SV P	30-76
RECTCFGx (x = 0, 2, 3)	Rectification Configuration Register	0XA8 _H	U, SV	U, SV P	30-80
CGSYNCx (x = 0, 2, 3)	Carrier Generator Synchronization Register	0XA0 _H	U, SV	U, SV P	30-81
TSTMPx (x = 0, 2, 3)	Time Stamp Register	0X50 _H	U, SV	U, SV P	30-83

Delta-Sigma Analog-to-Digital Converter (DSADC)
30.11.3 Analog Module Connections in the TC26x B-Step

The DSADC module accepts a positive and a negative analog input signal to be used as a differential input. Each analog input can also be used in single-ended mode.

The exact number of analog input channels and the available connection to port pins depend on the employed product type.

Table 30-12 lists the available inputs for each DSADC channel.

Table 30-12 Analog Connections in the TC26x B-Step

Signal	Dir.	Source/Destin.	Description
V_{AREF}	I	VAREF1	positive analog reference
V_{AGND}	I	VAGND1	negative analog reference
DS0PA	I	AN2	positive analog input of channel 0, pin A
DS0NA	I	AN3	negative analog input of channel 0, pin A
DS0PB	I	AN0	positive analog input of channel 0, pin B
DS0NB	I	AN1	negative analog input of channel 0, pin B
DS2PA	I	AN20	positive analog input of channel 2, pin A
DS2NA	I	AN21	negative analog input of channel 2, pin A
DS3PA	I	AN36, P40.6	positive analog input of channel 3, pin A
DS3NA	I	AN37, P40.7	negative analog input of channel 3, pin A
DS3PB	I	AN38, P40.8	positive analog input of channel 3, pin B
DS3NB	I	AN39, P40.9	negative analog input of channel 3, pin B
DS3PC	I	AN44	positive analog input of channel 3, pin C
DS3NC	I	AN45	negative analog input of channel 3, pin C
DS3PD	I	AN46	positive analog input of channel 3, pin D
DS3ND	I	AN47	negative analog input of channel 3, pin D

Delta-Sigma Analog-to-Digital Converter (DSADC)
30.11.4 Digital Module Connections in the TC26x B-Step

The DSADC modules accept a number of analog and digital input signals and generate a number of output signals. This section summarizes the connection of these signals to other on-chip modules or to external resources via port pins.

Table 30-13 Digital Connections in the TC26x B-Step

Signal	Dir.	Source/Destin.	Description
Channel 0			
ITR0A	I	dsadc0_trig0	Trigger signal, channel 0, input A
ITR0B	I	dsadc0_trig1	Trigger signal, channel 0, input B
ITR0C	I	adc0_trig0	Trigger signal, channel 0, input C
ITR0D	I	adc0_trig1	Trigger signal, channel 0, input D
ITR0E	I	P33.0	Trigger signal, channel 0, input E
ITR0F	I	P33.4	Trigger signal, channel 0, input F
ITR0G	I	SCU_PDOUT0	Trigger signal, channel 0, input G
ITR0H	I	-	Trigger signal, channel 0, input H
SRM0	O	ICU, GTM_dsadc0_trig_in_SRM	Service request output main channel 0
SRA0	O	ICU	Service request output aux. channel 0
SAUL0	O	GTM_dsadc0_trig_in_SAUL	Signal above upper limit ind., channel 0
SBLL0	O	GTM_dsadc0_trig_in_SBLL	Signal below lower limit ind., channel 0
Channel 2			
ITR2A	I	dsadc2_trig0	Trigger signal, channel 2, input A
ITR2B	I	dsadc2_trig1	Trigger signal, channel 2, input B
ITR2C	I	adc2_trig0	Trigger signal, channel 2, input C
ITR2D	I	adc2_trig1	Trigger signal, channel 2, input D
ITR2E	I	P33.2	Trigger signal, channel 2, input E
ITR2F	I	P33.6	Trigger signal, channel 2, input F
ITR2G	I	SCU_PDOUT2	Trigger signal, channel 2, input G
ITR2H	I	-	Trigger signal, channel 2, input H

Delta-Sigma Analog-to-Digital Converter (DSADC)
Table 30-13 Digital Connections in the TC26x B-Step (cont'd)

Signal	Dir.	Source/Destin.	Description
SRM2	O	ICU, GTM_dsadc2_trig_in_SRM	Service request output main channel 2
SRA2	O	ICU	Service request output aux. channel 2
SAUL2	O	GTM_dsadc2_trig_in_SAU2	Signal above upper limit ind., channel 2
SBLL2	O	GTM_dsadc2_trig_in_SBLL	Signal below lower limit ind., channel 2

Channel 3

ITR3A	I	dsadc3_trig0	Trigger signal, channel 3, input A
ITR3B	I	dsadc3_trig1	Trigger signal, channel 3, input B
ITR3C	I	adc3_trig0	Trigger signal, channel 3, input C
ITR3D	I	adc3_trig1	Trigger signal, channel 3, input D
ITR3E	I	P02.8	Trigger signal, channel 3, input E
ITR3F	I	P00.9	Trigger signal, channel 3, input F
ITR3G	I	SCU_PDOUT3	Trigger signal, channel 3, input G
ITR3H	I	-	Trigger signal, channel 3, input H
SRM3	O	ICU, GTM_dsadc3_trig_in_SRM	Service request output main channel 3
SRA3	O	ICU	Service request output aux. channel 3
SAUL3	O	GTM_dsadc3_trig_in_SAU3	Signal above upper limit ind., channel 3
SBLL3	O	GTM_dsadc3_trig_in_SBLL	Signal below lower limit ind., channel 3

General

f_{OSC0}	I	SCU	Internal clock signal A to feed modulators
f_{ERAY}	I	SCU	Internal clock signal B to feed modulators
MODCLK	I	SCU: f_{SPB}	Module clock
RESET	I	SCU	Reset signal (general)
SGNA	I	P00.4	Sign input A (carrier signal)
SGNB	I	P33.13	Sign input B (carrier signal)

Delta-Sigma Analog-to-Digital Converter (DSADC)

Table 30-13 Digital Connections in the TC26x B-Step (cont'd)

Signal	Dir.	Source/Destin.	Description
CGPWMP	O	P00.6, P02.1, P33.12	Positive PWM signal of carrier generator
CGPWMN	O	P00.5, P02.0, P33.11	Negative PWM signal of carrier generator

31 Inter-Integrated Circuit Module (I2C)

This chapter describes the Inter-Integrated Circuit (short I2C) Module of the TC26x B-Step. It contains the following sections:

- Overview (see [Page 31-1](#))
- I2C module functional specification (see [Page 31-4](#)) including:
 - I2C protocol (see [Page 31-4](#))
 - Clock and timing control (see [Page 31-11](#))
 - I2C kernel control logic (see [Page 31-15](#))
 - FIFO operation (see [Page 31-28](#))
 - Service request block operation (see [Page 31-45](#))
- I2C module internal registers (see [Page 31-51](#))
- I2C module implementation (see [Page 31-81](#)).

This specification is fully compliant with the I2C-bus specification version 2.1 [\[3\]](#).

Note: The I2C module register names described in this chapter are referenced in the TC26x B-Step User's Manual by the module name prefix "I2Cm_" with m being the number of the module.

31.1 Overview

This section gives an overview of the I2C-bus and of the I2C module.

31.1.1 I2C-bus Overview

The I2C protocol was developed to provide a simple and efficient data transfer between multiple devices over a short distance. It uses a bidirectional serial bus with two wires.

A serial data line (SDA) and a serial clock line (SCL) carry the information between the devices. These lines are connected to a positive supply voltage via pull-up resistors. In quiescent state, when the bus is free, both lines are high. During communication the lines are alternatively pulled to low. The output stages of devices connected to the bus must have an open-drain (CMOS) or open-collector (bipolar) to perform a wired-AND function.

A device can work as master or as slave. The master initiates the transfer, generates the clock pulses and terminates the transfer; it addresses a slave via a 7-bit or 10-bit address. Data can flow in either direction. In many applications there is only one master, typically a single-chip microcontroller, which communicates with several slaves, e.g. general purpose peripherals or application specific circuits. But also multi-master systems with arbitration and collision detection are possible.

Data on the I2C-bus can be transferred at rates of up to 100 kbit/s in standard mode, up to 400 kbit/s in fast mode and up to 3.4 Mbit/s in high-speed mode. A slow slave may stretch the clock period. The number of devices connected to the same I2C-bus is limited only by a maximum bus capacitance of 400 pF.

Inter-Integrated Circuit Module (I2C)

31.1.2 I2C Module Overview

Figure 31-1 shows a block diagram of the I2C module.

The I2C module works in accordance with I2C-bus specification version 2.1 [3]. It supports master mode, multi-master mode and slave mode. Communication is possible at different speed ranges: standard mode, fast mode and high-speed mode. Not only 7-bit I2C-bus addresses, but also 10-bit addresses can be handled. The module relieves the CPU from many time-critical tasks.

The clock for the kernel of the I2C module is derived from the system clock via a prescaler. An additional fractional divider generates the desired bit rate. The exact timing of the I2C-bus signals can be adjusted.

A FIFO is used for data transfer between CPU and I2C module during transmission and reception. This allows writing and reading of multiple bytes. Data alignment and data sizes can be configured.

Six separate interrupt requests are available: for filling or emptying the FIFO, for reacting on certain protocol events and for handling of errors.

External hardware is connected to the I2C module via a pair of receive and transmit pins.

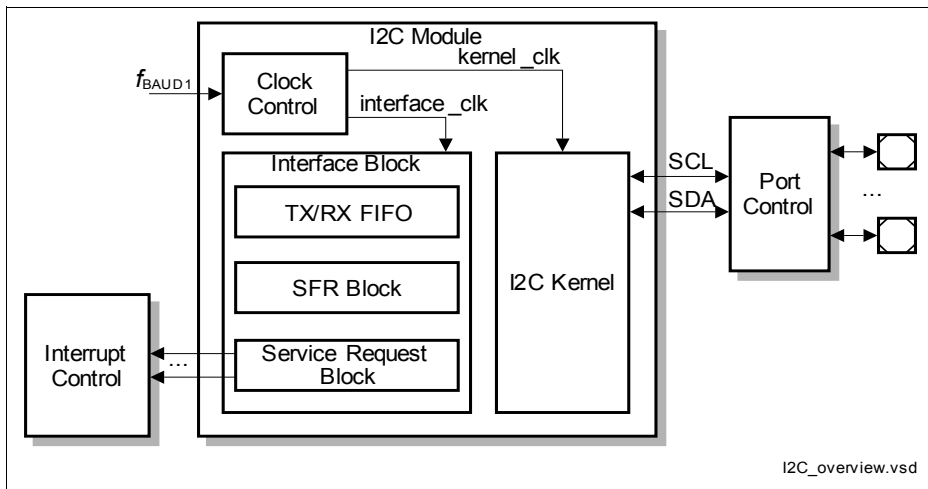


Figure 31-1 Block Diagram of the I2C Module

Features

- Compatible with I2C-bus specification version 2.1 [3]
- Master mode, multi-master mode and slave mode supported
- Different speed ranges available for data transfer:
 - Standard mode (0-100 kbit/s)
 - Fast mode (0-400 kbit/s)
 - High-speed mode (0-3.4 Mbit/s)
- 7-bit and 10-bit I2C-bus addressing supported
- Automatic execution of low-level tasks like:
 - (De)Serialization of the bus data
 - Generation/detection of start and stop signal
 - Generation/detection of acknowledge signal
 - Bus state detection
 - Bus access arbitration in multi-master mode
 - Recognition of device address in slave mode
 - Configurable detection of general call address
 - Configurable repeated start in master mode
- Flexible clock and timing control:
 - Prescaler for I2C kernel clock
 - Bit rate generation via fractional divider
 - I2C-bus signal timing adjustment
- TX/RX FIFO for buffering data from/to CPU with following features:
 - 8 FIFO stages
 - Configurable data alignment (byte, half word, word)
 - Configurable sizes for burst, transmit and receive package
 - FIFO usable as flow controller
- Advanced interrupt handling:
 - 4 data transfer interrupts (burst, last burst, single, last single)
 - Protocol interrupt with 7 sources (address match, general call, master code, arbitration lost, not-acknowledge received, transmission end, receive mode)
 - Error interrupt with 4 sources (FIFO transmit/receive overflow/underflow)

31.1.3 References

For more information, see the following documentation:

- [3] I2C-bus specification version 2.1 standard (released January-2000)
http://www.nxp.com/acrobat_download2/literature/9398/39340011.pdf

31.2 I2C Module Functional Specification

This specification is fully compatible with the I2C-bus specification version 2.1 [3].

31.2.1 I2C Protocol

Data is transmitted bit-by-bit on line SDA in conjunction with the clock on line SCL. To start communication, a master device generates a so called start condition. Subsequently data transfer starts. Therefore the data on the SDA line must be stable during the high period of the clock and may only change during SCL low phase. After all data have been transferred, the master closes transmission with a stop condition (see [Figure 31-2](#)).

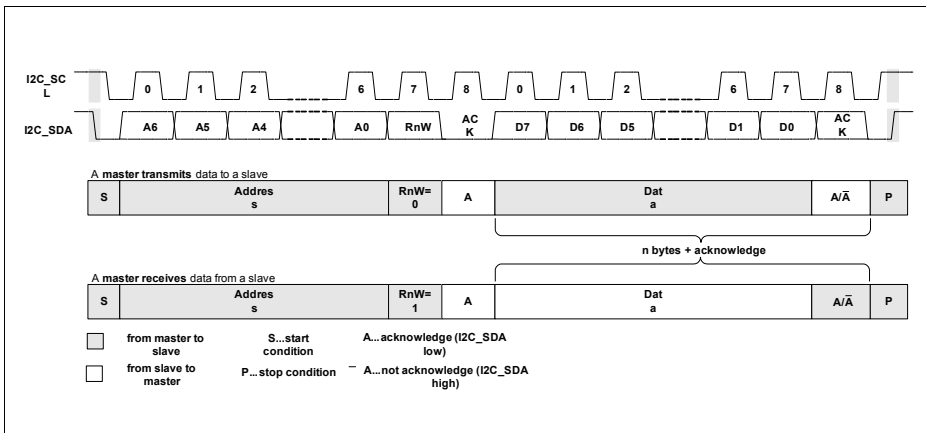


Figure 31-2 A Complete I2C Data Transfer

Start, Restart and Stop Conditions

A start condition is indicated by a high to low transition on line SDA while SCL remains high, a stop condition is defined by low to high transition under the same condition of high level on line SCL. The bus is considered to be busy after a start condition and to be free again after a stop condition. To allow continuous data transmission or reception without first generating a stop condition, a third condition has been introduced - the restart condition. The bus stays busy if this condition is generated and the same or another slave can be addressed. Start and restart condition are functionally identical.

Address Transmission

After the start condition a slave address is sent. Normally the address is 7 bits long followed by an 8th bit that is a data direction bit (Read/not Write see [Figure 31-3](#)). A zero indicates a transmission (write operation); a one indicates a request to data (read

Inter-Integrated Circuit Module (I2C)

operation). Following the address transmission the addressed device responds with an acknowledge (low on line SDA). If no acknowledge is returned, the master can then generate either a stop condition to abort the transfer or a repeated start condition to start another new transfer.

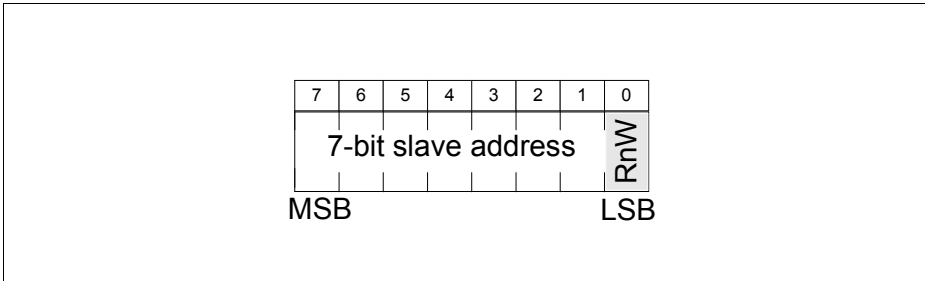


Figure 31-3 Address Byte Composition

Two groups of eight addresses are reserved for purposes, as shown in [Table 31-1](#). For example the bit combination “11110XX” is reserved for 10-bit addressing mode.

Table 31-1 Reserved I2C-bus Addresses

Address	RnW	Description
0000000	0	General call address
0000000	1	Start byte - no device is allowed to acknowledge
0000001	x	CBUS address - I2C-bus devices may not respond
0000010	x	Reserved for different bus format
0000011	x	Reserved for future use
00001XX	x	High-speed mode master code
11111XX	x	Reserved for future use
11110XX	x	10-bit slave addressing mode

General Call

A general call is characterized by transmission of address “0000000” and RnW bit = 0. It is used for addressing every device connected to the I2C-bus. However, if a device doesn’t need any of the data supplied within the general call structure, it can ignore this address by not issuing an acknowledgement. Otherwise it acknowledges this general call and the following data bytes (if required) and behaves like a slave-receiver. The received data (e.g. programmable part of slave address, address of I2C-bus master) has to be handled by software. For further information the reader is referred to the I2C-bus specification version 2.1 [\[3\]](#).

Inter-Integrated Circuit Module (I2C)

Data Transmission

Each byte transferred over the I2C-bus has to be 8 bits long whereby the number of bytes transmitted per transfer is unrestricted. Bytes are transferred MSB (Most Significant Bit) first. If the slave cannot receive or transmit a data byte until it has performed another function, it can hold the clock line SCL low (wired AND connection) to force the master into a wait state. Data transfer continues when the slave releases the clock line.

Acknowledge

Each transferred respectively received byte has to be followed by an acknowledge bit that is set by the recipient (master or slave depending on the transmission direction) onto the data line SDA. The acknowledge-related clock pulse is generated by the master. The device that has to set the acknowledge bit must pull down the SDA line during the acknowledge clock pulse so that it remains stable low during the high period of this clock pulse. Therefore in case of a slave the clock line may be held low until an acknowledge is released.

When the recipient of data doesn't acknowledge the data, for example because it was unable to receive the data or to transmit any data, the data line must be left high. A master-receiver that does not acknowledge the data from a transmitting slave device thereby tells the slave to stop transmission. The master then generates a stop condition to abort transfer or a repeated start condition to continue.

Clock Synchronization

To transfer messages a master generates its own clock on the SCL line. Data is only valid during the high period of the clock. Therefore data on line SDA has to be stable during this period and may only be changed during the low period of clock.

Two special cases must be considered. First, slave devices (or master devices operating as slaves) never generate clocks but are permitted to delay them. According to [Figure 31-4](#), if device 1 as master device for example sets line SCL to low, device 2 as slave device may extend the low phase by setting its SCL output to low. A device such as a microcontroller with limited hardware for the I2C-bus can thereby slow down the bus clock and adapt the master to the internal operating rate of this device.

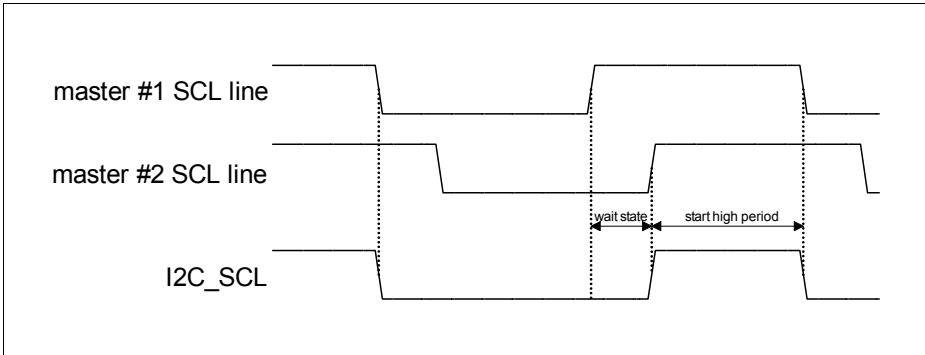


Figure 31-4 Clock Synchronization

Second, if several master devices access the I2C-bus at the same time, the SCL line is influenced by more than one device. As shown in the previous example, the SCL line will be held low by the device with the longest SCL low period. Devices with shorter periods enter a wait state during this time. In this way a synchronized SCL clock is generated.

Arbitration

To allow multi-master mode operation, bus arbitration is required. A master may start a transfer only if the bus is free. Two or more masters may generate a defined start condition simultaneously within a minimum hold time.

Arbitration takes place on the SDA line while SCL line is high, when one master transmits a high level while another master transmits a low level. Because of the wired AND functionality, the low level dominates the high level. As result the device that was trying to transmit a high level recognizes a different level on the SDA line and switches off its SDA and SCL output (see [Figure 31-5](#)).

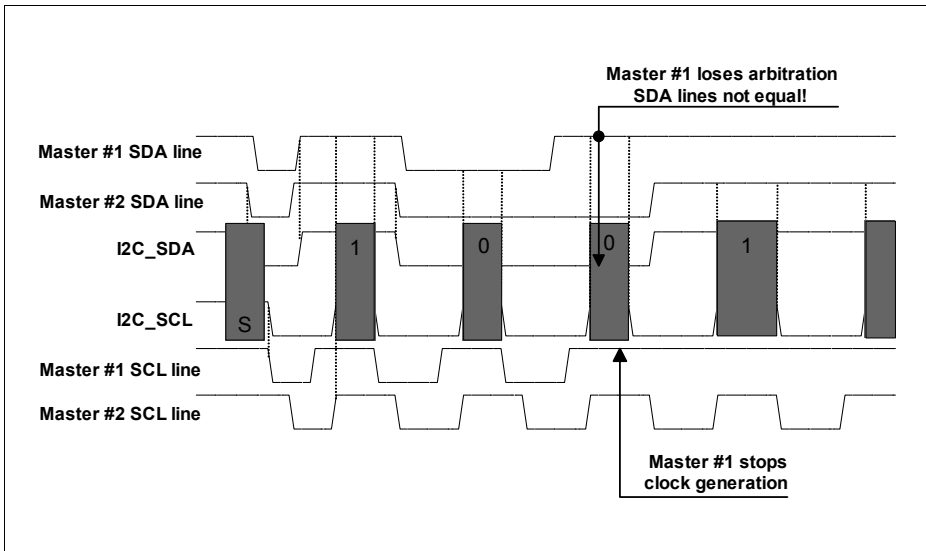


Figure 31-5 Example of an Arbitration Procedure

Arbitration may occur in different stages of transmission. A different level may potentially first appear during comparison of the address bits. If the masters are each trying to address the same device, arbitration continues with comparison of the RnW bit and subsequent data bits (if the masters are transmitting) or acknowledge bits (if the masters are receiving). Because information on the I2C-bus is determined by the winning master, no information is lost during the arbitration process. Since control of the I2C-bus is decided only by comparison of the data transmission stream over line SDA, there is no central master or any order of priority on the bus.

I2C-bus Formats

With the described rule types the formats shown in [Figure 31-6](#) are possible.

- Master-transmitter transmits to slave-receiver and stops after not-acknowledge.
- Master reads slave immediately after transmission of the RnW bit (high level) and the acknowledge from the slave, whereas the master becomes master-receiver and the slave becomes slave-transmitter.
- In combined format the just described regular sequences are handled, except for the repeated start condition instead of the stop condition between two sequences.

Inter-Integrated Circuit Module (I2C)

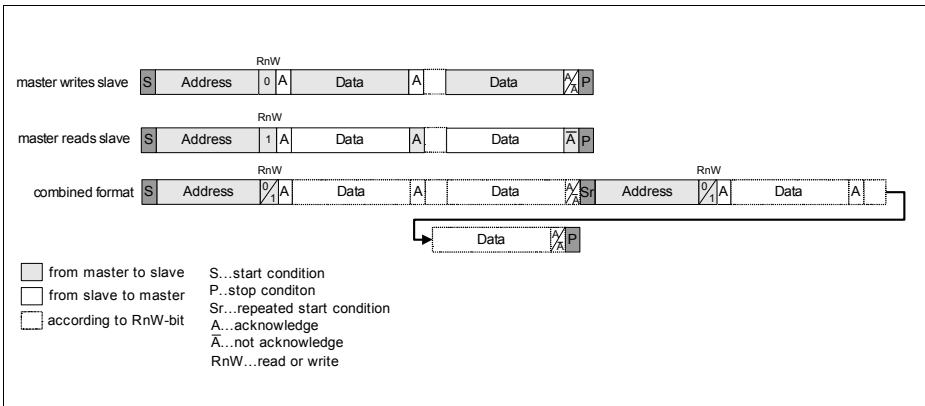


Figure 31-6 Types of I2C-bus Formats

10-bit Addressing Mode

The I2C address area is restricted to 112 applicable addresses with the 7-bit address mechanism described above. It turned out, that more combinations were required to prevent problems with the allocation of slave addresses for new devices. This problem was resolved with the 10-bit addressing scheme that is compatible with 7-bit addressing. Both modes may be used simultaneously. The following formats are possible:

To use 10 bits for addressing, the preamble address “11110XX” with RnW bit = 0 has to be used, including the two most significant bits “XX” of the 10-bit address, followed by an acknowledge from at least one 10-bit address matching slave and the second address byte containing the eight least significant bits. Only the (combined) 10-bit address matching slave now acknowledges. Data packages may now be transmitted to the addressed slave until stop condition or restart condition. If the master needs to read the slave, a combined format with a restart condition is necessary to

- First transmit the 10-bit address (first data package after preamble address contains 2nd address byte, RnW bit is 0) and (after the restart condition)
- Receive the requested data packages. Note that after the restart condition the preamble address “11110XX” has to be sent one more time, now with RnW bit = 1; the matching slave remembers that it was addressed before (see [Figure 31-7](#)). In case of a non matching preamble address, a not-acknowledge is returned.

The slave remains active as slave-transceiver until a stop condition or a restart condition occurs. According to the value of the following RnW bit a new slave may be addressed or the same slave is requested to transmit more data. Additionally the following combined formats (and mixtures of these formats) are allowed ([Figure 31-7](#)):

- A master transmits data to a slave and then reads back data from the same slave. The transfer direction is changed after the restart condition.

Inter-Integrated Circuit Module (I2C)

- A master transmits data to one slave and then transmits data to another slave.
- 10-bit and 7-bit addressing combined in one serial transfer. After restart condition the new addressing mode is selected by the address itself.

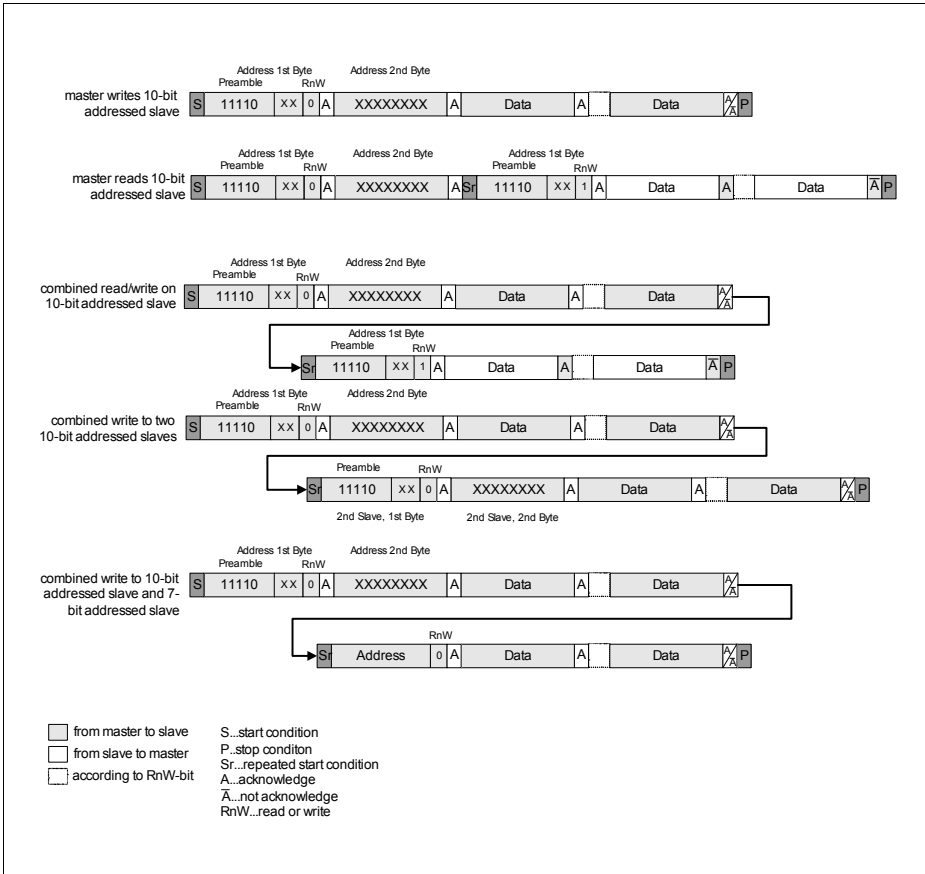


Figure 31-7 Types of I2C-bus 10-bit Address Formats

High-speed Mode

To support higher baud rates on the I2C-bus (up to 3.4 Mbaud) the specification introduces the high-speed mode. To communicate at this higher baud rate the protocol uses a so called master code in place of the address byte to indicate that a master wants to change to the high-speed mode.

Inter-Integrated Circuit Module (I2C)

Special attention must be paid to meet the requirements of rise and fall times of the signal edges and other timing characteristics when working in high-speed mode. Also it is necessary to disconnect I2C-bus devices which are not able to follow the fast communication.

After the master sends this master code, no device is allowed to acknowledge it. The involved devices change to high-speed mode and the master starts the communication. The high-speed mode transmission behavior is the same as at lower baud rates with the exception that only restart conditions and stop conditions appear (no start condition). Since the master code determines a unique master device to control the bus, the arbitration process is finished after the master code sequence is put on the bus and must not be continued when working at the new communication speed.

Improvements to the regular I2C-bus specification for high-speed mode are given in section 13.1 of the I2C-bus specification version 2.1 [3].

For high-speed mode there exists a separate configuration register **FDIVHIGHCFG** to program the baud rate; see also **Baudrate Generation**.

31.2.2 Clock and Timing Control

The I2C module offers the following features to control clock and timing:

- Module clock control for integration logic clock and I2C kernel clock (see **Section 31.4.2**)
- Baud rate generation
- I2C signal timing adjustment

31.2.2.1 Baudrate Generation

A baudrate generation unit in the I2C kernel generates the SCL signal from the kernel_clk, controlled by settings of parameters INC, DEC, and FS_SCL_LOW in registers **FDIVCFG** (normal and fast mode), **FDIVHIGHCFG** (high speed mode), and **TIMCFG**.

The I2C standard has some special requirements for the clock, like

- asymmetric duty cycle
- slave or other masters can delay the clock by extending the low period

Therefore, the mechanism of baudrate generation is not quite simple, and needs some explanation, as given hereafter. Please refer also to **Figure 31-8**.

The baudrate generator produces a clock output SCL_OUT, and also uses the feedback clock SCL_IN from the pad. SCL_IN is delayed with respect to SCL_OUT by the chip internal delays, to for SCL_OUT and t_i for SCL_IN, and by the pad rise and fall time. The pad times depend on the pad design, and on the external load and the pullup resistor. Also, the pad times may be different for normal/full speed mode and for high speed mode, if a dedicated high speed pad is being used.

Inter-Integrated Circuit Module (I2C)

SCL is the clock visible on the PCB.

The timing diagram in **Figure 31-8** shows the relations between these 3 clock signals, using nominal low and high times for the clock, as produced internally by the baudrate generator.

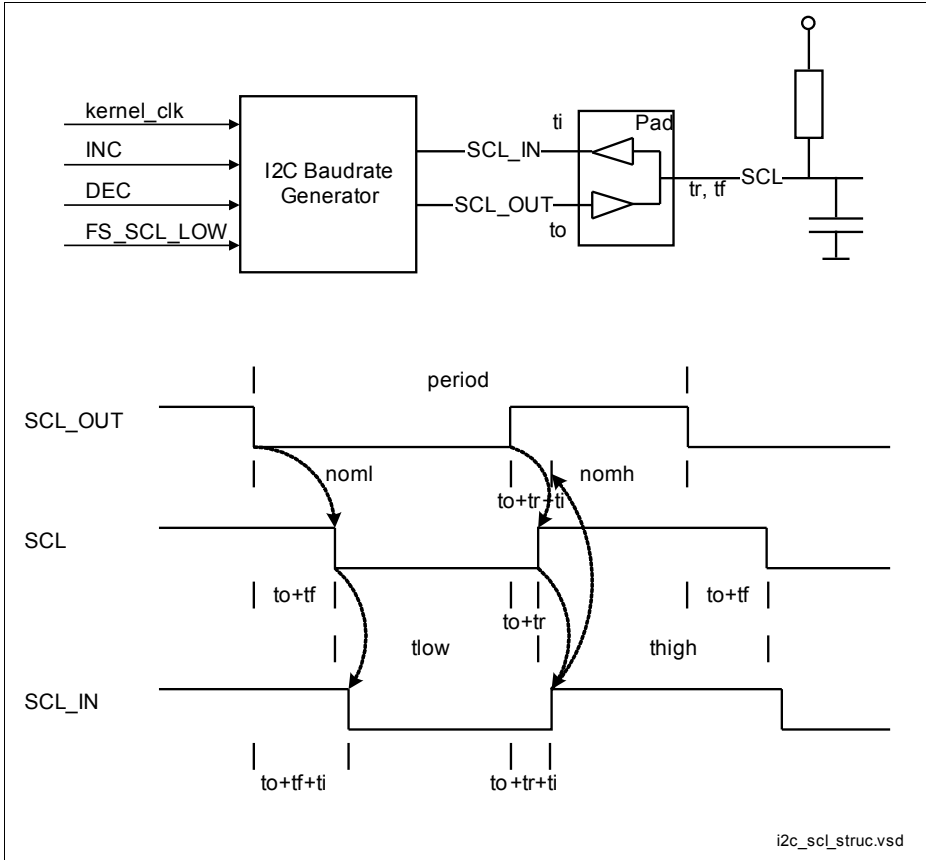


Figure 31-8 SCL Generation

The baudrate generator sets the internal SCL_OUT signal to low for the time noml, with

- Normal and full speed mode:
 - $noml = (DEC + LTC) / INC$ kernel_clk cycles
- High Speed mode:
 - $noml = (5 * DEC - 3 * INC) / INC - 1$ kernel_clk cycles when $DEC \leq 4 * INC$
 - $noml = (4 * DEC + INC) / INC - 1$ kernel_clk cycles when $DEC > 4 * INC$

Inter-Integrated Circuit Module (I2C)

Note: The operator div denotes the integer division: $a \text{ div } b = \text{greatest integer not greater than } a/b$.

Note: These and the following formulas show the mean values over a longer period of time. These may be non-integer multiples of a kernel_clk cycle. The actual value of each clock phase is always an integer multiple of a kernel_clk cycle. It is either the next smaller or the next larger one, compared to the mean value.

When the low time has passed, SCL_OUT is set to high and the baudrate generator waits until the feedback clock SCL_IN goes to high. Next the baudrate generator continues to keep SCL_OUT at high for

- Normal and full speed mode:
 - $\text{nomh} = (\text{DEC} + 3 \cdot \text{INC} - \text{LTC}) / \text{INC}$ kernel_clk cycles
- High Speed mode:
 - $\text{nomh} = 6$ kernel_clk cycles when $\text{DEC} \leq 4 \cdot \text{INC}$
 - $\text{nomh} = (\text{DEC} - 3 \cdot \text{INC}) / \text{INC} + 5$ kernel_clk cycles when $\text{DEC} > 4 \cdot \text{INC}$

Then the next low phase starts.

The Low Time Correction term LTC for normal and fullspeed mode is

- $\text{LTC} = \text{SCL_LOW_LEN}$ when $\text{EN_SCL_LOW_LEN} = 1$
- $\text{LTC} = \text{DEC} \text{ div } 4$ when $\text{EN_SCL_LOW_LEN} = 0$ and $\text{FS_SCL_LOW} = 1$
- $\text{LTC} = \text{DEC} \text{ div } 8$ when $\text{EN_SCL_LOW_LEN} = 0$ and $\text{FS_SCL_LOW} = 0$

In full speed mode, EN_SCL_LOW_LEN should be set to 1 or FS_SCL_LOW should be set to 1 in order to meet the high requirement for the SCL low time in this mode.

SCL_LOW_LEN must not be greater than DEC , when enabled via EN_SCL_LOW_LEN is set to 1.

SCL_LOW_LEN and EN_SCL_LOW_LEN are set in register **TIMCFG**.

This ends up in a period

- Normal and full speed mode:
 - $\text{period} = (2 \cdot \text{DEC} + 3 \cdot \text{INC}) / \text{INC}$ kernel_clk cycles + t_o + t_r + t_i
- High Speed mode:
 - $\text{period} = (5 \cdot \text{DEC} - 3 \cdot \text{INC}) / \text{INC} + 5$ kernel_clk cycles + t_o + t_r + t_i when $\text{DEC} \leq 4 \cdot \text{INC}$
 - $\text{period} = (5 \cdot \text{DEC} - 2 \cdot \text{INC}) / \text{INC} + 4$ kernel_clk cycles + t_o + t_r + t_i when $\text{DEC} > 4 \cdot \text{INC}$

The mechanism to enlarge the SCL_OUT high time by the time needed until the external clock has safely reached the high value ensures, that external slaves or masters can delay the clock, as requested by the standard. As a side effect, it introduces also a frequency reduction even if no external device keeps SCL at low level. This reduction is affected by process/voltage/temperature (PVT) variations on chip, and by the external load and pullup resistor.

Inter-Integrated Circuit Module (I2C)

Therefore, a perfect formula for the baudrate cannot be given. The best settings of INC and DEC for a given system should be found by measurements, taking as a start value the above formula for period with a rough estimate for $t_o + t_r + t_i$. It has to be accepted, that there is some PVT variation.

Note: In normal and full speed mode, the actual low and high times t_{low} and t_{high} of the the external clock SCL are significantly influenced by t_f and t_r . The fall time of the pad is strongly depending on the pad driver characteristic. The rise time is mostly affected by the external circuitry.

In high speed mode, this impact is less critical, due to high speed pad characteristics.

31.2.2.2 I2C Signal Timing Adjustment

The I2C standard includes a number of requirements for the SCL and the SDA timing. These cannot always be fulfilled by the timing of the pads. Therefore, additional timing adjustment options are provided in the controller logic.

Some signal timings can be shifted by multiples of `kernel_clk` cycles, as defined in register **TIMCFG**.

The proper settings have to be determined during chip evaluation.

31.2.3 I2C Kernel Control Logic

The I2C kernel can work in four different modes:

- Master-transmitter
- Master-receiver
- Slave-transmitter
- Slave-receiver

A state machine description can give a good and fast overview of the functionality. Also nearly all possible conditions can be rebuilt by stepping through the state machine and most of the error handling is covered by it. So this method is used to describe the I2C kernel.

Beginning with the top-level state machine, the starting up procedure is covered and together with the slave process and the master process sub-state machines, the active operations are comprehensible. These three state machines and their state transitions are described in the following.

As the I2C-bus working principle is packet orientated, the FIFO is typically configured as flow controller and so it issues the burst- and single-requests. This configuration is established by programming the bits RXFC and TXFC of the **FIFOCFG** register. Writing to the TPS bit-field of the **TPSCTRL** register starts the transfer.

Note: In principle it is possible to use the FIFO in non flow controller mode. When the FIFO is not configured to operate as flow controller, the behavior of the FIFO operation has to be considered. The update of the FIFO fill level is only done when a stage is filled completely. In case of a (not protocol compliant) stopped transfer of the transmitter, the device is not able to evaluate the valid bytes since the FIFO issues interrupts/updates fill level only when a stage is filled completely. To prevent this behavior, the FIFO can be, for example, programmed to word alignment, so every received byte issues an interrupt/update. Anyway the software has to take care of such upcoming not protocol compliant data transmission stops, and has to handle this for example by programming an appropriate time-out.

The Top-level State Machine

Figure 31-9 shows the top-level state machine. There are four states in the top-level state machine:

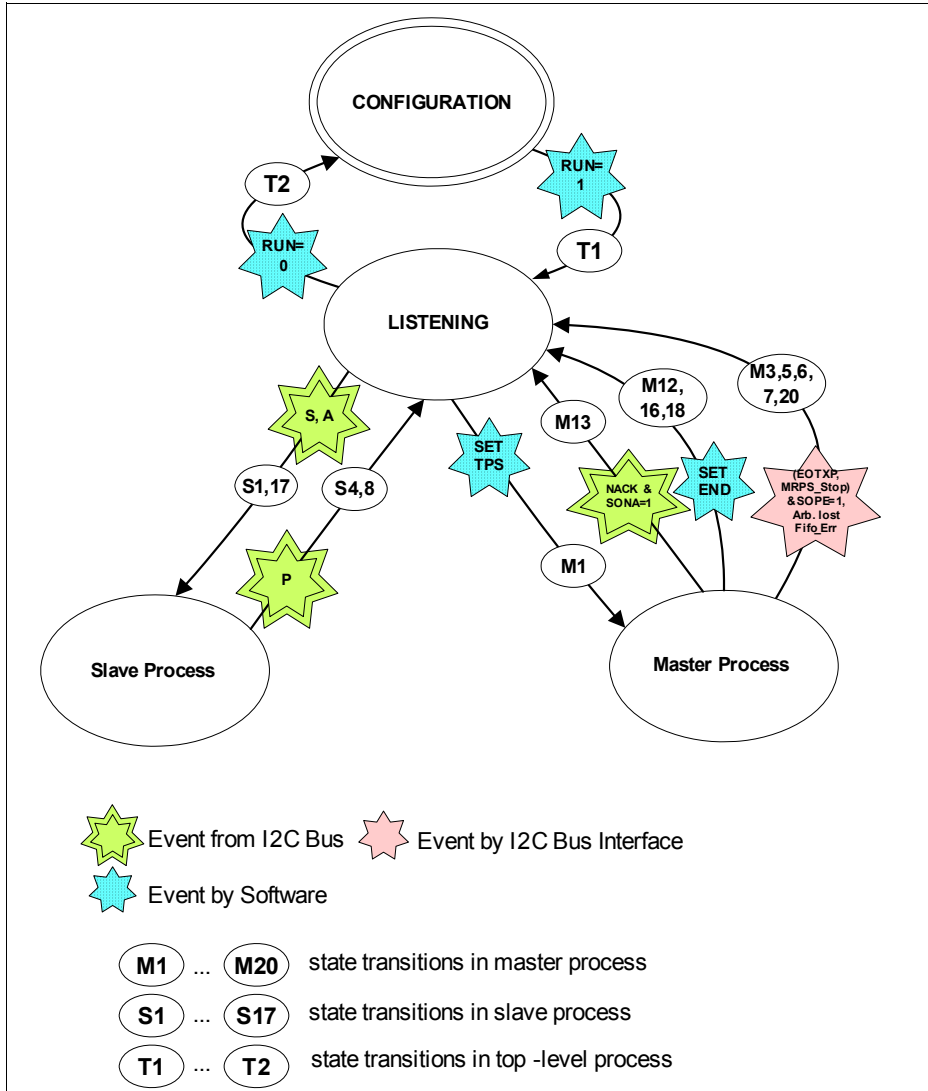


Figure 31-9 Top-level State Machine

Inter-Integrated Circuit Module (I2C)

CONFIGURATION: The peripheral is inactive; this is indicated by the RUN bit in **RUNCTRL** register, which is set to 0. In this state the peripheral should be configured by software by writing appropriate data words to the **ADDRCFG**, **FDIVCFG**, **FDIVHIGHCFG** and **FIFOCFG** registers. This determines the following features:

- Peripheral is master or slave
- Address when accessed by another device
- Answering behavior to master calls and general calls
- Operating speed on the I2C-bus
- FIFO behavior

LISTENING: The peripheral has been activated by software. This state is entered in master and slave mode. The peripheral has to observe the bus for a certain time¹⁾ to ensure that there is no running transmission on the bus after this state has been entered. If no transitions on the I2C-bus are detected during this time period, the bus is supposed to be free. Otherwise the bit-field BS (bus status) in register **BUSSTAT** is set to 01_B (bus busy). Further functionality depends on the configuration: If the peripheral is a slave, only the slave process can be entered. If the device works as master, slave and master processes can be entered.

SLAVE PROCESS: A Start condition has been detected. The device is waiting for commands from the I2C-bus. This state represents the operation when acting as slave-transmitter or slave-receiver and is itemized in an own sub state machine.

MASTER PROCESS: The device is able to control the bus and work either as master-receiver or master-transmitter. Before the master can control the bus it has to set its clock frequency by programming its fractional divider with the appropriate value by writing it to the **FDIVCFG** register (when the I2C-bus is also operating in high-speed mode the appropriate values for INC and DEC have to be written to **FDIVHIGHCFG**). The master process state is itemized in an own sub state machine.

The transition between the three states on the top-level are:

T1. The RUN bit has been set to 1 by software. The peripheral is activated and ready to observe the I2C-bus or react on transmission start commands.

T2. The RUN bit has been set to 0 again. I.e., because of new configuration effort, the peripheral is switched back to CONFIGURATION mode.

IMPORTANT: When “turning off” the peripheral it must be assured that no I2C-bus control responsibility is owned by the interface! Therefore the shut off process has to change the peripheral into a “secure” state and free the I2C-bus if necessary (generate a stop condition if in master mode). There are signals provided by the interface block which can be used to guarantee a secure shut off when turning off the I2C kernel.

1) One period of 1/100 kHz should be sufficient.

Inter-Integrated Circuit Module (I2C)

Slave Process Sub-state Machine

Figure 31-10 shows the slave process sub-state machine. There are 5 plus 1 (the listening state belongs to master and slave) different states in this sub-state machine:

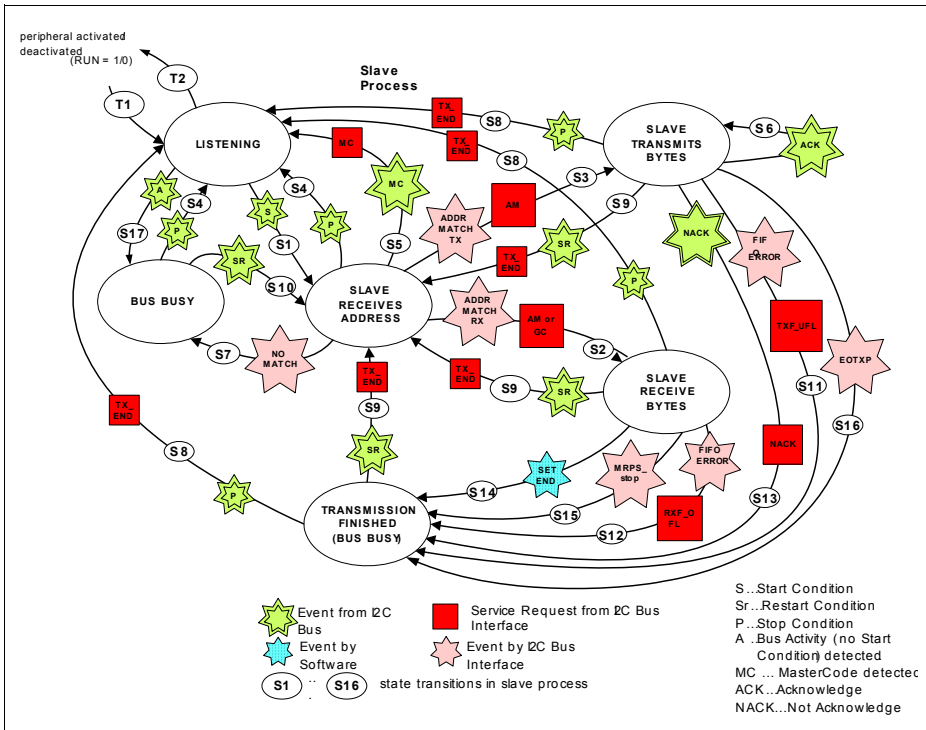


Figure 31-10 Slave Process Sub-state Machine

LISTENING: This state has been previously described (Figure 31-9). If the device is configured as slave, the module is listening only to the bus activity.

SLAVE RECEIVES ADDRESS: The module is shifting in the serial bits from the I2C-bus and acknowledges if the received byte(s) (first one in 7-bit address mode, first and second one in 10 bit address mode) match the adjusted address. During address matching the slave-receiver has the possibility to slow down the master clock by holding the SCL line low after each bit, since this can be used as a bit-by-bit handshake procedure.

Note: The address byte(s) is (are) not shifted into the FIFO.

SLAVE RECEIVES BYTES: The module is shifting in the serial bits from the I2C-bus and acknowledges when the whole data byte has been transported to the FIFO. The

Inter-Integrated Circuit Module (I2C)

slave-receiver has the possibility to slow down the master clock by holding the SCL line low after each bit since this can be used as a bit-by-bit handshake procedure.

SLAVE TRANSMITS BYTES: The slave is allowed to hold down the SCL line when preparing the data for transmission. When valid data is available the slave releases the SCL line and waits for the clock cycles from the master. The interrupt routine which proceeds when entering this state may write to the **TPSCTRL** register to initiate a transmission. This is only allowed if the bit-field BS (bus status) in register **BUSSTAT** in combination with the RnW clearly identifies the slave-transmitter mode of the peripheral. Otherwise writing to this register has no effect. First the I2C kernel gets a byte from the FIFO and puts it into a shift register. The module is then putting single data bits onto the I2C-bus and releases the SDA line after the 8th bit is sent.

*Note: Since the data flow control responsibility is task of the master device the slave should not stop transmitting bytes until the master tells it to do so. Therefore the TPS value in register **TPSCTRL** should be large enough to not run out of data bytes. In case of running out of data, the TRANSMISSION FINISHED state is reentered. The I2C data line is released to high level. The master continues receiving wrong values (0xFF).*

BUS BUSY: The bus is busy and the module is not involved in any data transfer. This is also the entry state when the bus is not free when starting up procedure is taking place (see state S17).

TRANSMISSION FINISHED (BUS BUSY): The module was involved in any data transfer. In distinction to the BUS BUSY state, the slave is still addressed by a master. Every time this state is left, caused due to a stop/restart condition, a TX_END request is generated as indication that the previous transmission has been closed.

The transitions which are connecting the several states are in detail:

S1: The module is detecting a start condition on the bus. It has to set the bit-field BS (bus status) in register **BUSSTAT** to 0_B. When this situation is occurring the I2C kernel is initialized since this indicates a new data transmission.

S2: The address which has been received matches the one stored in the address field in the **ADDRCFG** register. The RnW bit (8th bit in the first received byte) was set to 0. This means that the master wants to write to the slave. The corresponding bit RnW in register **BUSSTAT** is set to 1, i.e. the device continues working as slave-receiver. Also an acknowledge has to be set on the I2C-bus in the appropriate place (9th clock cycle). The I2C kernel sets the bit-field BS (bus status) in register **BUSSTAT** to 11_B and generates an AM (address match) request.

If the general call (GC) address matching is enabled by setting bit GCE in the **ADDRCFG** register, the device has to react also when it receives a general call (0x00). The GC request is issued and also an acknowledge has to be set on the I2C-bus when the GC is detected.

Inter-Integrated Circuit Module (I2C)

S3: The address which has been received matches the one stored in the address field in the **ADDRCFG** register and the SDA line was at a high level when the RnW bit was sent. This means that the remote master wants to read from the slave. The corresponding bit RnW in the **BUSSTAT** register has to be set to 0 (write). Also an acknowledge has to be set on the I2C-bus in the appropriate place (9th clock cycle). The I2C kernel sets the bit-field BS (bus status) in register **BUSSTAT** to 11_B and generates an AM (address match) request. The device is now working as a slave-transmitter.

S4: The device detects a stop condition on the I2C-bus and switches back to SLAVE LISTENING state. No interrupt is generated after the device was not addressed. The bit-field BS (bus status) in register **BUSSTAT** is set to 00_B (bus free).

S5: If the device is configured via bit MCE (master code enable) in the **ADDRCFG** register to react on a master code, the device can change baud rate to high-speed (Hs) mode when a remote master sends a master code. No device is allowed to acknowledge this master call. The interrupt routine which is called (MC request issued), has to handle the clearing of the interrupt bit in register **PIRQSC**. The LISTENING state is reentered. From now on the device works as slave in high-speed mode until a stop condition occurs.

S6: The slave-transmitter has received an acknowledge from the master-receiver and continues transmission.

S7: After the reception of the address bytes has been finished and no address match has taken place, the device changes to BUS BUSY state.

S8: The device has detected a stop condition on the I2C-bus after/while it was addressed. A TX_END (transmission end) request is generated, the bus is free again respectively the bit-field BS (bus status) in register **BUSSTAT** has to be set to 00_B again. The slave is now in LISTENING state.

S9: The slave has received a restart condition (repeated start condition) after/while it was addressed. A TX_END (transmission end) request is generated, the state switches back to SLAVE RECEIVES ADDRESS. The slave is reset again to a new transmission pending state (bit-field BS in register **BUSSTAT** = 01_B). When the TBAM is activated (configuration register) the slave has to remember that it was addressed before because this event may be followed by a read access of the master to this device. So the address procedure after reset can be shortened and the transition S3 occurs one byte earlier (master sends 10-bit address mode byte with RnW bit = 1 (read)).

S10: A running transmission is ended by a remote master and this device wants to continue without losing control of the I2C-bus. The slave was not in transmission and resets to receive address bytes again since the new data transmission is concerning all connected slaves.

S11: The peripheral has problems to read data bytes from the FIFO (underflow/not ready). This is indicated by generating a TXF_UFL (TX FIFO underflow) request. It also changes to TRANSMISSION FINISHED state.

Inter-Integrated Circuit Module (I2C)

*Note: The device is still addressed by a master (bit-field BS in register **BUSSTAT** = 11_B) until a stop or restart condition occurs.*

S12: The peripheral has problems to write the received byte into the FIFO (overflow/not ready). This is indicated by generating a RXF_OFL (RX FIFO overflow) request. The slave-transmitter puts a not-acknowledge on the I2C-bus to indicate its situation to the controlling master. This byte is not valid and will be discarded.

*Note: The device is still addressed by a master (bit-field BS in register **BUSSTAT** = 11_B) until a stop or restart condition occurs.*

S13: The remote master wishes to close the connection to this slave and sets a not-acknowledge on the I2C-bus. The device generates a NACK (not-acknowledge) request and changes to TRANSMISSION FINISHED state (bit-field BS in register **BUSSTAT** = 01_B). The interrupt routine can clear the appropriate bit via the corresponding bit in the **PIRQSC** register.

S14: The slave wants to stop the transmission and sets the SETEND bit in register **ENDDCTRL**. It puts a not-acknowledge on the I2C-bus after the next received byte. With this the slave gets the possibility to cancel the transmission via software intervention.

S15: The value of received bytes is equal to bit-field MRPS (maximum receive packet size) of the FIFO register **MRPSCTRL**. The FIFO part of the interface generates the signal MRPS_stop to the I2C kernel, which indicates the end of the received packet capacity. The slave produces a not-acknowledge to the master and changes to TRANSMISSION FINISHED. For more details on FIFO MRPS operation see [Section 31.2.4.5](#).

S16: The number of transmitted bytes is equal to the TPS bit-field of the FIFO register **TPSCTRL**. The FIFO controller generates the EOTXP (end of transmit packet) signal to the I2C kernel, which indicates the last byte to transmit. After the last byte the device changes to TRANSMISSION FINISHED. For more details on FIFO EOTXP operation see [Section 31.2.4.2](#).

Note: The master is not informed about this situation. Since the data flow control responsibility is task of the master device, it may continue receiving wrong values (0xFF).

S17: The module detects bus activity; no start condition has been detected before. A transmission may be currently in progress. The device changes to BUS BUSY state. This transition is performed if a start condition has been missed because the device was not activated (CONFIGURATION state).

Master Process Sub-state Machine

Figure 31-11 shows the master process sub-state machine. There are four different states in this sub-state machine:

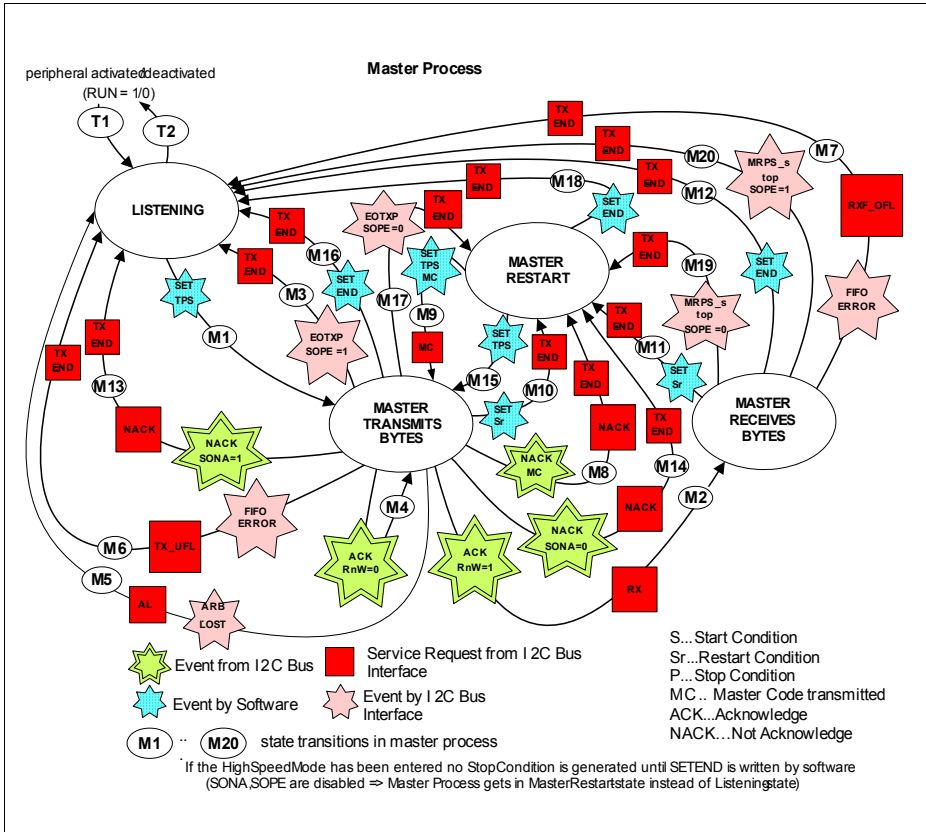


Figure 31-11 Master Process Sub-state machine

LISTENING: This state has been previously described (**Figure 31-9**). If the device is configured as master, it is listening to events either from software (writing to **TPSCTRL** register) or from the I2C-bus.

MASTER TRANSMITS BYTES: The device has started a transmission on the I2C-bus by software. The first byte after a start condition contains the address byte including the RnW bit. According to this bit (RnW bit in **BUSSTAT** register is set by hardware) the direction of the following transfer is determined. Bytes are transmitted on the I2C-bus until the FIFO is empty or a stop is initiated by software.

Inter-Integrated Circuit Module (I2C)

If bit MCE (master code enable) is set in the **ADDRCFG** register, the device has to check if the transmitted address byte is a master code.

MASTER RESTART: The device wants to restart a transmission without losing the control over the I2C-bus. The clock SCL stays low as long as no further communication is requested. The only event that should occur in this state is writing to the **TPSCTRL** register. Nevertheless the device can give up the I2C-bus control by writing to bit SETEND in the **ENDDCTRL** register. Of course, when this situation occurs, the peripheral produces a stop condition on the I2C-bus.

If bit MCE (master code enable) is set in the **ADDRCFG** register, the device has to check if the transmitted address byte is a master code.

MASTER RECEIVES BYTES: The master has ordered bytes from the slave and handles the data transfer from the I2C-bus to the FIFO. This means that the bits are shifted into a buffer register in the I2C kernel and afterwards are passed to the FIFO. Every time a byte was transferred to the FIFO successfully the master-receiver produces an acknowledge in the according position.

The state transitions in the master process are:

M1: The software has written the number of bytes which have to be transmitted over the I2C-bus into the control register **TPSCTRL**. The FIFO generates appropriate data transfer requests which transfers valid data into the FIFO. A start condition is set on the bus and the bit-field BS (bus status) in register **BUSSTAT** is set to 10_B (busy master). The first (the first and the second) byte is (are) the 7-bit (10-bit) address of the slave which shall be accessed.

If the master wishes to initiate a transition to high-speed mode, the transition is only valid in combination with the MCE (master code enable) bit in the **ADDRCFG** register. The software has to write the master code into the FIFO, this means also the value of the **TPSCTRL** is predetermined with 1.

M2: The master receives an acknowledge from slave and the bit RnW in the **BUSSTAT** register was set to 1 during the addressing phase. A slave has been addressed successfully as transmitter, the state moves to MASTER RECEIVES BYTES. An RX (Receive) request is generated to distinguish between transmit- and receive-FIFO requests in non flow controller mode.

M3: The valid data packet from the FIFO was transferred successfully to the I2C-bus and the FIFO indicates this by the EOTXP (end of transmit packet) signal. If the RnW bit was set to 0 (writing to a slave) and bit SOPE (stop on packet end) in register **ADDRCFG** was set to 1, the transfer is closed by putting a stop condition on the bus. After this stop condition has been detected the peripheral indicates this successful transfer to the software via the TX_END (transmission end) request. The bus status bit-field is set to 0x0 (bus free).

Inter-Integrated Circuit Module (I2C)

M4: After the master transmits a byte it releases the SDA line to allow the addressed slave to pull down the SDA line (acknowledge). If this event occurs (and RnW in register **BUSSTAT** was set to 0), the master can continue to transfer bytes on the I2C-bus.

M5: The SDA line level is not equal to the one that is set by the master. This means that another master also tries to control the I2C-bus. This is indicated by the AL (arbitration lost) request and the device switches to the LISTENING state. The interrupt routine has to clear bit AL in the **PIRQSC** register. The Arbitration lost indication is not executable when the device works in high-speed mode (no arbitration procedure in high-speed mode since there is only one master remaining). The arbitration process must be operating at each bit which is sent by the master! Before interrupting the CPU, the bit-field BS (bus status) in register **BUSSTAT** is set to 01_B.

Note: It could be the case that the other master tried to address this device at the same time. Since this device could not check the transmitted address bytes till arbitration lost, no acknowledge is generated. The other master has to address this device once more.

M6: If the master wants to transmit bytes but the FIFO is empty (underflow), it has to generate a TXF_UFL (TX FIFO underflow) request. The bit-field BS (bus status) in register **BUSSTAT** is reset to 00_B, and a TX_END (transmission end) request is generated after the master has put a stop condition on the bus.

M7: The I2C kernel wants to transfer bytes to the FIFO but without success. Then it has to produce an RXF_OFI (RX FIFO overflow) request. The current byte is not-acknowledged and a stop condition is put on the bus (bit-field BS in register **BUSSTAT** is set to 00_B and a TX_END (transmission end) request is generated). (

M8: If master code has been transmitted by the device and a NACK (not-acknowledge) has been received, the MASTER RESTART state is entered. The clock SCL stays low as long as no further communication is requested. This is done by writing bit-field TPS in register **TPSCTRL** (see transition M9).

M9: A new communication in high-speed mode is started by writing bit-field TPS in register **TPSCTRL** when a master code has been previously transmitted (M8). The switch into high-speed mode (MC request) is done as soon as clock and data are released to high (see **Figure 31-12**). State MASTER TRANSMITS BYTES is entered. During high-speed mode, all fast and standard mode devices must be disconnected from the bus. This can be done in the MC request.

M10: Bit SETRSC has been set during the transmission of the data (the software wants to change direction/slave and therefore writes to the **ENDCTRL** register). The I2C kernel resets and changes to MASTER RESTART state and a TX_END (transmission end) request is generated. Therefore, the master does not lose the control of the I2C-bus after the transmission has ended. The master stops transmission after the current byte was acknowledged/not acknowledged. Valid data in the FIFO are discarded. In this situation the FIFO is flushed to produce an initial situation.

Inter-Integrated Circuit Module (I2C)

M11: If the master wants to change direction/slave immediately in this state, it has to write to bit SETRSC in the **ENDCTRL** register. The master puts a not-acknowledge on the appropriate place during the current transmitted byte. This indicates the slave-transmitter to stop transmission. The master switches to MASTER RESTART state; a TX_END (transmission end) request is generated.

M12: By sending a not-acknowledge on the bus, the master indicates the slave an upcoming stop of the transmission. This is established by writing to bit SETEND in register **ENDCTRL**. This end of transmission is indicated to the FIFO which then transports the remaining valid data to the memory (indicating the data end either by a LSREQ or a LBREQ). Stop condition and TX_END (transmission end) request are generated. The LISTENING state is entered; the bit-field BS (bus status) in register **BUSSTAT** is set to 00_B.

M13: If the master is receiving a not-acknowledge and bit SONA (stop on not-acknowledge) in the **ADDRCFG** register is 1, it produces a stop condition, resets the bit-field BS (bus status) in register **BUSSTAT** and issues a NACK (not-acknowledge) request and a TX_END (transmission end) request. The FIFO is flushed; remaining data in the FIFO stages are discarded.

M14: If bit SONA (stop on not-acknowledge) in the **ADDRCFG** register is 0, the peripheral wants to remain the controlling master of the I2C-bus after receiving a not-acknowledge. The next state is MASTER RESTART. A NACK (not-acknowledge) request and a TX_END (transmission end) request indicate this event to the software. The FIFO transfers the valid data to the memory.

M15: The master decided to transfer data on the bus again and writes to the **TPSCTRL** register. The first byte is interpreted as address byte (the 8th bit indicates RnW). The appropriate data transfer requests are activated by the FIFO controller. After the SCL line is high again (slave can stretch the low period) the master is allowed to put a restart condition onto the bus and is again in the MASTER TRANSMITS BYTES state starting to transmit bytes again.

M16: The software has written the SETEND bit in the **ENDCTRL** register. The peripheral stops transmission of bytes after the current byte has been written on the bus. The I2C kernel flushes the FIFO since the remaining data in the FIFO is invalid. A TX_END (transmission end) request is generated.

M17: The FIFO indicates the end of the data packet and the peripheral was configured to go into MASTER RESTART state after that event. Since the data transfer has ended as intended a TX_END (transmission end) request is produced.

M18: The master has no data to send anymore but still controls the I2C-bus. Therefore it can use the SETEND bit to give up the I2C-bus control by setting a stop condition. A TX_END (transmission end) request is generated.

M19: A value different to zero is programmed to bit-field MRPS (maximum receive packet size) of the **MRPSCTRL** register. If the number of received bytes is equal to the MRPS bit-field, the FIFO produces the MRPS_stop signal. When bit SOPE (stop on

Inter-Integrated Circuit Module (I2C)

packet end) in the **ADDRCFG** register is set to 0, the master generates a not-acknowledge and changes to MASTER RESTART state, so the master does not lose the control over the I2C-bus. A TX_END (transmission end) request is generated.

M20: (See also M19.) The FIFO produces the MRPS_stop signal which indicates that the desired amount of received data bytes is reached. The peripheral generates a not-acknowledge and changes to LISTENING state because bit SOPE (stop on packet end) in the **ADDRCFG** register is set to 1. Of course, the interface produces a stop condition; so the I2C-bus is free again (change also the bus status bit-field!). A TX_END (transmission end) request is also generated. The LBREQ or the LSREQ indicate the completed reception of data. For more details on FIFO MRPS operation see [Section 31.2.4.5](#).

Inter-Integrated Circuit Module (I2C)

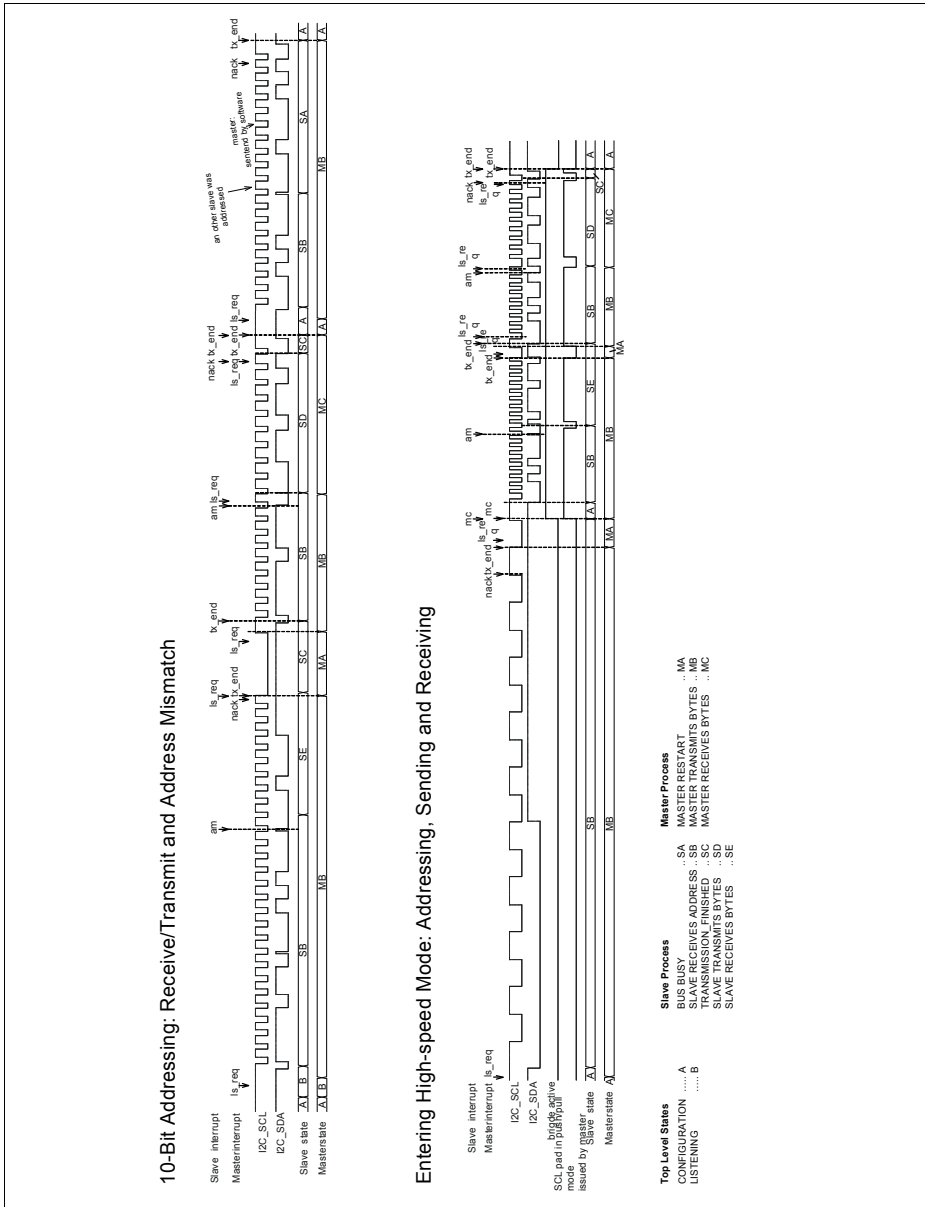


Figure 31-12 Connection Examples including Requests in Master and Slave Mode

Inter-Integrated Circuit Module (I2C)

31.2.4 FIFO Operation

The I2C module uses a FIFO between its kernel and the bus, in order to adapt the character processing speed of the peripheral to the transfer rate of the bus system. The FIFO has a transmission state and a reception state. **Figure 31-13** shows the bidirectional half duplex TX/RX FIFO unit with the input and output lines and with the corresponding FIFO registers.

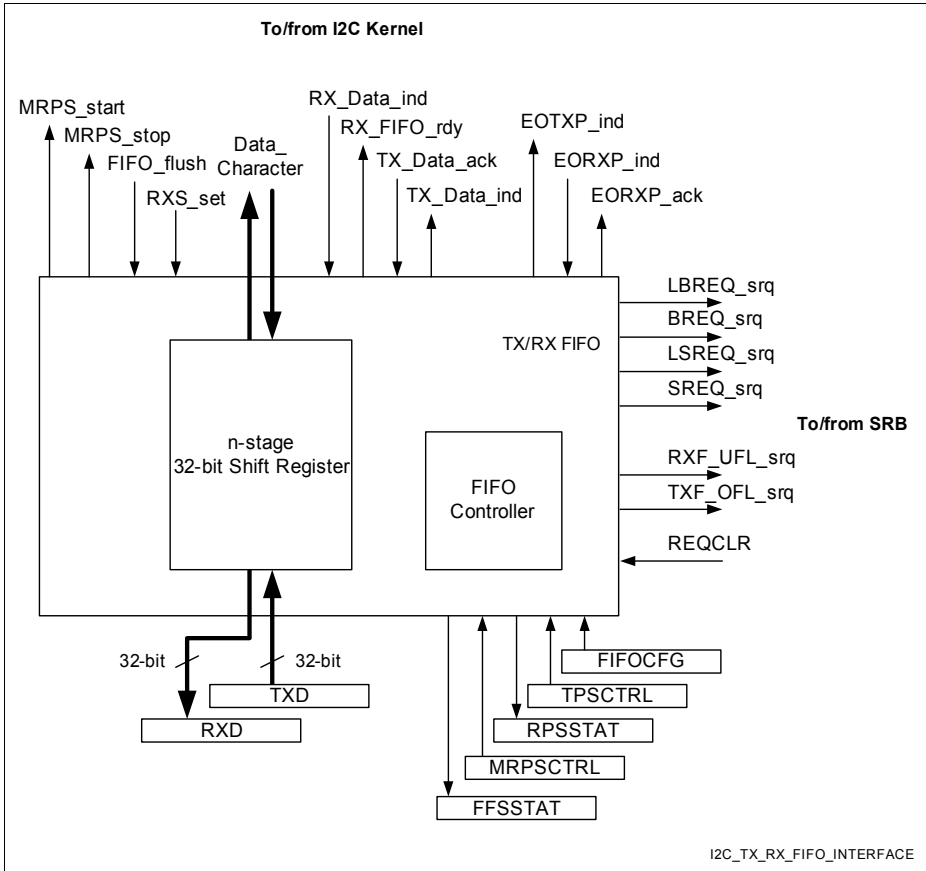


Figure 31-13 Bidirectional Half Duplex TX/RX FIFO

31.2.4.1 Data Transmission

The software can move transmit data to the FIFO by writing to the Transmission Data Register **TXD**. Moving data to the FIFO can be done according to the data transfer request signals generated by the FIFO controller as described in [Section 31.2.4.2](#). But with the register **FFSSTAT**, which indicates the number of full FIFO stages, the data transfer to the FIFO can be handled completely without data transfer requests. The data alignment within the FIFO is described in [Section 31.2.4.3](#). The FIFO shifts data to the I2C kernel by means of the handshake lines `tx_data_ind` and `tx_data_ack`.

With the signal `FIFO_flush` the I2C kernel is able to clear the FIFO content. But if a data transfer request (see [Section 31.2.4.2](#)) is pending, then the FIFO is not cleared until the corresponding signal `REQCLR` has been set by software.

The I2C-bus interface should know when it is receiving the last character of a data packet from the FIFO. Therefore the software should define a transmit packet size in bit-field `TPS` of register **TPSCTRL** and then the FIFO controller indicates the last character of the packet to the I2C kernel by setting the `EOTXP` (end of transmit packet) signal.

Since `TPS` is double buffered the software can write the size of the next packet into `TPS` as soon as the `TPS` value of the current packet has been loaded into the `TPS` counter, i.e. reading `TPS` returns 0. This happens a few clock cycles after the previous packet has been transmitted.

Also the characters of the next packet can be moved to the FIFO immediately after the current packet characters as indicated in [Figure 31-14](#). If byte or half word alignment is used as described in [Section 31.2.4.3](#), then the characters of two different packets must not be aligned in a single stage.

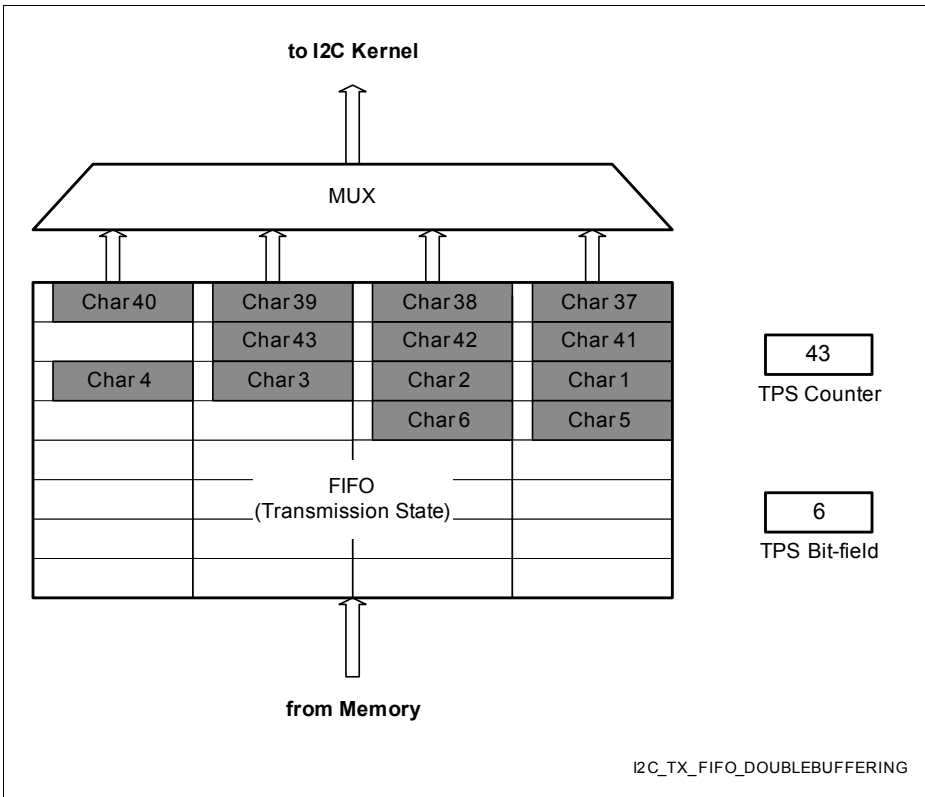


Figure 31-14 FIFO Containing Data of Two Different Transmit Packets

Error Handling

- If a TX FIFO overflow occurs (i.e. the software writes data too fast to the FIFO), then the new incoming character is discarded and a TX FIFO overflow interrupt request is generated.
- If a TX FIFO underflow occurs (i.e. the I2C kernel tries to read from an empty FIFO), then the I2C kernel generates a TX FIFO underflow interrupt request.

31.2.4.2 Transmit Request Generation

For the following it is assumed that the FIFO is used as flow controller, selected via bit TXFC of the **FIFOCFG** register.

In this case the bit-field TPS of the **TPSCTRL** register is not only used for the generation of the EOTXP (end of transmit packet) signal (see **Section 31.2.4.1**), but also to initiate

Inter-Integrated Circuit Module (I2C)

the whole data transfer. I.e. the software indicates that it wants to transmit a data packet by writing the packet size to the bit-field TPS. Then the FIFO unit asserts burst requests BREQ until the amount of data still to be transferred is less than or equal to the burst size set in bit-field TXBS of register **FIFOCFG**. At this point, if the remaining data is equal to the burst size, then a last burst request LBREQ is issued. Otherwise, single requests SREQ and a last single request LSREQ will be issued.

Notes

1. *The next request is only generated if previous request was cleared.*
2. *Pre-loading TPS with the packet size of the next frame, before the current packet has been transmitted completely, enables a continuous data throughput (e.g. of audio data).*
- 3.
4. *If the FIFO is not used as flow controller, the I2C-bus interface should use bit-field TPS of the TPSCCTRL register to set the EOTXP signal. A burst request BREQ is generated each time the number of empty FIFO stages is equal or greater than the burst size set in bit-field TXBS and the previous request was cleared. No single requests SREQ and LSREQ will be issued.*

31.2.4.3 Transmit Data Alignment

Depending on bit-field TXFA (TX FIFO Alignment) in register **FIFOCFG**, the FIFO can deal with byte aligned, half word aligned or word aligned characters (or even more) as it is described in the following.

Note: Byte alignment is synonymous with character alignment, half word alignment with character alignment of two data characters, and word alignment with character alignment of four characters.

Byte Aligned TX Characters

If the TX characters are byte aligned, then the FIFO extracts the bytes and shifts them to the I2C kernel via a multiplexer as it is shown in **Figure 31-15**.

Since the number of characters of the packet to be transmitted is not necessarily a multiple of 4, the last word must be filled up with dummy bytes if necessary. The FIFO transmits only the valid bytes of the last word depending on the setting of the bit-field TPS of the **TPSCTRL** register.

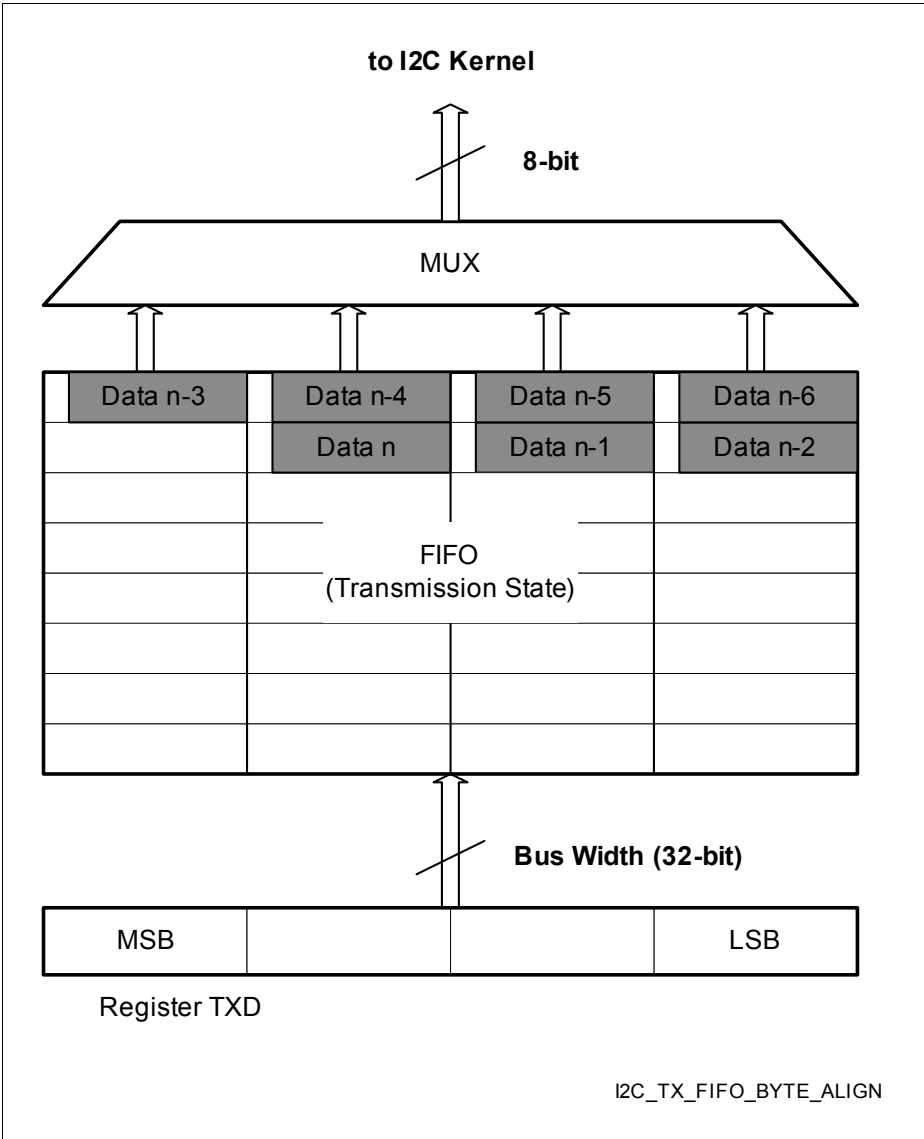


Figure 31-15 FIFO in Transmission State with Byte Aligned Data

Inter-Integrated Circuit Module (I2C)**Half Word Aligned**

If the TX characters are half word aligned, then the FIFO extracts the half words and shifts them to the I2C kernel via a multiplexer as it is shown in [Figure 31-16](#).

Note: The I2C kernel extracts the right data bits from the half word it receives from the FIFO.

Since the number of characters of the packet to be transmitted is not necessarily a multiple of 2, the last word must be filled up with a dummy half word if necessary. The FIFO transmits only the valid half words of the last word depending on the setting of the bit-field TPS.

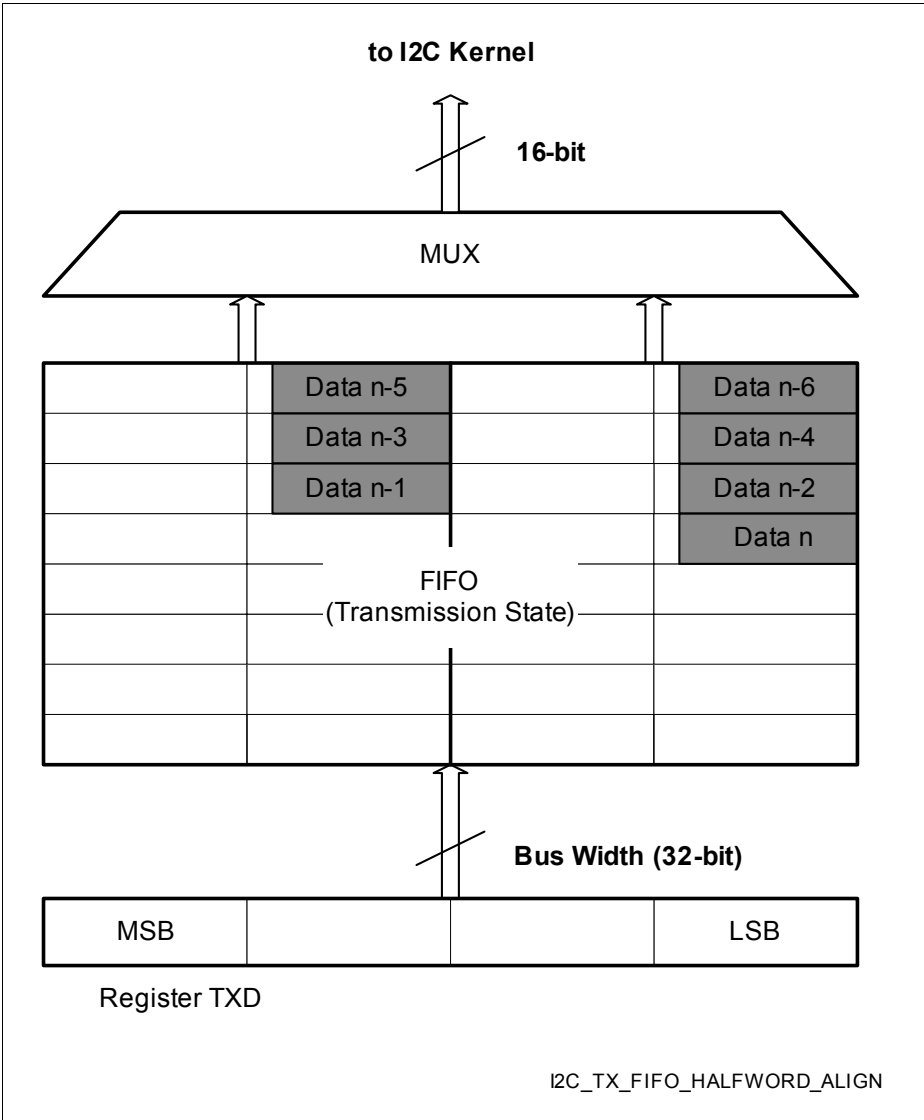


Figure 31-16 FIFO in Transmission State with Half Word Aligned Data

Inter-Integrated Circuit Module (I2C)

Word Aligned

If the TX characters are word aligned, then the FIFO shifts them to the I2C kernel as it is shown in **Figure 31-17**.

Note: The I2C kernel extracts the right data bits from the word it receives from the FIFO.

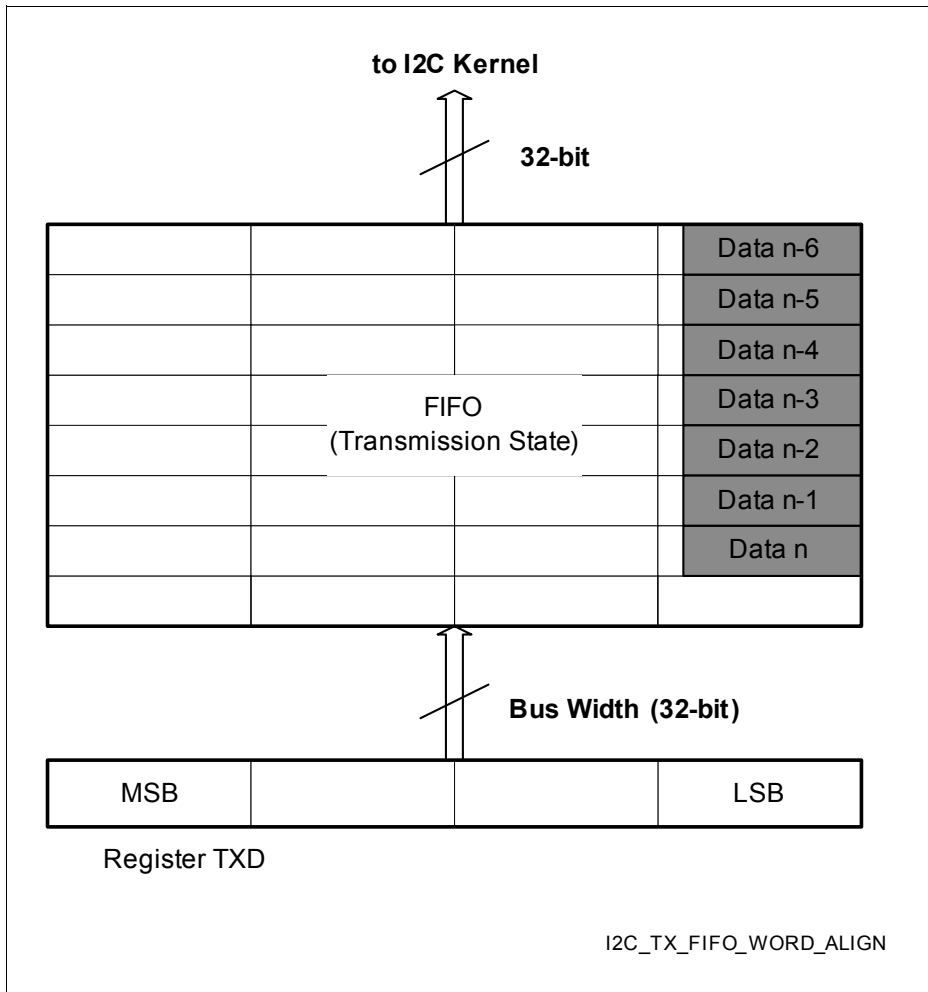


Figure 31-17 FIFO in Transmission State with Word Aligned Data

31.2.4.4 Data Reception

The software can read received data from the FIFO by reading the Reception Data Register **RXD**. Reading data from the FIFO should be done according to the data transfer request signals generated by the FIFO controller. The register **FFSSTAT** indicates the number of filled FIFO stages. The data alignment within the FIFO is described in [Section 31.2.4.6](#). The I2C kernel shifts data into the FIFO by means of the handshake signals **RX_FIFO_rdy** and **RX_Data_ind**.

Before the first reception starts, the I2C kernel has to set the **RXS_set** line. This causes the assertion of **RX_FIFO_rdy** to indicate the I2C kernel that one data character can be written into the FIFO.

With the signal **FIFO_flush** the I2C kernel is able to clear the FIFO content. But if a data transfer request is pending and the FIFO is configured as flow controller, then the FIFO is not cleared until the corresponding signal **REQCLR** has been set by software.

Error Handling

- If the RX FIFO is completely full (i.e. the software reads out the data too slow), then the I2C kernel generates an RX FIFO overflow interrupt request.
- If an RX FIFO underflow occurs (i.e. the software reads from empty FIFO), then an RX FIFO underflow interrupt request is generated.
- If an error (including the FIFO overflow condition described above) is detected in the I2C kernel and the current reception is stopped, then it also has to generate an **EORXP_ind** signal, so that the remaining characters in the FIFO can be moved out by means of data transfer requests.

31.2.4.5 Receive Request Generation

If the whole data of the current received packet has been moved into the buffer, then the I2C kernel will set the **EORXP_ind** (end of receive packet indication) signal. The received characters, which are shifted in the FIFO, are counted by an **RPS** (received packet size) counter. If an **EORXP_ind** occurs, then the content of the **RPS** counter is moved to the register **RPSSTAT** and the counter is reset. If data alignment in the FIFO is used, then the **RPSSTAT** value can be used to check for valid characters in the last read word (see [Section 31.2.4.6](#)).

For the following it is assumed that the FIFO is used as flow controller, selected via bit **RXFC** of the **FIFOCFG** register.

The FIFO unit asserts burst requests **BREQ** each time the FIFO filling level is greater than the burst size set in bit-field **RXBS** of register **FIFOCFG**. An **EORXP_ind** causes the FIFO unit to assert burst requests **BREQ** until the amount of data still to be transferred is less than or equal to the burst size. At this point, if the remaining data is equal to the burst size, a last burst request **LBREQ** is issued. Otherwise, single requests **SREQ** and a last single request **LSREQ** will be issued.

Inter-Integrated Circuit Module (I2C)

When the packet has been shifted out of the FIFO by software, the FIFO controller provides the EORXP_ack (end of receive packet acknowledge) signal to the I2C kernel, which allows the kernel to set the next EORXP_ind.

The next received packet may be shifted into FIFO immediately after the current packet, so that the FIFO contains data of two different packets. But the EORXP_ind of the next packet must not be asserted before the EORXP_ind of the current packet has been acknowledged. The RPS counter is counting the characters of the next packet, while in the register **RPSSTAT** the packet size of the current packet remains readable. The first character of the new packet is always shifted in a new FIFO stage as indicated in **Figure 31-18**. (The FIFO data alignment is described in **Section 31.2.4.6**.)

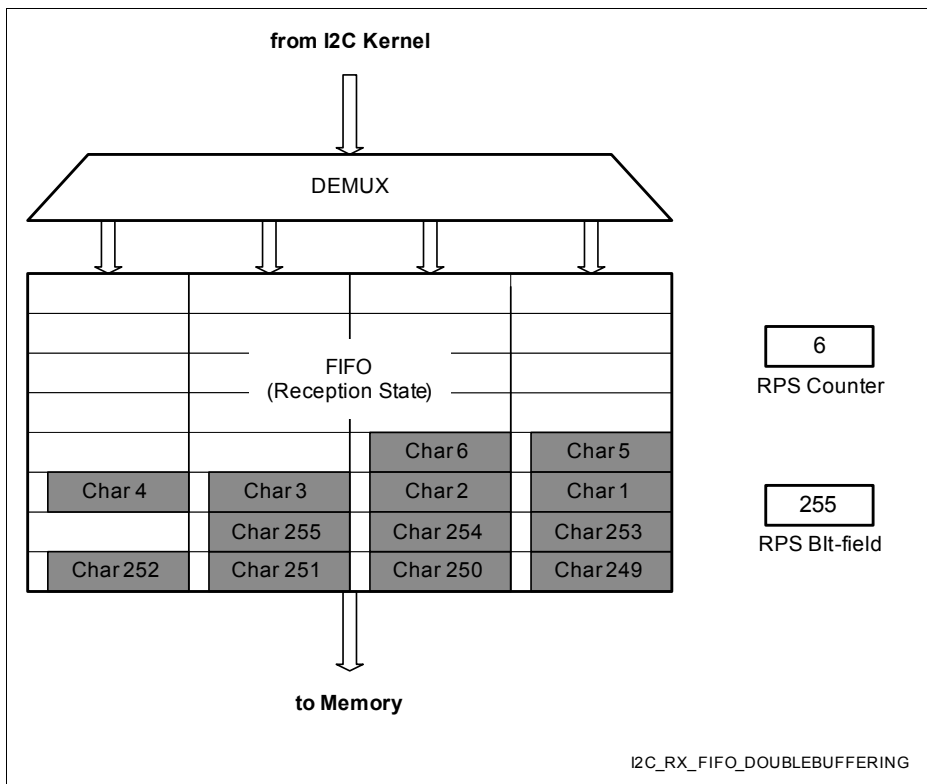


Figure 31-18 FIFO Containing Data of Two Different Receive Packets

If the number of the received characters of one packet is equal to the maximum received packet size, which is defined by bit-field MRPS in register **MRPCTRL**, then the FIFO controller gets into the state, where it generates LBREQ or SREQ and LSREQ. If the I2C

Inter-Integrated Circuit Module (I2C)

kernel generates an EORXP_ind in this state, then this indication is valid for the received packet and the FIFO accepts no more characters from the I2C kernel until the current packet has been moved out of the FIFO or the start of a new packet has been detected by asserting RXS_set.

The bit-field MRPS is double buffered, i.e. the software can write the MRPS value of the next packet as soon as the current MRPS value has been loaded, i.e. reading MRPS returns 0. This happens a few clock cycles after all data of the previous packet have been read from the FIFO. If an MRPS overflow occurs (one character before the last character is received), then the MRPS_stop trigger to the I2C kernel is generated. But if the next MRPS value is written to MRPS before the last data character of current frame has been completely received, then no MRPS_stop is generated. The I2C-bus interface will use the MRPS_stop trigger to stop the reception of the corresponding data packet.

But the MRPS bit-field can also be used to initiate a reception, when the peripheral is master-receiver. Therefore writing to MRPS generates a trigger MRPS_start to the I2C kernel. The next MRPS value written to MRPS generates only an MRPS_start when an MRPS_stop was generated before.

The MRPS features can only be used if the FIFO is the flow controller.

Notes

- 1. When the MRPS control feature of the FIFO is used to limit the packet size of a continuous incoming data stream (MRPS is only written once and left unchanged), characters exceeding MRPS are discarded. The FIFO can not handle more than two packets. When the FIFO is filled with data characters of two packets then a further RXS_set will be overseen by the FIFO. Additionally the FIFO does not set the RX_FIFO_rdy.*
- 2. If the FIFO is not used as flow controller, a single request SREQ is always generated as soon as the readable FIFO stage is filled up with characters. Additionally, a burst request BREQ is generated as soon as the number of filled FIFO stages is equal to or greater than the burst size set in bit-field RXBS of register FIFOCFG. When the I2C kernel sets EORXP_ind, the FIFO controller provides the EORXP_ack (end of receive packet acknowledge) signal to the I2C kernel, which allows the kernel to set the next EORXP_ind.*

31.2.4.6 Receive Data Alignment

Depending on bit-field RXFA (RX FIFO Alignment) in register **FIFOCFG**, the FIFO can deal with byte aligned, half word aligned or word aligned characters (or even more) as it is described in the following.

Byte Aligned

If the RX data should be byte aligned, then the characters from the I2C kernel are aligned and shifted to the FIFO by a demultiplexer as it is shown in **Figure 31-19**.

Since the number of characters of the received packet is not necessarily a multiple of 4, the upper bytes of the last word can be invalid. The software has to check for invalid bytes of the last word by means of the bit-field RPS (received packet size) in register **RPSSTAT**.

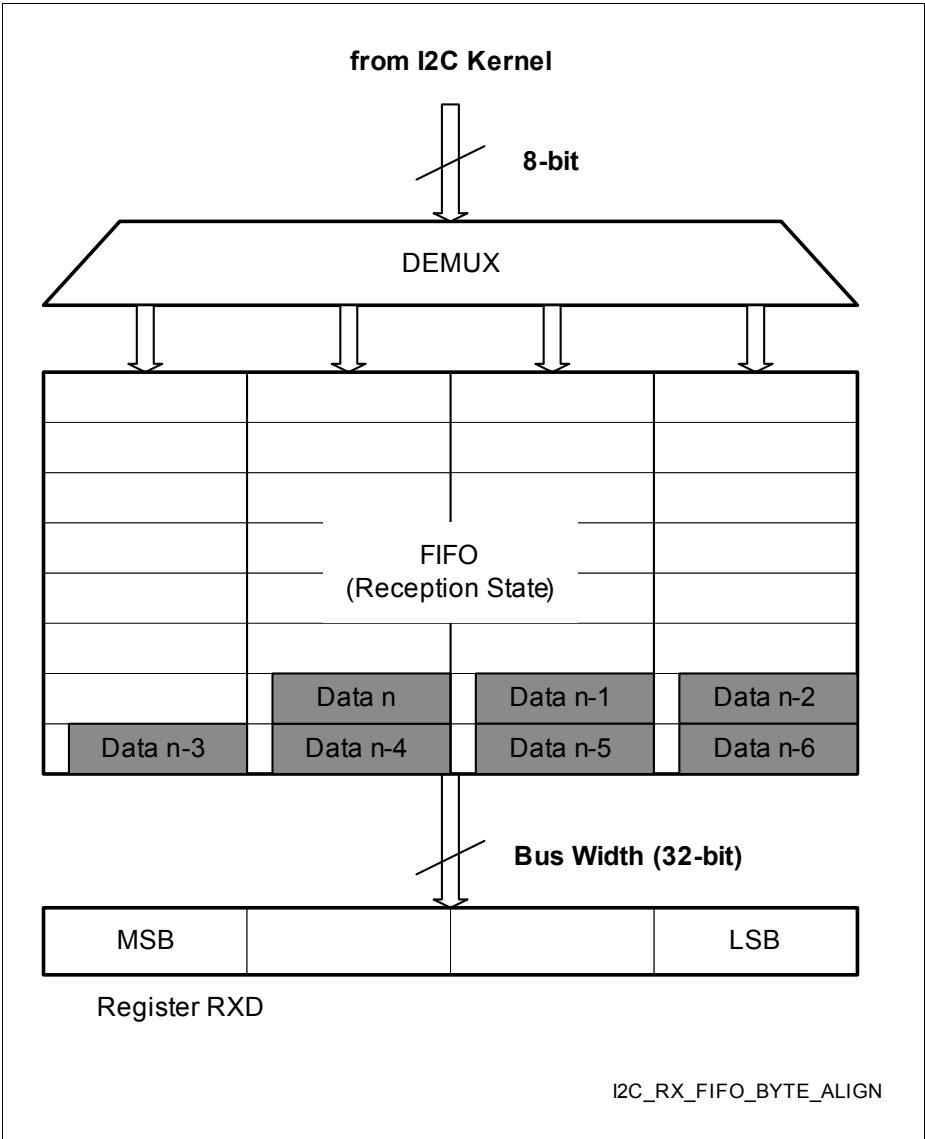


Figure 31-19 FIFO in Reception State with Byte Aligned Data

Inter-Integrated Circuit Module (I2C)**Half Word Aligned**

If the RX data should be half word aligned, then the characters from the I2C kernel are aligned and shifted to the FIFO by a demultiplexer as it is shown in [Figure 31-20](#).

Note: The I2C kernel fills up the half words with dummy bits.

Since the number of characters of the received packet is not necessarily a multiple of 2, the upper bytes of the last word can be invalid. The software has to check for invalid half words of the last word by means of the bit-field RPS (received packet size) in register [RPSSTAT](#).

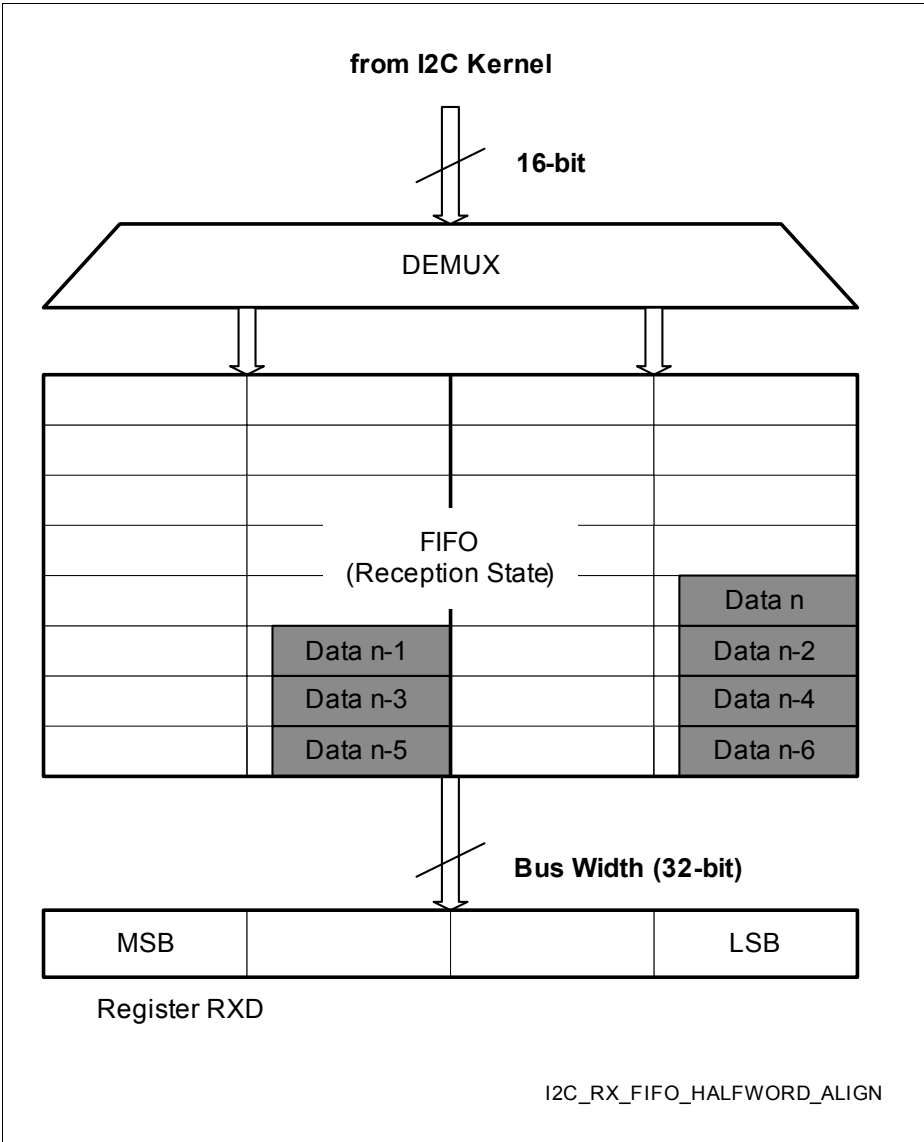


Figure 31-20 FIFO in Reception State with Half Word Aligned Data

Inter-Integrated Circuit Module (I2C)

Word Aligned

If the RX data should be word aligned, then the characters from the I2C kernel are shifted to the FIFO as it is shown in **Figure 31-21**.

Note: The I2C kernel fills up the words with dummy bits.

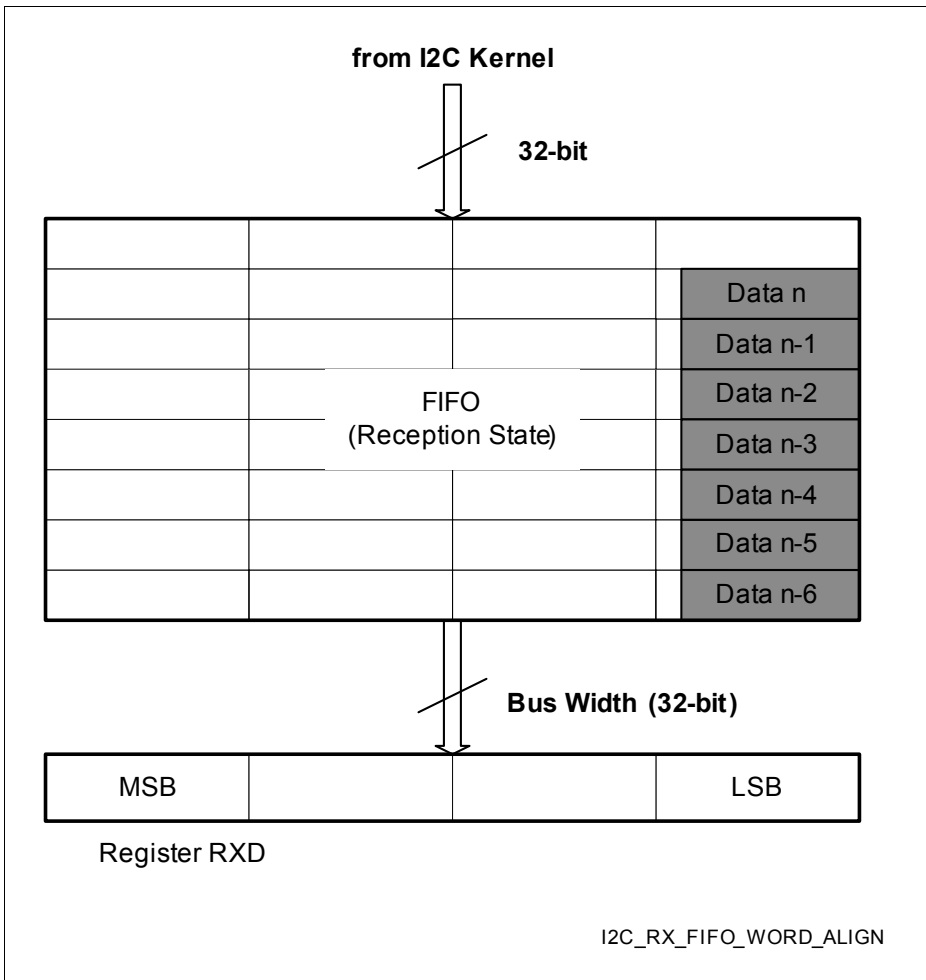


Figure 31-21 FIFO in Reception State with Word Aligned Receive Data

31.2.4.7 Switching between Transmission and Reception

Initially, the FIFO is in the TX state, so that the software can initiate a TX transfer simply by writing data to the bit-field TXD. If the I2C kernel wants the FIFO to switch to the RX state in the meantime, then it can set the RXS_set line and even if transmission is ongoing the TX transfer is aborted. This causes the FIFO to flush its content, switch to the RX state and reset the received packet size counter. If the software wants to write to the register TXD when the FIFO is in RX state, then this causes a bus error. As soon as the FIFO has received an EORXP_ind signal from the kernel and the software has moved all RX characters out of the FIFO via the register RXD, the FIFO automatically switches back into the TX state.

If the software writes to the register TPSCTRL when the FIFO is in the RX state, the TPS value is pending until the FIFO returns to the TX state. Then the transmission is initiated, if the FIFO is the flow controller.

If the FIFO is flow controller and the software writes to the register MRPSCTRL, then an MRPS_start is generated independent of the state of the FIFO.

With the signal FIFO_flush the I2C kernel is able to clear the FIFO content. But if a data transfer request (xREQ) is pending, then the FIFO is not cleared until the corresponding signal REQCLR has been set by software. If the FIFO has been cleared in the RX state, then it switches back to the TX state afterwards.

31.2.5 Service Request Block Operation

The SRB (Service Request Block) of the I2C module is used to prepare the interrupt and data transfer requests for the interrupt controller.

31.2.5.1 Overview of Service Requests

The I2C service requests are partially combined in the SRB (see [Section 31.3.4](#) and [Section 31.3.5](#)). [Figure 31-22](#) shows an overview of the service requests. [Table 31-2](#) provides a list of all service requests of the I2C module.

Inter-Integrated Circuit Module (I2C)

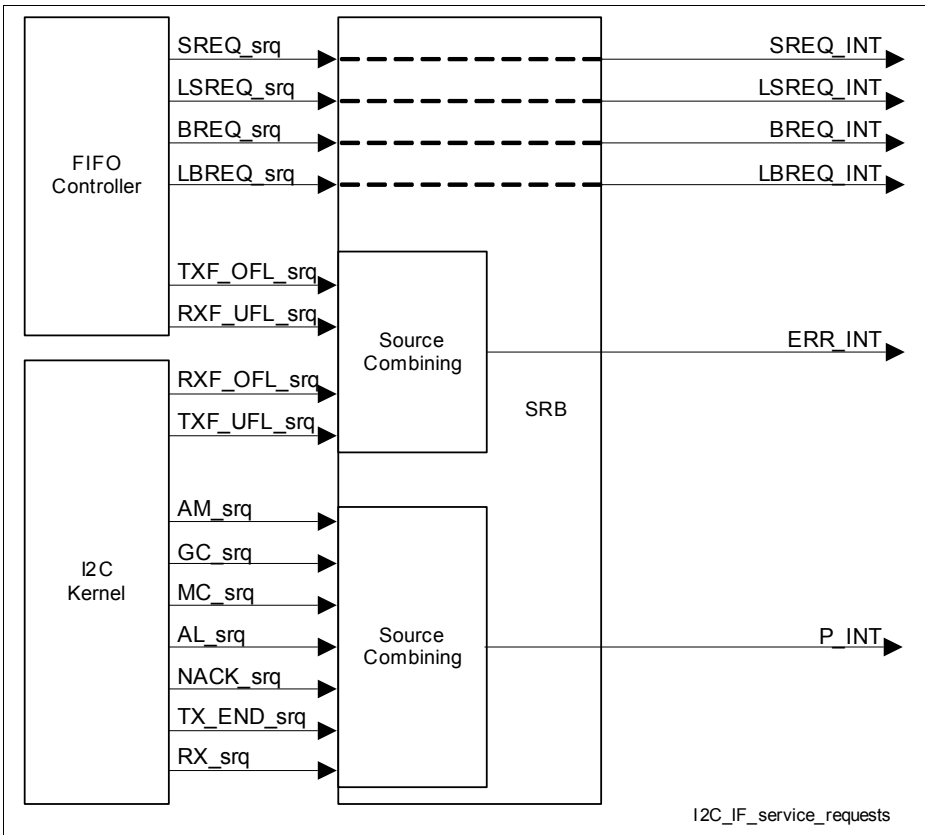


Figure 31-22 Overview of I2C Module Service Requests

Table 31-2 I2C Module Service Requests

Service Request	InterruptRequest	Description
BREQ_srq	BREQ_INT	Burst Data Transfer Request FIFO requests a transfer of a programmed burst number of words from/to the memory.
LBREQ_srq	LBREQ_INT	Last Burst Data Transfer Request FIFO requests a last burst transfer from/to the memory.

Inter-Integrated Circuit Module (I2C)
Table 31-2 I2C Module Service Requests (cont'd)

Service Request	InterruptRequest	Description
SREQ_srq	SREQ_INT	Single Data Transfer Request FIFO requests a single transfer of a word from/to the memory.
LSREQ_srq	LSREQ_INT	Last Single Data Transfer Request FIFO requests a last single transfer from/to the memory.
TXF_OFL_srq	ERR_INT	TX FIFO Overflow Request FIFO has detected a TX FIFO overflow.
TXF_UFL_srq	ERR_INT	TX FIFO Underflow Request I2C kernel has detected a TX FIFO underflow. The transmission is finished after the current byte. A stop condition is generated if the kernel works as master. The kernel changes to listening state.
RXF_OFL_srq	ERR_INT	RX FIFO Overflow Request I2C kernel has detected an RX FIFO overflow and the incoming character is discarded. The kernel puts a not-acknowledge on the bus and changes to listening state. A stop condition is generated if the kernel works as master.
RXF_UFL_srq	ERR_INT	RX FIFO Underflow Request FIFO has detected an RX FIFO underflow.
AM_srq	P_INT	Address Match Request Device (master/slave) has been addressed by a remote master (also indicated in bit-field BS in register BUSSTAT).
GC_srq	P_INT	General Call Request When the general call matching process is activated this interrupt indicates that another master has put a general call on the I2C-bus.
MC_srq	P_INT	Master Code Request When the master code matching process is activated this interrupt indicates the appearing of a master code on the I2C-bus issued by a remote master. The request is generated after a not-acknowledge and the clock is released to high again.

Inter-Integrated Circuit Module (I2C)
Table 31-2 I2C Module Service Requests (cont'd)

Service Request	InterruptRequest	Description
AL_srq	P_INT	Arbitration Lost Request Arbitration is lost after the device has tried to start a transmission on the I2C_bus.
NACK_srq	P_INT	Not-acknowledge Received Request Not-acknowledge received when working as transmitter (i.e. RnW bit is 0).
TX_END_srq	P_INT	Transmission End Request In master mode this event is produced by the I2C kernel to indicate that the transmission of the current packet has ended properly after the stop condition has been put on the I2C-bus or MASTER RESTART state has been entered. (At this point, a restart condition can be generated or the connection can be finished by generating a stop condition). This request is created in master mode at any stop condition to indicate that the bus is free again and it could be obtained. In slave mode this event is produced by the I2C kernel if it was addressed by a master and the current transmission was terminated by a stop or restart condition.
RX_srq	P_INT	Receive Mode Request I2C kernel indicates to the FIFO switching from transmit to receive mode. When FIFO is operating in non flow controller mode, this service request can be used to distinguish between transmit and receive FIFO data requests.

The generation of these events is visualized in the state machine [Figure 31-9](#), [Figure 31-10](#) and [Figure 31-11](#). The timing is shown in [Figure 31-12](#).

31.2.5.2 Interrupt Service Request Structure

The I2C module provides level based service request lines, which can be processed by an interrupt controller. The service requests must be cleared in the interrupt service routine. For test purposes, all service requests can also be set by SW via a register bit. All service requests can be masked within the peripheral. Furthermore, requests that are not necessarily mutually exclusive are combined in order to reduce the number of request lines.

Flow of Data Transfer Requests

Figure 31-23 shows the flow of a data transfer request, which comes from the FIFO controller.

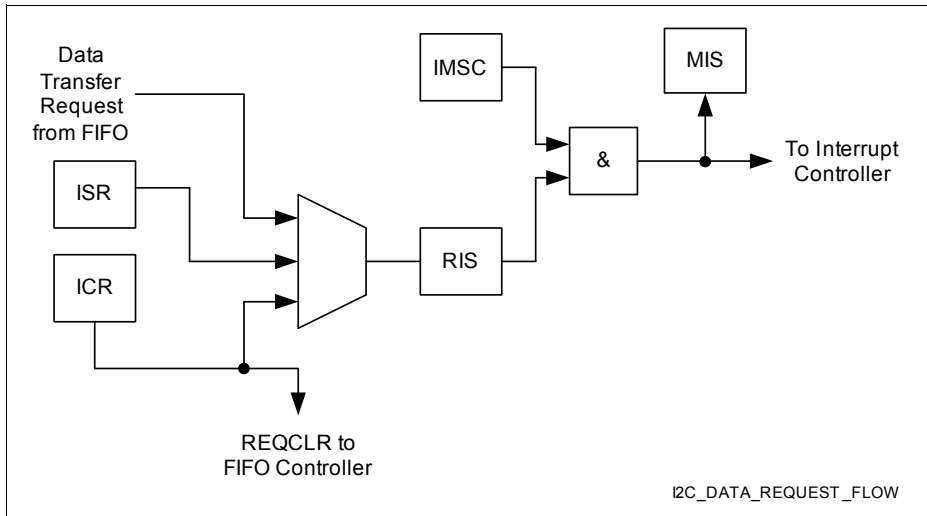


Figure 31-23 Data Transfer Request Flow

A data transfer request sets the corresponding status bit in the Raw Interrupt Status Register **RIS**. The status bit can also be set by writing 1 to the corresponding bit in the Interrupt Set Register **ISR**. It will be cleared by writing 1 to the corresponding bit in the Interrupt Clear Register **ICR**.

The Interrupt Mask Control Register **IMSC** enables or disables the requests to the interrupt controller. The Masked Interrupt Status Register **MIS** contains the current masked values of the requests.

If a request is disabled via register **IMSC** while the request is active, the request will be removed but only until the **IMSC** bit is set again, unless the request has been cleared in the meantime.

If a request is disabled via register **IMSC** and the source becomes active, then the request bit will only be set in the **RIS** register. If the corresponding **IMSC** bit is later enabled, the request will consequently become active in the **MIS** register.

Before enabling a request, it is good practice to always first clear the corresponding bit in the **RIS** register via **ICR**.

Multiple Source Interrupt Requests

For error and protocol interrupt requests which have multiple sources, the interrupt structure is extended. Additional register sets are implemented within the SRB.

Figure 31-24 gives an overview of the combining of several request sources to a single request.

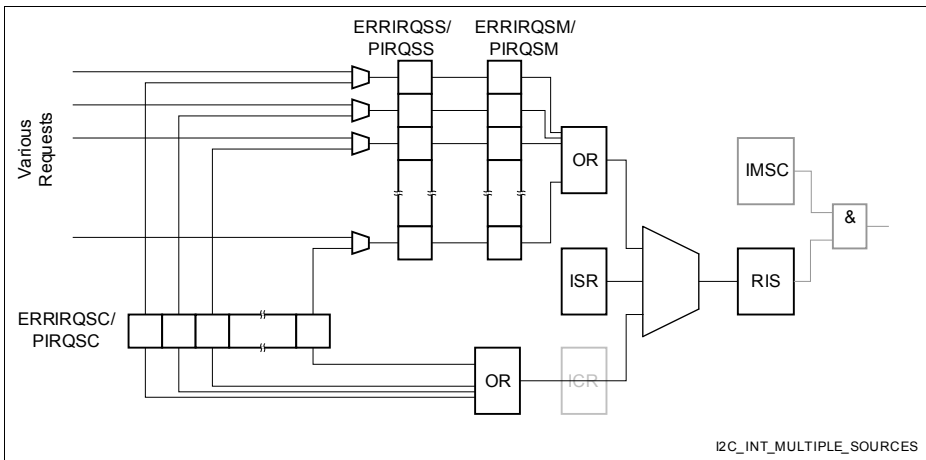


Figure 31-24 Interrupt Request with Multiple Sources

An interrupt request sets the corresponding raw status bit in the Interrupt Request Source Status Register **ERRIRQSS** (for error) or **PIRQSS** (for protocol).

The status bits can be cleared via the Interrupt Request Source Clear Register **ERRIRQSC** (for error) or **PIRQSC** (for protocol). If no further error/protocol request sources are active, the whole error/protocol interrupt request is cleared. This register replaces the functionality of the bit I2C_ERR_INT (for error) or bit I2C_P_INT (for protocol) in the Interrupt Clear Register **ICR**.

The Interrupt Request Source Mask Register **ERRIRQSM** (for error) or **PIRQSM** (for protocol) enables or disables the interrupt requests. The request lines from the enabled sources are combined (OR) to a single level sensitive request line, which acts as input for the corresponding bit I2C_ERR_INT (for error) or I2Cm_bit I2C_P_INT (for protocol) in the Raw Interrupt Status Register **RIS**.

Inter-Integrated Circuit Module (I2C)

31.3 I2C Module Internal Registers

This section describes the internal registers of the I2C module. All register names described in this section are also referenced in other parts of the TC26x B-Step User's Manual by the module name prefix "I2Cm_". For an overview of all module registers, see [Section 31.4.4](#).

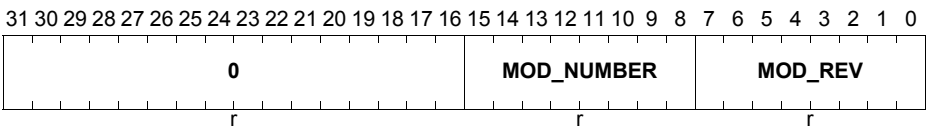
31.3.1 Global Module Control Registers

Module Identification Register

This register contains read-only information about the module and its revision.

ID

Module Identification Register (08_H) Reset Value: 0000 5703_H



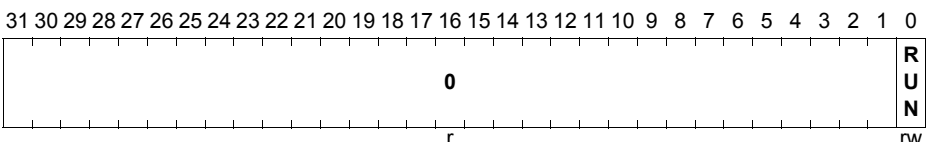
Field	Bits	Type	Description
MOD_REV	7:0	r	Module Revision Number This bit-field defines the revision number.
MOD_NUMBER	15:8	r	Module Number This bit-field defines the module identification number..
0	31:16	r	Reserved Read as 0; should be written with 0.

RUN Control Register

This register selects configuration mode or run mode.

RUNCTRL

RUN Control Register (10_H) Reset Value: 0000 0000_H



Inter-Integrated Circuit Module (I2C)

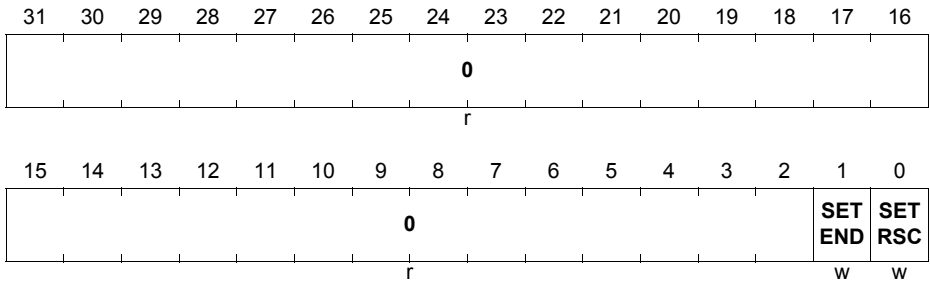
Field	Bits	Type	Description
RUN	0	rw	Enable I2C-bus Interface 0 _B I2C-bus interface disabled; write access to configuration registers enabled 1 _B Participation in I2C-bus communication enabled (if properly configured); write access to configuration registers disabled
0	31:1	r	Reserved Read as 0; should be written with 0.

End Data Control Register

This register is used to either turn around the data transmission direction or address another slave without sending a stop condition. Also the software can stop the slave-transmitter by sending a not-acknowledge when working as master-receiver or even stop data transmission immediately when operating as master-transmitter. The writing to the bits of this control register is only effective in certain states.

ENDDCTRL

End Data Control Register (14_H) **Reset Value: 0000 0000_H**



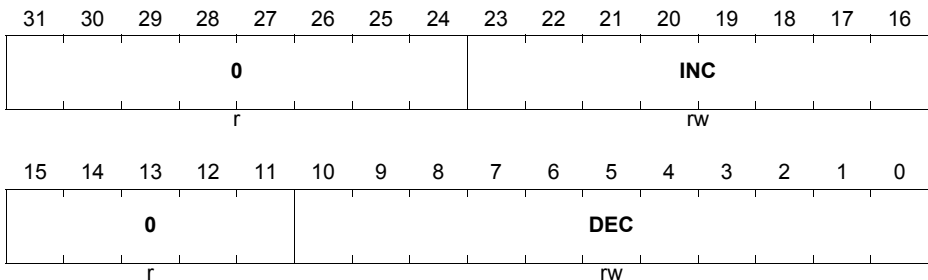
Field	Bits	Type	Description
SETRSC	0	w	<p>Set Restart Condition</p> <p>This bit is always read as 0.</p> <p>0_B Has no effect.</p> <p>1_B The master wants to restart a data transmission (changing slave/direction). The effect depends on the current state.</p> <p>MASTER RECEIVES BYTES: The master puts a not-acknowledge on the bus and switches to MASTER RESTART state.</p> <p>MASTER TRANSMITS BYTES: After the current byte has been sent, the master switches to MASTER RESTART state.</p>
SETEND	1	w	<p>Set End of Transmission</p> <p>This bit is always read as 0.</p> <p><i>Note: Do not write 1 to this bit when bus is free. This will cause an abort after the first byte when a new transfer is started.</i></p> <p>0_B Has no effect.</p> <p>1_B The effect depends on the current state.</p> <p>MASTER RECEIVES BYTES: After receiving the current byte, the master puts a not-acknowledge on the bus to indicate the transmission end to the slave-transmitter. Next it produces a stop condition on the bus and changes its state to LISTENING.</p> <p>MASTER TRANSMITS BYTES: After sending the current byte and receiving an acknowledge or not-acknowledge from the slave-receiver, the master puts a stop condition on the bus to close the data transmission and changes its state to LISTENING.</p> <p>MASTER RESTART: The master puts a stop condition on the bus to close the data transmission and changes its state to LISTENING.</p> <p>SLAVE RECEIVES BYTES: The slave-receiver puts a not-acknowledge on the bus after the received byte and changes its state to TRANSMISSION FINISHED.</p>

Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
0	31:2	r	Reserved Read as 0; should be written with 0.

Fractional Divider Configuration Register

This configuration register is used to program the fractional divider of the I2C-bus for standard and fast mode. Before the peripheral is switched on by setting the RUN bit, the register should be configured.

FDIVCFG
Fractional Divider Configuration Register
(18_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
DEC	10:0	rw	Decrement Value of Fractional Divider For standard/fast mode, see Baudrate Generation .
INC	23:16	rw	Increment Value of Fractional Divider For standard/fast mode, see Baudrate Generation .
0	15:11, 31:24	r	Reserved Read as 0; should be written with 0.

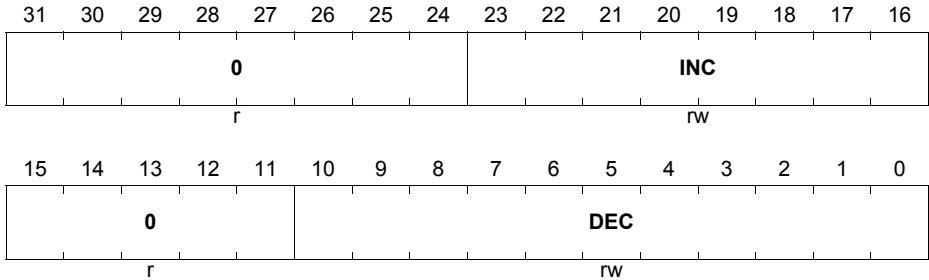
Fractional Divider High-speed Mode Configuration Register

This configuration register is used to program the fractional divider of the I2C-bus for high-speed mode. Before the peripheral is switched on by setting the RUN bit, the register should be configured if high-speed mode is used.

Inter-Integrated Circuit Module (I2C)

FDIVHIGHCFG
Fractional Divider High-speed Mode Configuration Register

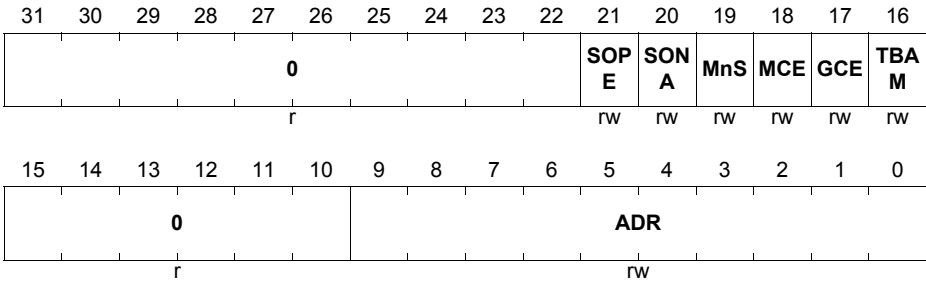
 (1C_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
DEC	10:0	rw	Decrement Value of Fractional Divider For high-speed mode, see Baudrate Generation .
INC	23:16	rw	Increment Value of Fractional Divider For high-speed mode, see Baudrate Generation .
0	31:24, 15:11	r	Reserved Read as 0; should be written with 0.

Inter-Integrated Circuit Module (I2C)
Address Configuration Register

This configuration register contains the I2C-address (when addressed as a slave) and some bits that control the basic operation of the peripheral.

ADDRCFG
Address Configuration Register
(20_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
ADR	9:0	rw	I2C-bus Device Address This bit-field determines the address of the device when addressed as a slave. (Watch out for reserved addresses by referring to I2C-bus spec V2.1.) Depending on setting of TBAM, this is either a 7-bit address (bits [7:1]) or a 10-bit address (bits [9:0]).
TBAM	16	rw	Ten Bit Address Mode <i>Note: When this bit is zero, only bits 7 down to 1 of the ADR field are valid.</i> 0 _B 7-bit address mode enabled. 1 _B 10-bit address mode enabled.
GCE	17	rw	General Call Enable 0 _B Ignore general call occurrence. 1 _B Enable general call detection; when detected, an acknowledge will be put on the bus
MCE	18	rw	Master Code Enable 0 _B Device is not able to get along with high-speed mode 1 _B Device is able to handle master code

Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
MnS	19	rw	Master / not Slave 0 _B Peripheral is configured as slave 1 _B Peripheral is configured as master
SONA	20	rw	Stop on Not-acknowledge <i>Note: After successful transmission of a master code (during high-speed mode) SONA is not considered till a stop condition is manually generated by SETEND.</i> 0 _B Device changes to MASTER RESTART state. 1 _B Device puts a stop condition on the bus and changes to LISTENING state.
SOPE	21	rw	Stop on Packet End <i>Note:</i> <ol style="list-style-type: none"> If device works as receiver a not-acknowledge is always generated on package end. After successful transmission of a master code (during high-speed mode) SOPE is not considered till a stop condition is manually generated by SETEND. 0 _B Device enters MASTER RESTART state when the data packet end is indicated by the FIFO. 1 _B Device puts a stop condition on the bus when the data packet end is indicated by the FIFO and changes to MASTER LISTENING state.
0	15:10, 31:22	r	Reserved Read as 0; should be written with 0.

Inter-Integrated Circuit Module (I2C)

Bus Status Register

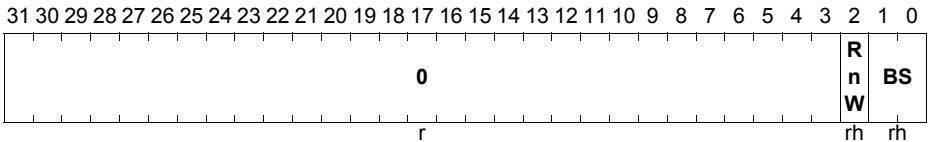
This register contains status information of the I2C-bus. This additional information can be used by software to start appropriate actions.

BUSSTAT

Bus Status Register

(24_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
BS	1:0	rh	<p>Bus Status</p> <p>Shows the current status on the I2C-bus.</p> <p>00_B I2C-bus is free (no start condition detected).</p> <p>01_B A start condition has been detected on the bus (bus busy).</p> <p>10_B The device is working as master and has claimed the control on the I2C-bus (busy master).</p> <p>11_B A remote master has accessed this device as slave.</p>
RnW	2	rh	<p>Read/not Write</p> <p>Set by hardware automatically after address byte has been sent/received.</p> <p>0_B Working as transmitter (Write to I2C-bus).</p> <p>1_B Working as receiver (Read from I2C-bus).</p>
0	31:3	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Timing Configuration Register

This configuration register adjusts some timings of the I2C-bus signals SCL and SCA. The delays are given in kernel_clk cycles (denoted as stages below).

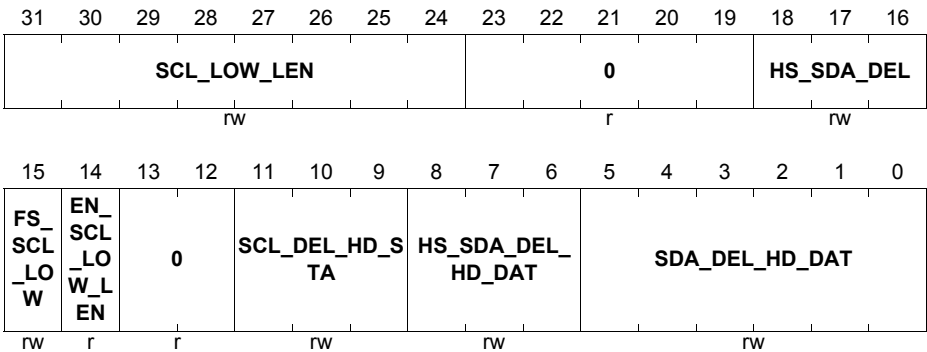
Inter-Integrated Circuit Module (I2C)

TIMCFG

Timing Configuration Register

(40_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
SDA_DEL_HD_DAT	5:0	rw	SDA Delay Stages for Data Hold Time SDA delay stages for data hold time in standard and fast mode. 000000 _B 3 stages delay 000001 _B 4 stages delay XXXXXX _B ... 111111 _B 66 stages delay
HS_SDA_DEL_HD_DAT	8:6	rw	SDA Delay Stages for Data Hold Time in High-speed Mode 000 _B 3 stages delay 001 _B 4 stages delay XXX _B ... 111 _B 10 stages delay
SCL_DEL_HD_STA	11:9	rw	SCL Delay Stages for Hold Time Start (Restart) Bit 000 _B 2 stages delay 001 _B 3 stages delay XXX _B .. 111 _B 9 stages delay

Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
EN_SCL_LOW_LEN	14	rw	Enable Direct Configuration of SCL Low Period Length in Fast Mode 0B DIS , SCL low period is a fixed part of the whole period, as defined by FS_SCL_LOW 1B EN , SCL low period is determined by the setting of SCL_LOW_LEN
FS_SCL_LOW	15	rw	Set Fast Mode SCL Low Period Timing 0 _B Standard mode SCL low period timing 1 _B Fast mode SCL low period timing
HS_SDA_DEL	18:16	rw	SDA Delay Stages for Start/Stop bit in High-speed Mode 000 _B 3 stages delay 001 _B 4 stages delay XXX _B ... 111 _B 10 stages delay
SCL_LOW_LEN	31:24	rw	SCL Low Length in Fast Mode If enabled by EN_SCL_LOW_LEN setting, this field determines the extension of the SCL low time. In case of INC = 1, the low time is extended by the number of kernel_clk cycles. In general, there is a more complex formula, as given in the functional specification. The total period time is not changed, i.e., the SCL high period is reduced accordingly. Setting SCL low time to period length or higher is not supported and would lead to unpredictable results. Reset: 00 _H
0	13:12, 23:19	r	Reserved Read as 0; should be written with 0.

The delayed stages may have +/- 1 stage deviation.

31.3.2 FIFO Registers

Transmission Data Register

The software has to write the characters to be transmitted into this register.

A larger address range (8000_H to BFFC_H) is reserved for the FIFO. Accessing any address in the defined range has the same effect as accessing the first address.

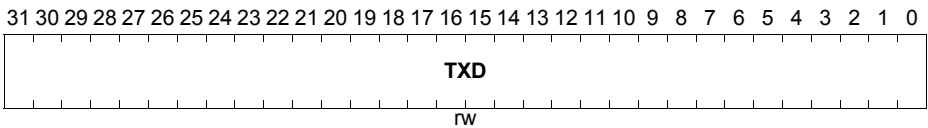
A read access to TXD register is not possible, it will return 0 in all cases. Reading has no effect on the FIFO.

When using byte or half word access from the bus, the TX FIFO pointer will only be increased, if one of the following conditions is fulfilled:

- The most significant byte or half word of the FIFO stage is written
- The packet end is reached

TXD

Transmission Data Register (8000_H) Reset Value: 0000 0000_H

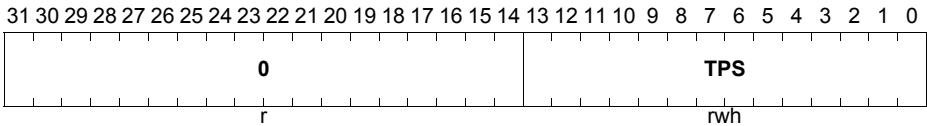


Field	Bits	Type	Description
TXD	31:0	rw	Transmission Data Characters to be transmitted

Inter-Integrated Circuit Module (I2C)
Transmit Packet Size Control Register

This register is used to indicate the peripheral the size of the packet to be transmitted. Writing the packet size to this register if the FIFO controller is configured for flow controller mode initiates the data requests (BREQ, SREQ, ...).

Writing to this register in configuration state has no impact.

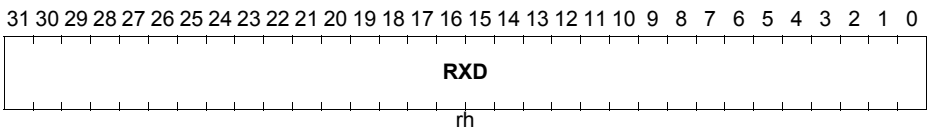
TPSCTRL
Transmit Packet Size Control Register(34_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
TPS	13:0	rwh	Transmit Packet Size Length in characters of the transmit packet, write value range: 1 to 16383 Reading returns the written value as long as it is not loaded to an internal counter. After that, reading returns 0 and a new value can be written.
0	31:14	r	Reserved Read as 0; should be written with 0.

Reception Data Register

The software can read the received characters from this register.

A larger address range (C000_H to FFFC_H) is reserved for the FIFO. Reading from any address in the defined range has the same effect as reading from the first address.

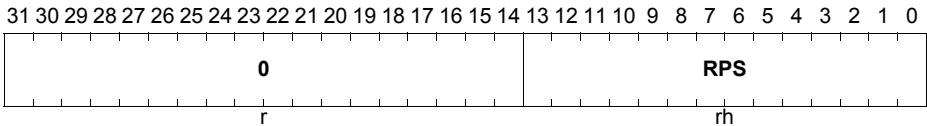
RXD
Reception Data Register
(C000_H)
Reset Value: 0000 0000_H


Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
RXD	31:0	rh	Reception Data Received characters

Received Packet Size Status Register

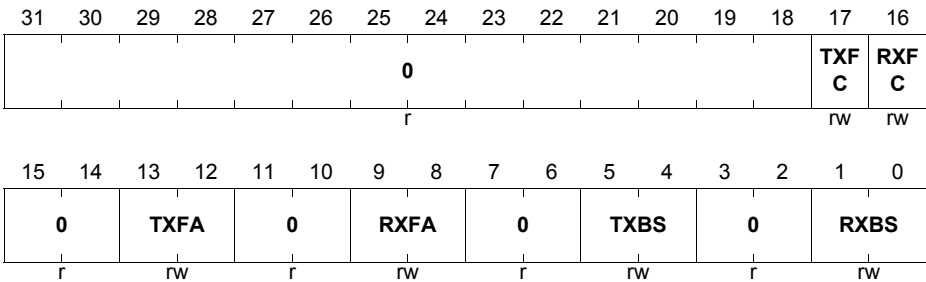
This register indicates the size of the received data packet to the software. The software should read this register after the last request of a packet.

RPSSTAT
Received Packet Size Status Register
(30_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RPS	13:0	rh	Received Packet Size Length in characters of the received packet (0 to 16383)
0	31:14	r	Reserved Read as 0; should be written with 0.

FIFO Configuration Register

This configuration register is used to set up the FIFO before the peripheral is enabled and data is received or transmitted.

Inter-Integrated Circuit Module (I2C)
FIFO CFG
FIFO Configuration Register
(28_H)
Reset Value: 0000 0022_H


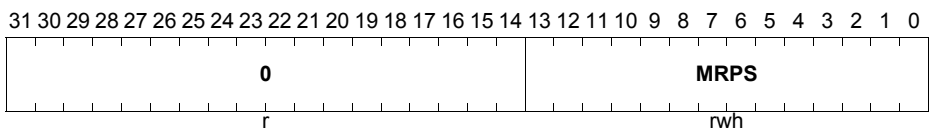
Field	Bits	Type	Description
RXBS	1:0	rw	RX Burst Size 00 _B 1 word 01 _B 2 words 10 _B 4 words 11 _B Do not use this combination
TXBS	5:4	rw	TX Burst Size 00 _B 1 word 01 _B 2 words 10 _B 4 words 11 _B Do not use this combination
RXFA	9:8	rw	RX FIFO Alignment Use byte alignment wherever it is possible. 00 _B Byte aligned (character alignment) 01 _B Half word aligned (character alignment of two characters) 10 _B Word aligned (character alignment of four characters) 11 _B Do not use this combination

Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
TXFA	13:12	rw	TX FIFO Alignment Use byte alignment wherever it is possible. 00 _B Byte aligned (character alignment) 01 _B Half word aligned (character alignment of two characters) 10 _B Word aligned (character alignment of four characters) 11 _B Do not use this combination
RXFC	16	rw	RX FIFO Flow Control 0 _B RX FIFO not as flow controller 1 _B RX FIFO as flow controller
TXFC	17	rw	TX FIFO Flow Control 0 _B TX FIFO not as flow controller 1 _B TX FIFO as flow controller
0	3:2, 7:6, 11:10, 15:14, 31:18	r	Reserved Read as 0; should be written with 0.

Maximum Received Packet Size Control Register

This register is used to limit the received packet size. The register value may be changed in any state of the FIFO.

MRPSCTRL
Maximum Received Packet Size Control Register
(2C_H)
Reset Value: 0000 0000_H


Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
MRPS	13:0	rwh	Maximum Received Packet Size Length in characters of packet to be received; write value range: 0 (unlimited size) to 16383 Reading returns the written value as long as the previous packet has not been read completely from the FIFO. After that, MRPS is loaded to an internal register, reading returns 0 and a new value can be written.
0	31:14	r	Reserved Read as 0; should be written with 0.

Filled FIFO Stages Status Register

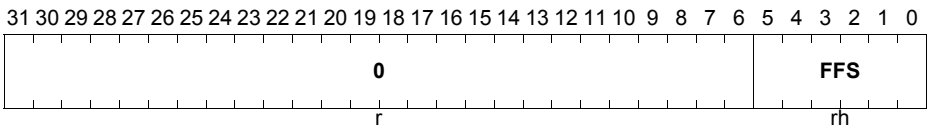
This register is used to indicate the number of filled FIFO stages.

FFSSTAT

Filled FIFO Stages Status Register

(38_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
FFS	5:0	rh	Filled FIFO Stages Number of filled FIFO stages (0 to 8)
0	31:6	r	Reserved Read as 0; should be written with 0.

31.3.3 Basic Interrupt Registers

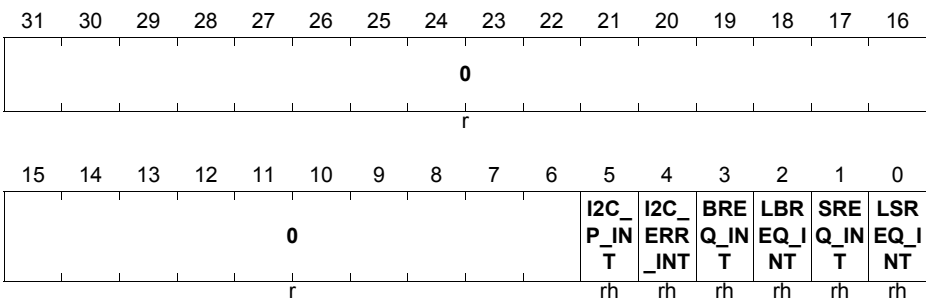
For an overview of the Service Request Block (SRB) see [Section 31.2.5](#).

Raw Interrupt Status Register

This read-only register returns the current raw status value (without reflecting the mask) of the interrupt request sources. One status bit is provided for each request. A write to this register has no effect. The status bits are set by hardware or software (via register [ISR](#)) and can be cleared by software (via register [ICR](#)).

RIS

Raw Interrupt Status Register (80_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
LSREQ_INT	0	rh	Last Single Request Interrupt 0 _B No interrupt request 1 _B Interrupt request pending
SREQ_INT	1	rh	Single Request Interrupt 0 _B No interrupt request 1 _B Interrupt request pending
LBREQ_INT	2	rh	Last Burst Request Interrupt 0 _B No interrupt request 1 _B Interrupt request pending
BREQ_INT	3	rh	Burst Request Interrupt 0 _B No interrupt request 1 _B Interrupt request pending

Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
I2C_ERR_INT	4	rh	I2C Error Interrupt This is the combined bit for indication of FIFO errors due to overflow and underflow. 0 _B No interrupt request 1 _B Interrupt request pending
I2C_P_INT	5	rh	I2C Protocol Interrupt This is the combined bit for indication of a protocol event in the I2C kernel. 0 _B No interrupt request 1 _B Interrupt request pending
0	31:6	r	Reserved Read as 0; should be written with 0.

Interrupt Mask Control Register

A write of 1 to a particular bit of this register enables the corresponding interrupt request; a write of 0 disables it. A read to this register returns the current mask bits. After reset all requests are disabled.

IMSC
Interrupt Mask Control Register (84_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										I2C_	I2C_	BRE	LBR	SRE	LSR
										P_IN	ERR	Q_IN	EQ_I	Q_IN	EQ_I
										T	_INT	T	NT	T	NT
r										rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
LSREQ_INT	0	rw	Last Single Request Interrupt 0 _B Interrupt request disabled 1 _B Interrupt request enabled

Inter-Integrated Circuit Module (I2C)

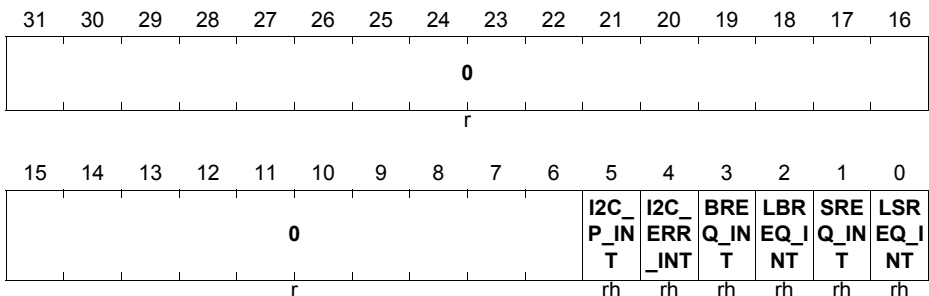
Field	Bits	Type	Description
SREQ_INT	1	rw	Single Request Interrupt 0 _B Interrupt request disabled 1 _B Interrupt request enabled
LBREQ_INT	2	rw	Last Burst Request Interrupt 0 _B Interrupt request disabled 1 _B Interrupt request enabled
BREQ_INT	3	rw	Burst Request Interrupt 0 _B Interrupt request disabled 1 _B Interrupt request enabled
I2C_ERR_INT	4	rw	I2C Error Interrupt 0 _B Interrupt request disabled 1 _B Interrupt request enabled
I2C_P_INT	5	rw	I2C Protocol Interrupt 0 _B Interrupt request disabled 1 _B Interrupt request enabled
0	31:6	r	Reserved Read as 0; should be written with 0.

Masked Interrupt Status Register

This read-only register returns the masked status value (derived from registers **RIS** and **IMSC**) of the corresponding interrupt requests. A write to this register has no effect.

MIS

Masked Interrupt Status Register (88_H) **Reset Value: 0000 0000_H**

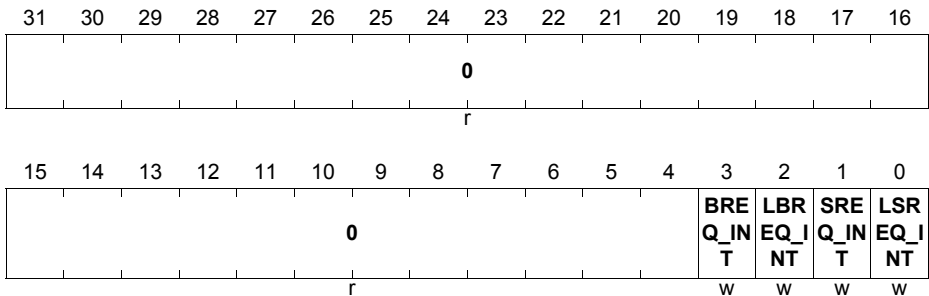


Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
LSREQ_INT	0	rh	Last Single Request Interrupt 0_B No interrupt request 1_B Interrupt request pending
SREQ_INT	1	rh	Single Request Interrupt 0_B No interrupt request 1_B Interrupt request pending
LBREQ_INT	2	rh	Last Burst Request Interrupt 0_B No interrupt request 1_B Interrupt request pending
BREQ_INT	3	rh	Burst Request Interrupt 0_B No interrupt request 1_B Interrupt request pending
I2C_ERR_INT	4	rh	I2C Error Interrupt This is the combined bit for indication of FIFO errors due to overflow and underflow. 0_B No interrupt request 1_B Interrupt request pending
I2C_P_INT	5	rh	I2C Protocol Interrupt This is the combined bit for indication of a protocol event in the I2C kernel. 0_B No interrupt request 1_B Interrupt request pending
0	31:6	r	Reserved Read as 0; should be written with 0.

Interrupt Clear Register

On a write of 1 to a particular bit of this write-only register, the corresponding interrupt request is cleared; a write of 0 has no effect. Reading the register returns 0.

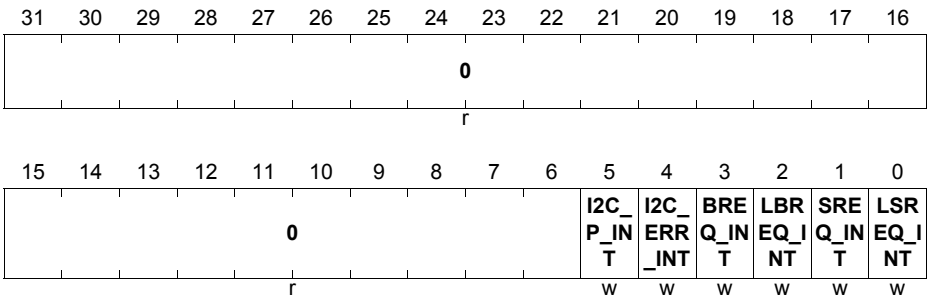
Inter-Integrated Circuit Module (I2C)
ICR
Interrupt Clear Register
(8C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
LSREQ_INT	0	w	Last Single Request Interrupt 0 _B No change 1 _B Clear interrupt request
SREQ_INT	1	w	Single Request Interrupt 0 _B No change 1 _B Clear interrupt request
LBREQ_INT	2	w	Last Burst Request Interrupt 0 _B No change 1 _B Clear interrupt request
BREQ_INT	3	w	Burst Request Interrupt 0 _B No change 1 _B Clear interrupt request
0	31:4	r	Reserved Read as 0; should be written with 0.

Interrupt Set Register

On a write of 1 to a particular bit of this write-only register, the corresponding interrupt request is set; a write of 0 has no effect. Reading the register returns 0.

Inter-Integrated Circuit Module (I2C)

ISR
Interrupt Set Register
(90_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
LSREQ_INT	0	w	Last Single Request Interrupt 0 _B No change 1 _B Set interrupt request
SREQ_INT	1	w	Single Request Interrupt 0 _B No change 1 _B Set interrupt request
LBREQ_INT	2	w	Last Burst Request Interrupt 0 _B No change 1 _B Set interrupt request
BREQ_INT	3	w	Burst Request Interrupt 0 _B No change 1 _B Set interrupt request
I2C_ERR_INT	4	w	I2C Error Interrupt 0 _B No change 1 _B Set interrupt request
I2C_P_INT	5	w	I2C Protocol Interrupt 0 _B No change 1 _B Set interrupt request
0	31:6	r	Reserved Read as 0; should be written with 0.

31.3.4 Error Interrupt Source Registers

For an overview of the source register operation see [Section 31.2.5.2](#).

Error Interrupt Request Source Mask Register

A write of 1 to a particular bit of this register enables the corresponding error interrupt request source; a write of 0 disables it. A read to this register returns the current mask bits. After reset all sources are enabled.

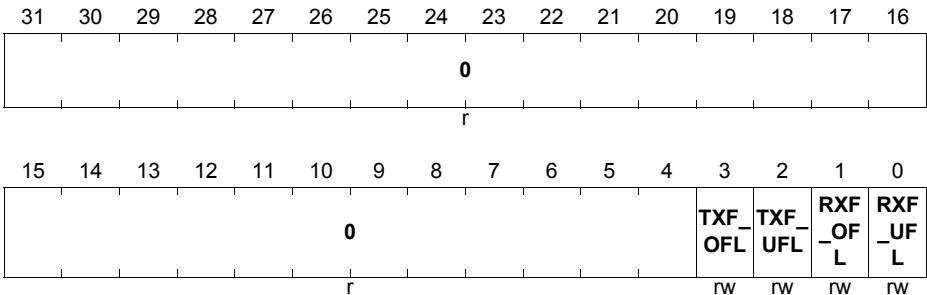
The interrupts are explained in detail in description of register [ERRIRQSS](#).

ERRIRQSM

Error Interrupt Request Source Mask Register

(60_H)

Reset Value: 0000 000F_H



Field	Bits	Type	Description
RXF_UFL	0	rw	RX FIFO Underflow 0 _B Interrupt request source disabled 1 _B Interrupt request source enabled
RXF_OFL	1	rw	RX FIFO Overflow 0 _B Interrupt request source disabled 1 _B Interrupt request source enabled
TXF_UFL	2	rw	TX FIFO Underflow 0 _B Interrupt request source disabled 1 _B Interrupt request source enabled
TXF_OFL	3	rw	TX FIFO Overflow 0 _B Interrupt request source disabled 1 _B Interrupt request source enabled
0	31:4	r	Reserved Read as 0; should be written with 0.

Inter-Integrated Circuit Module (I2C)

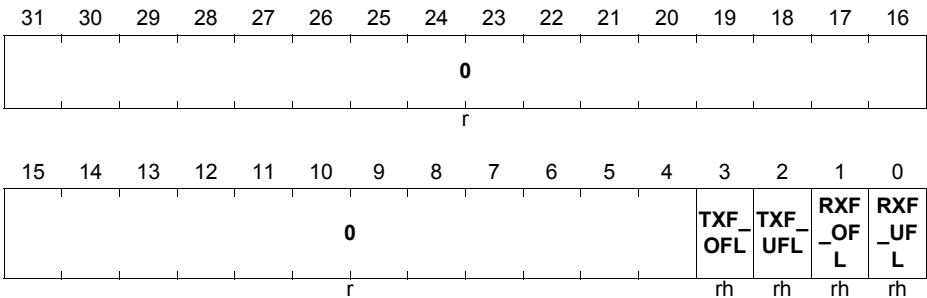
Error Interrupt Request Source Status Register

This read-only register returns the current raw status value (without reflecting the mask) of the error interrupt request sources. A write to this register has no effect. The error status bits are set by hardware and can be cleared by software (via register **ERRIRQSC**).

ERRIRQSS

Error Interrupt Request Source Status Register (64_H)

Reset Value: 0000 0000_H

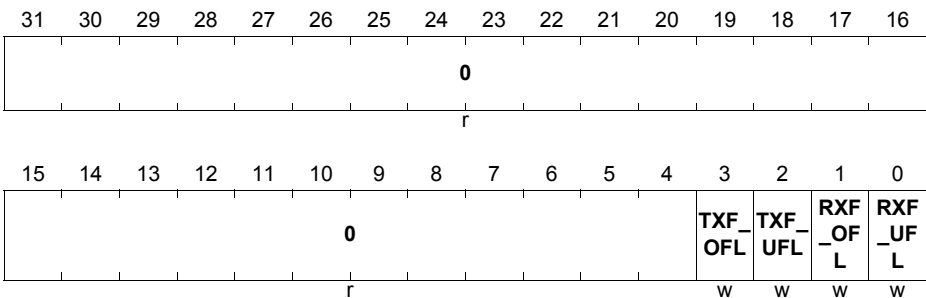


Field	Bits	Type	Description
RXF_UFL	0	rh	RX FIFO Underflow 0 _B No interrupt request 1 _B Interrupt request pending
RXF_OFL	1	rh	RX FIFO Overflow 0 _B No interrupt request 1 _B Interrupt request pending
TXF_UFL	2	rh	TX FIFO Underflow The I2C kernel has detected a TX FIFO underflow. 0 _B No interrupt request 1 _B Interrupt request pending
TXF_OFL	3	rh	TX FIFO Overflow 0 _B No interrupt request 1 _B Interrupt request pending
0	31:4	r	Reserved Read as 0; should be written with 0.

Inter-Integrated Circuit Module (I2C)
Error Interrupt Request Source Clear Register

On a write of 1 to a particular bit of this write-only register, the corresponding error interrupt request source is cleared and if no further error interrupt request sources are active, the whole error interrupt is cleared. If the corresponding bit is set by SW via the **ISR** register, then it can be cleared by setting any non-reserved bit of the interrupt request source clear register. A write of 0 has no effect. Reading the register returns 0.

The interrupts are explained in detail in description of register **ERRIRQSS**.

ERRIRQSC
Error Interrupt Request Source Clear Register
(68_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RXF_UFL	0	w	RX FIFO Underflow 0 _B No change 1 _B Clear interrupt request source
RXF_OFL	1	w	RX FIFO Overflow 0 _B No change 1 _B Clear interrupt request source
TXF_UFL	2	w	TX FIFO Underflow 0 _B No change 1 _B Clear interrupt request source
TXF_OFL	3	w	TX FIFO Overflow 0 _B No change 1 _B Clear interrupt request source
0	31:4	r	Reserved Read as 0; should be written with 0.

31.3.5 Protocol Interrupt Source Registers

For an overview of the source register operation see [Section 31.2.5.2](#).

Protocol Interrupt Request Source Mask Register

A write of 1 to a particular bit of this register enables the corresponding protocol interrupt request source; a write of 0 disables it. A read to this register returns the current mask bits. After reset all sources are enabled.

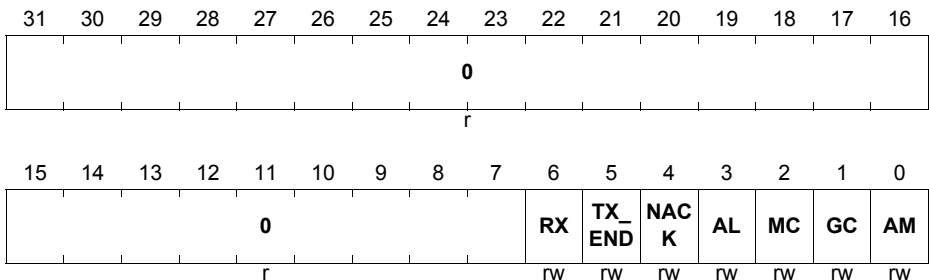
The interrupts are explained in detail in description of register [PIRQSS](#).

PIRQSM

Protocol Interrupt Request Source Mask Register

(70_H)

Reset Value: 0000 007F_H



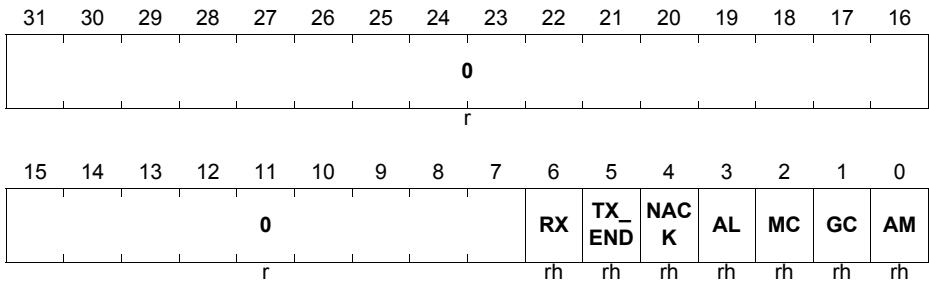
Field	Bits	Type	Description
AM	0	rw	Address Match 0 _B Interrupt request source disabled 1 _B Interrupt request source enabled
GC	1	rw	General Call 0 _B Interrupt request source disabled 1 _B Interrupt request source enabled
MC	2	rw	Master Code 0 _B Interrupt request source disabled 1 _B Interrupt request source enabled
AL	3	rw	Arbitration Lost 0 _B Interrupt request source disabled 1 _B Interrupt request source enabled

Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
NACK	4	rw	Not-acknowledge Received 0 _B Interrupt request source disabled 1 _B Interrupt request source enabled
TX_END	5	rw	Transmission End 0 _B Interrupt request source disabled 1 _B Interrupt request source enabled
RX	6	rw	Receive Mode 0 _B Interrupt request source disabled 1 _B Interrupt request source enabled
0	31:7	r	Reserved Read as 0; should be written with 0.

Protocol Interrupt Request Source Status Register

This read-only register returns the current raw status value (without reflecting the mask) of the protocol interrupt request sources. A write to this register has no effect. The protocol interrupt status bits are set by hardware and can be cleared by software (via register **PIRQSC**).

PIRQSS
Protocol Interrupt Request Source Status Register
(74_H)
Reset Value: 0000 0000_H


Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
AM	0	rh	Address Match Device (in master or slave mode) is addressed by remote master (matching device address). Accordingly, bit-field BS in register BUSSTAT is set to 11 _B . 0 _B No interrupt request 1 _B Interrupt request pending
GC	1	rh	General Call Remote master has transmitted a general call. 0 _B No interrupt request 1 _B Interrupt request pending
MC	2	rh	Master Code Remote master has transmitted a master call. 0 _B No interrupt request 1 _B Interrupt request pending
AL	3	rh	Arbitration Lost Device (master mode) lost the control on the I2C-bus due to losing arbitration procedure. Accordingly, bit-field BS in register BUSSTAT is set to 01 _B . 0 _B No interrupt request 1 _B Interrupt request pending
NACK	4	rh	Not-acknowledge Received When working as transmitter this interrupt indicates a not-acknowledge from the remote receiver. The SW has to decide what further steps have to be taken. 0 _B No interrupt request 1 _B Interrupt request pending
TX_END	5	rh	Transmission End The device has ended the data transfer properly (after stop condition has been put on the bus or the MASTER RESTART state has been entered.) 0 _B No interrupt request 1 _B Interrupt request pending

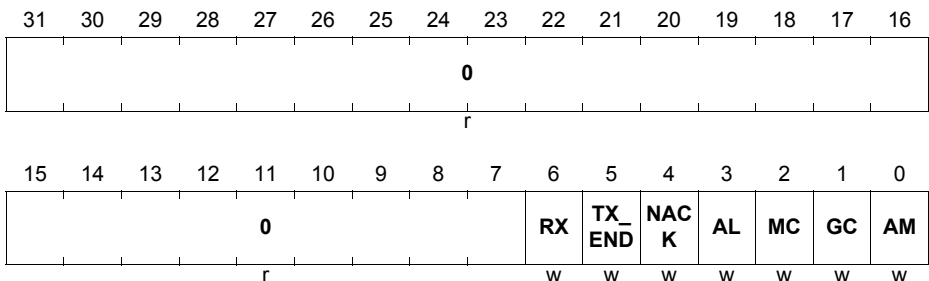
Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
RX	6	rh	Receive Mode I2C kernel indicates switching from transmitting data to receiving data. 0 _B No interrupt request 1 _B Interrupt request pending
0	31:7	r	Reserved Read as 0; should be written with 0.

Protocol Interrupt Request Source Clear Register

On a write of 1 to a particular bit of this write-only register, the corresponding protocol interrupt request source is cleared and if no further protocol interrupt request sources are active, the whole protocol interrupt is cleared. If the corresponding **RIS** bit is set by SW via the **ISR** register, then it can be cleared by setting any non-reserved bit of the interrupt request source clear register. A write of 0 has no effect. Reading the register returns 0.

The interrupts are explained in detail in description of register **PIRQSS**.

PIRQSC
Protocol Interrupt Request Source Clear Register
(78_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
AM	0	w	Address Match 0 _B No change 1 _B Clear Interrupt source
GC	1	w	General Call 0 _B No change 1 _B Clear Interrupt source

Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
MC	2	w	Master Code 0 _B No change 1 _B Clear Interrupt source
AL	3	w	Arbitration Lost 0 _B No change 1 _B Clear Interrupt source
NACK	4	w	Not-acknowledge Received 0 _B No change 1 _B Clear Interrupt source
TX_END	5	w	Transmission End 0 _B No change 1 _B Clear Interrupt source
RX	6	w	Receive Mode 0 _B No change 1 _B Clear Interrupt source
0	31:7	r	Reserved Read as 0; should be written with 0.

Inter-Integrated Circuit Module (I2C)

31.4 I2C Module Implementation

This section describes I2C module interfaces with the clock control, port control and interrupt control. TC26x B-Step provides one module I2C0.

Note: This module may not be available in all marking options. If it is not available then the functionality behind the associated registers is not guaranteed.

31.4.1 Interfaces of the I2C Module(s)

Figure 31-25 shows the TC26x B-Step specific implementation details and interconnections of the I2C module(s):

- A clock control block generates the clock signals required for the module.
- For the I2C-bus lines SCL and SDA, different I/O lines can be selected.
- The interrupt outputs of the module are connected to the interrupt control unit.

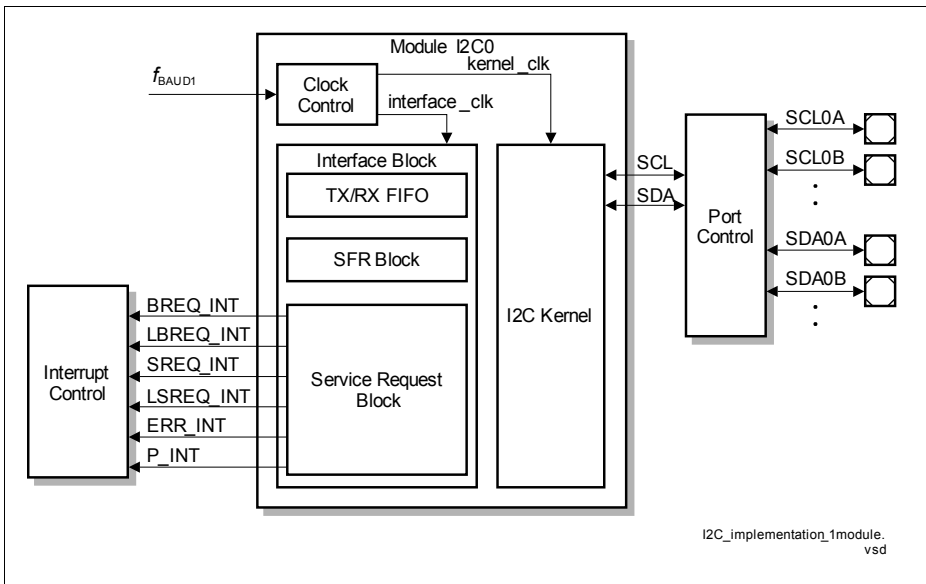


Figure 31-25 I2C Module Implementation and Interconnections

31.4.2 Module Clock Control

The clock control allows the programmer to adapt the peripheral's functionality and power consumption to the application's requirement. The clock f_{BAUD1} is connected to the bus clock, which generates the clock for the interface block, and to the module clock, from which the I2C kernel clock is derived. By programming the kernel's operating frequency, an optimal ratio between power consumption, EMC behavior and functionality can be achieved. Furthermore, for power saving reasons the peripheral can be disabled as a whole by switching off the peripheral's clocks via clock gating cells.

A simplified description of the clock control is shown in [Figure 31-26](#). See register [CLC1](#) for a description of the clock control parameters.

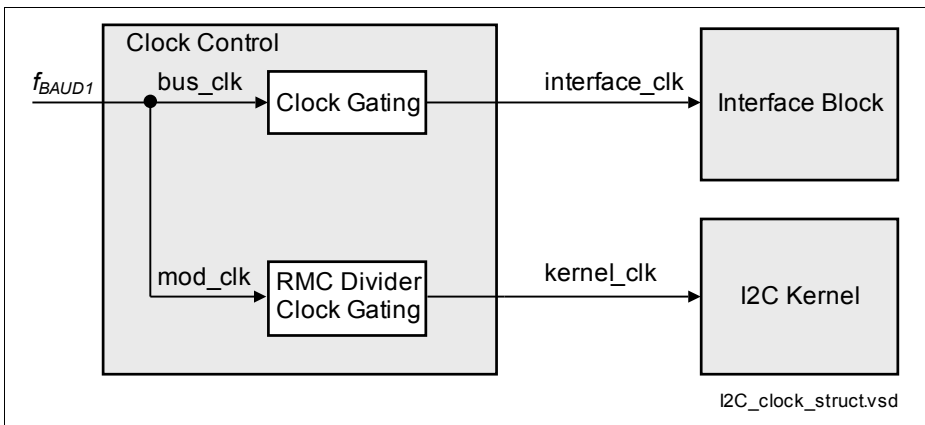


Figure 31-26 Module Clock Control

The following clocking modes are supported for the peripheral:

- Standard run mode
- Disable mode
- OCDS suspend mode

Standard Run Mode

In this mode the module clock is divided by the value of bit-field RMC in register [CLC1](#). The value written to this bit field is equal to the dividing factor (e.g. RMC=04_H is equal to dividing by factor 4). If 00_H is written into RMC, the kernel clock is stopped.

Inter-Integrated Circuit Module (I2C)

Kernel Clock Calculation in Standard Run Mode:

$$f_{kernel_clk} = \frac{f_{mod_clk}}{RMC}; (RMC > 0) \quad (31.1)$$

I2C_FORMULA_RMC

Disable Mode

In this mode the clocks are switched off. It can be entered by either activating the module disable input (if not disabled by setting bit EDIS in register **CLC1**) or by setting bit DISR in register **CLC1**. Note that a secure clock shut off with handshake is performed before it is disabled (see “Secure Clock Shut Off” in **OCDS Suspend Mode**).

Additionally a peripheral can be disabled without handshake (see “Fast Clock Shut Off” in **OCDS Suspend Mode**) by setting the bit field of the clock divider to 00_H.

OCDS Suspend Mode

In this mode the clocks are disabled. If bit SPEN in register **CLC1** is set, the mode can be entered by activating the input module_ocds.

In OCDS suspend mode all registers of the suspended module can be read. Note that each read during disabled clock can not affect hardware. For example multiple reads to FIFO port registers always return the same value as the FIFO control is not working.

No write access to the registers is supported during suspend mode, except writes to the **CLC1** register. For writing other registers the suspend mode for the module has to be removed first by clearing bit SPEN. After the write access is executed, bit SPEN should be set again.

OCDS suspend mode supports two different disable features, selectable by bit FSOE in register **CLC1**:

- **Fast Clock Shut Off**

Selecting the fast clock shut off stops the clock immediately after all pending read and write accesses are finished (including synchronization mechanisms with the I2C kernel). This shut off mode can be activated by setting bit DISR in register **CLC1**.

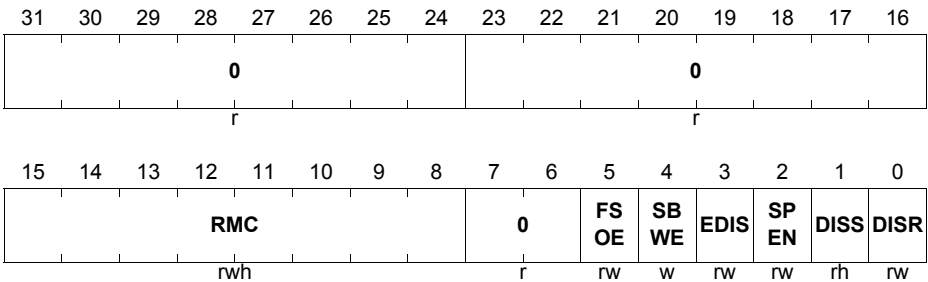
- **Secure Clock Shut Off**

Disabling the clock via secure clock shut off additionally activates the handshake mechanism with the peripheral. The peripheral is switched to a secure state before disabling the clock.

*Note: To write to SPEN or FSOE, bit SBWE in register **CLC1** must also be set in the same write access.*

31.4.3 Bus Peripheral Interface Registers

This register controls the clock gating and dividing circuitry of the peripheral.

CLC1
Clock Control 1 Register
(00_H)
Reset Value: 0000 0003_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module. 0 _B Module disable not requested 1 _B Module disable requested
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module. 0 _B Module enabled 1 _B Module disabled
SPEN	2	rw	Module Suspend Enable Bit for OCDS 0 _B Module suspend disabled 1 _B Module suspend enabled
EDIS	3	rw	External Request Disable 0 _B External clock disable request is enabled 1 _B External clock disable request is disabled
SBWE	4	w	Module Suspend Bit Write Enable for OCDS This bit is always read as 0. 0 _B Bits SPEN and FSOE are write protected 1 _B Bits SPEN and FSOE are overwritten by respective value of SPEN or FSOE

Inter-Integrated Circuit Module (I2C)

Field	Bits	Type	Description
FSOE	5	rw	Fast Switch Off Enable 0 _B FSOE Clock switch off in OCDS suspend mode via Disable Control Feature (Secure Clock Switch Off) 1 _B Fast clock switch off in OCDS suspend mode
RMC	15:8	rwh	Clock Divider for Standard Run Mode Max. 8-bit divider value If RMC is set to 0 the module is disabled. <i>Note: As long as the new divider value RMC is not valid, reading register returns 0000 00XX_H</i>
0	7:6, 31:16	r	Reserved Read as 0; should be written with 0.

31.4.4 I2C Module Registers Overview

In the following, the registers of the I2C module are listed. First of all, some explanation on the access conditions is given.

Special I2C Register Access Condition

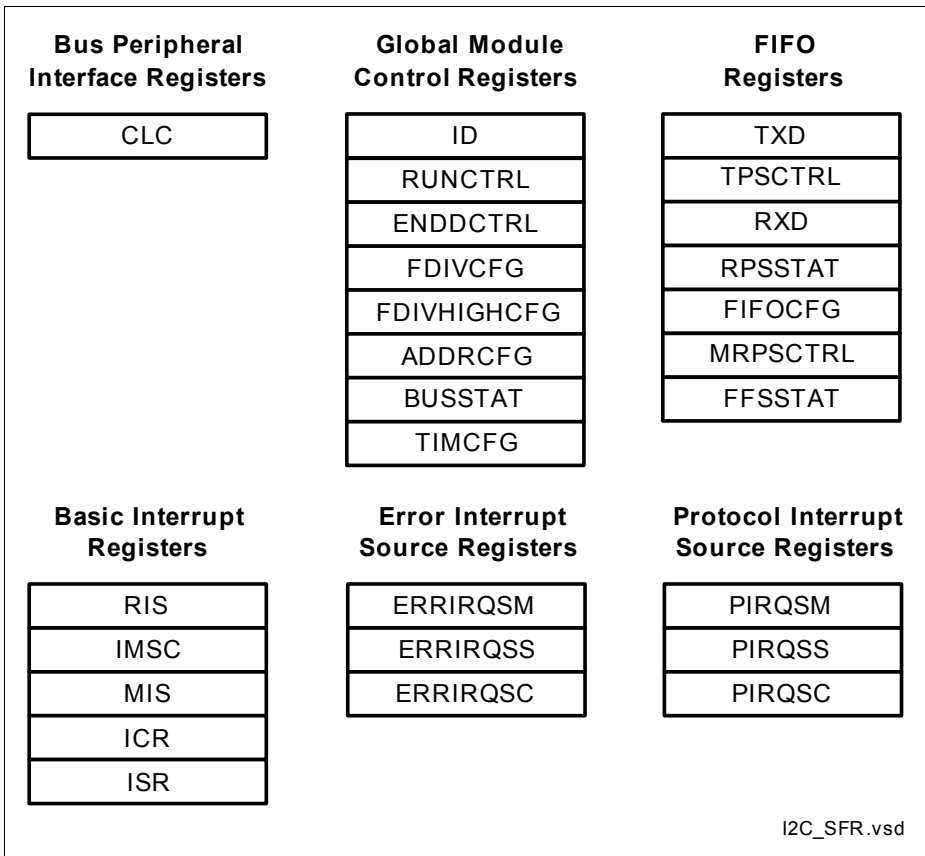
Besides the general register protection, the I2C module has two main modes that must be considered when programming the peripheral:

- **Configuration Mode:** In this mode the peripheral can be prepared for transmission and reception via the configuration registers, which are only writable in this mode. The peripheral is in the configuration mode when bit RUN is set to 0.
- **Run Mode:** In this mode the peripheral is ready to transmit or receive data. Its configuration registers are locked for write access which will generate a bus error. The peripheral is in the run mode when bit RUN is set to 1.

I2C Registers Overview

There are the following blocks of registers (see [Figure 31-27](#)):

- Bus Peripheral Interface Registers
- Global Module Control Registers
- FIFO Registers
- Basic Interrupt Registers
- Error Interrupt Source Registers
- Protocol Interrupt Source Registers


Figure 31-27 I2C Module Registers

The registers address space of the modules is defined in [Table 31-3](#).

The registers overview in [Table 31-4](#) shows the register names of the module instances, the offset addresses and the links to the names used in this specification.

An access to an address in the address range of this module which is not given in the table causes a bus error.

Table 31-3 Registers Address Space of I2C Module

Module	Base Address	End Address	Note
I2C0	F00C 0000 _H	F00D 00FF _H	~64 KByte

Inter-Integrated Circuit Module (I2C)
Table 31-4 Registers Overview of I2C Module

Short Name	Description	Offset Address	Access Mode		Page Number
			Read	Write	
Global Module Control Registers					
CLC1	Clock Control Register	00 _H	U, SV	U, SV, P	31-84
ID	Module Identification Register	08 _H	U, SV	U, SV, P	31-51
RUNCTRL	RUN Control Register	10 _H	U, SV	U, SV, P	31-51
ENDDCtrl	End Data Control Register	14 _H	U, SV	U, SV, P	31-52
FDIVCFG	Fractional Divider Configuration Register	18 _H	U, SV	U, SV, P	31-54
FDIVHIGHCFG	Fractional Divider High-speed Mode Configuration Register	1C _H	U, SV	U, SV, P	31-55
ADDRCFG	Address Configuration Register	20 _H	U, SV	U, SV, P	31-56
BUSSTAT	Bus Status Register	24 _H	U, SV	BE	31-58
TIMCFG	Timing Configuration Register	40 _H	U, SV	U, SV, P	31-59
	(Reserved)	44 _H	U, SV	U, SV, P	
FIFO Registers					
TXD	Transmit Data Register	8000 _H	U, SV	U, SV, P	31-61¹⁾
TPSCTRL	Transmit Packet Size Control Register	34 _H	U, SV	U, SV, P	31-62
RXD	Reception Data Register	C000 _H	U, SV	BE	31-62²⁾
RPSSTAT	Received Packet Size Status Register	30 _H	U, SV	BE	31-63
FIFOCFG	FIFO Configuration Register	28 _H	U, SV	U, SV, P	31-64
MRPSCTRL	Maximum Received Packet Size Control Register	2C _H	U, SV	U, SV, P	31-65

Inter-Integrated Circuit Module (I2C)
Table 31-4 Registers Overview of I2C Module (cont'd)

Short Name	Description	Offset Address	Access Mode		Page Number
			Read	Write	
FFSSTAT	Filled FIFO Stages Status Register	38 _H	U, SV	BE	31-66
Basic Interrupt Registers					
RIS	Raw Interrupt Status Register	80 _H	U, SV	BE	31-67
IMSC	Interrupt Mask Control Register	84 _H	U, SV	U, SV, P	31-68
MIS	Masked Interrupt Status Register	88 _H	U, SV	BE	31-69
ICR	Interrupt Clear Register	8C _H	U, SV	U, SV, P	31-71
ISR	Interrupt Set Register	90 _H	U, SV	U, SV, P	31-72
	(Reserved)	94 _H	U, SV	U, SV, P	
Error Interrupt Source Registers					
ERRIRQSM	Error Interrupt Request Source Mask Register	60 _H	U, SV	U, SV, P	31-73
ERRIRQSS	Error Interrupt Request Source Status Register	64 _H	U, SV	BE	31-74
ERRIRQSC	Error Interrupt Request Source Clear Register	68 _H	U, SV	U, SV, P	31-75
Protocol Interrupt Source Registers					
PIRQSM	Protocol Interrupt Request Source Mask Register	70 _H	U, SV	U, SV, P	31-76
PIRQSS	Protocol Interrupt Request Source Status Register	74 _H	U, SV	BE	31-77
PIRQSC	Protocol Interrupt Request Source Clear Register	78 _H	U, SV	U, SV, P	31-79

1) The allowed offset address range for the TXD register is 8000_H - BBFC_H.

2) The allowed offset address range for the RXD register is C000_H - FFFC_H.

Table 31-5 Registers Overview of the BPI

Short Name	Description	Offset Addr.	Access Mode		Reset Value
			Read	Write	
I2C0_CLC	Clock Control Register	10000 _H	U, SV	SV, E, P	0000 0003 _H
I2C0_MODID	Module ID	10004 _H	SV	BE	00C2 C0XX _H
I2C0_GPCTL	General Purpose Control Register	10008 _H	SV	SV, P	0000 0000 _H
I2C0_ACCEN0	Access Enable Register 0	1000C _H	SV	SV,SE	0000 0000 _H
I2C0_ACCEN1	Access Enable Register 1	10010 _H	SV	SV,SE	0000 0000 _H
I2C0_KRST0	Kernel Reset Register 0	10014 _H	SV	SV, E, P	0000 0000 _H
I2C0_KRST1	Kernel Reset Register 1	10018 _H	SV	SV, E, P	0000 0000 _H
I2C0_KRSTCLR	Kernel Reset Status Clear Register	1001C _H	SV	SV, E, P	0000 0000 _H

Note: The base address of the I2C0 BPI registers is F00D 0000_H.

Note: All I2C registers are Reset Class 3 registers.

31.4.5 Port and I/O Line Control

Not only the interconnections between the I2C module and the port I/O lines have to be configured, but also the function and the characteristics of the port pins. The following control operations must be executed for the used port pins:

- Selection of I2C module via output port multiplexer
- Configuration of output port function with open drain
- Configuration of pad driver characteristics
- Selection of I2C input lines

31.4.6 Interrupt Control

The interrupt functionality is described in the chapter for the interrupt router (IR).

The signal names used in the interrupt router description are explained in [Table 31-6](#).

Table 31-6 Interrupt Router Signal Names for I2Cm Module (m = 0/1)

Signal Name	Name in Module	Explanation
SRC_I2CmBREQ	BREQ_INT	Burst Data Transfer
SRC_I2CmLBREQ	LBREQ_INT	Last Burst Data Transfer
SRC_I2CmSREQ	SREQ_INT	Single Data Transfer
SRC_I2CmLSREQ	LSREQ_INT	Last Single Data Transfer
SRC_I2CmERR	ERR_INT	Error
SRC_I2CmP	P_INT	Protocol

31.5 Module Integration

This section describes the topics regarding the integration of the module on chip.

31.5.1 Integration Overview

The I2C module consists of a delivery providing an AHB bus interface integrated into the FPI bus system by using an FPI2AHB adapter.

The adapter contains a slave bus interface providing translation between the FPI and the AHB bus protocols. The module supports additionally the following features:

- power saving (sleep mode)
- debug suspend
- local module reset
- PISEL - port input selection regarding the serial data input

The power saving and debug suspend features are supported by the AHB based I2C module itself, and the local module reset and PISEL features are supported by the FPI2AHB adapter, see [Figure 31-28](#).

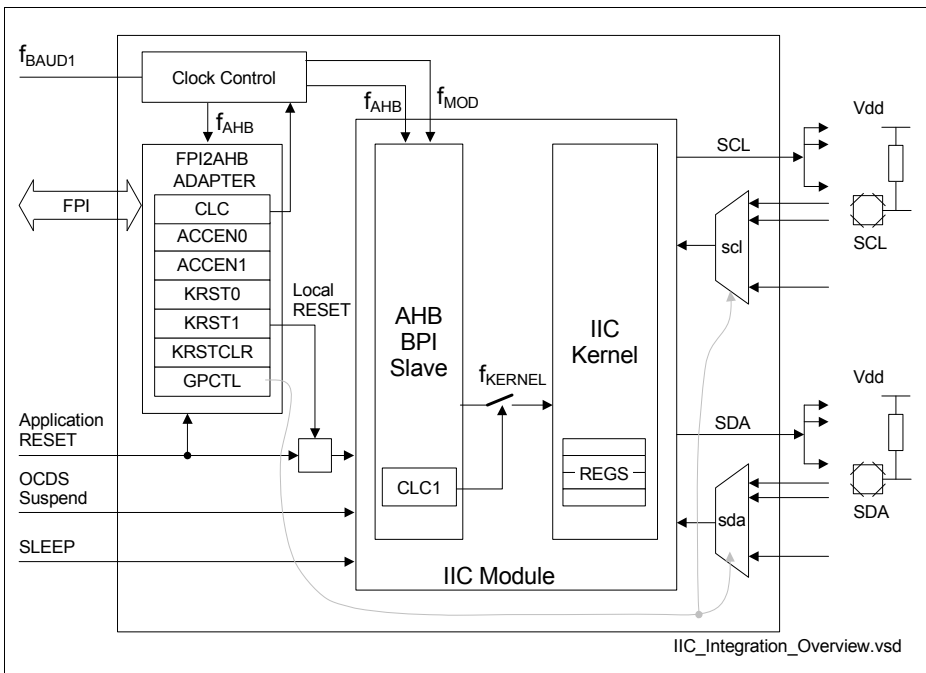


Figure 31-28 Integration Overview

31.5.2 BPI_SPB Module Registers

This section describes the registers implemented in the slave (FPI2AHB) adapter component.

31.5.2.1 System Registers

Figure 31-29 shows all registers associated with the BPI_FPI module, configured for one kernel. See also **Table 31-5** for the address mapping.

BPI_FPI Registers Overview

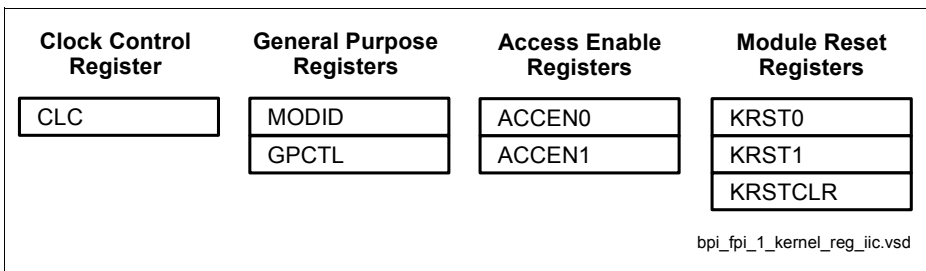


Figure 31-29 BPI_FPI Registers

Kernel Reset

If a kernel reset is requested, the adapter will synchronously assert the internal kernel reset output and error any ongoing accesses on the FPI bus not addressed to the adapter's internal SFRs.

The method of using the kernel reset output from the adapter to initialise the associated AHB module is module specific and outside the scope of this specification.

Any AHB accesses in progress will be synchronously terminated.

GPCTL

A single, adapter specific, SFR is implemented in the adaptor.

The SFR contains general purpose read/write control bits without ENDINIT protection. All the bits are connected to output ports on the FPI2AHB entity. The SFR will implement no safety requirements and is not be suitable for controlling hardware related to safety functions. The SFR supports, byte, half word and word transactions only. All other transactions are rejected with an FPI error termination.

Note: In the I2C module, this register implements the PISEL functionality. The SDA and SCL input multiplexors are controlled in parallel with each PISEL setting. Since 3 bits are implemented for PISEL, up to 8 sets of SDA and SCL pins can be defined.

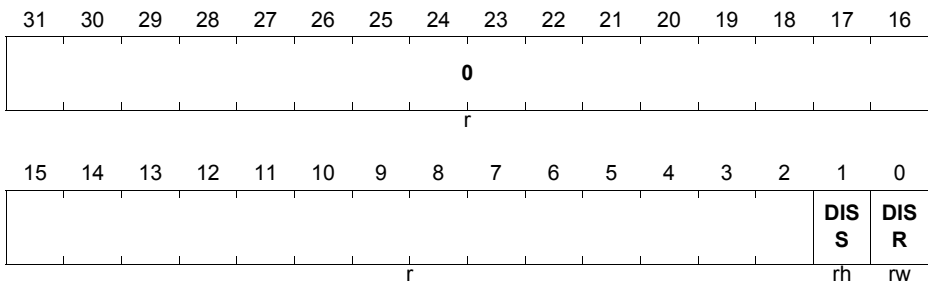
Inter-Integrated Circuit Module (I2C)
Principle of Operation

When the controlling state machine detects an FPI access, it compares the address to the address of the SFR. In the case that the address matches the SFR address, the normal state machine transitions are interrupted and the access is not passed to the AHB bus.

If the access opcode is not SDTB, SDTH or SDTW, then the access is terminated with an FPI error. If the opcode is supported, the SFR data will be returned on the FPI bus if the transaction is a read or the appropriate bits of the register will be updated if the access is a write.

Clock Control Register (CLC)

The Clock Control Register, CLC, acts globally and allows the complete AHB module to be disabled to reduce power consumption when the module is not required. When the module is disabled (DISS=1_B), only register accesses to the adapter register address space are permitted. All other accesses to the module address space will be errored.

I2C0_CLC
Clock Control Register
(10000_H)
Reset Value: 0000 0003_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

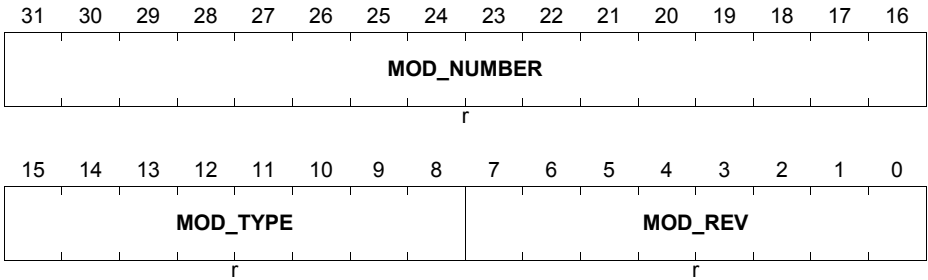
Module Identification Register (ID)

The identification register allows the programmer version-tracking of the module.

Inter-Integrated Circuit Module (I2C)

I2C0_MODID

Module Identification Register (10004_H) Reset Value: 00C2 C0XX_H



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MOD_TYPE	[15:8]	r	Module Type The bit field is set to C0 _H which defines the module as a 32-bit module.
MOD_NUMBER	[31:16]	r	Module Number Value This bit field defines a module identification number.

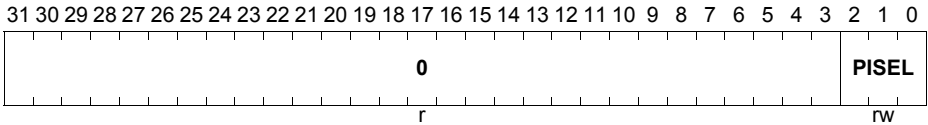
Inter-Integrated Circuit Module (I2C)

General Purpose Control Register

I2C0_GPCTL

General Purpose Control Register (10008_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
PISEL	[2:0]	rw	<p>Port Input Select</p> <p>Used to select the input pins providing the serial data and clock input signals, in the range of 0 to 7.</p> <p>000_B SDA0A and SCL0A are selected. 001_B SDA0B and SCL0B are selected. 010_B SDA0C and SCL0C are selected. 011_B SDA0D and SCL0D are selected. 100_B SDA0E and SCL0E are selected. 101_B SDA0F and SCL0F are selected. 110_B SDA0G and SCL0G are selected. 111_B SDA0H and SCL0H are selected.</p> <p><i>Note: In TC26x B-Step, not all PISEL options are available. See the Ports chapter.</i></p>
0	[31:3]	r	<p>reserved</p> <p>Reads 0. Should be written with 0.</p>

Inter-Integrated Circuit Module (I2C)
Access Enable Register 0

The Access Enable Register 0 controls write access¹⁾ for transactions to registers with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The adapter is prepared for an 6 bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ... ,EN31 -> TAG ID 011111_B.

I2C0_ACCEN0
Access Enable Register 0
(1000C_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Field	Bits	Type	Description
ENn (n = 0-31)	n	r/w	Access Enable for Master TAG ID n This bit enables write access to the module register addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed. Read accesses will be executed. 1 _B Write and read accesses will be executed

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

Inter-Integrated Circuit Module (I2C)

Access Enable Register 1

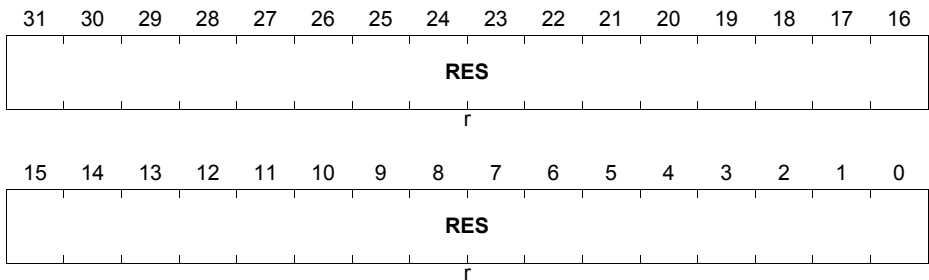
The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). These tags are not used in this system and so no programmable bits are provided.

I2C0_ACCEN1

Access Enable Register 1

(10010_H)

Reset Value: 0000 0000_H



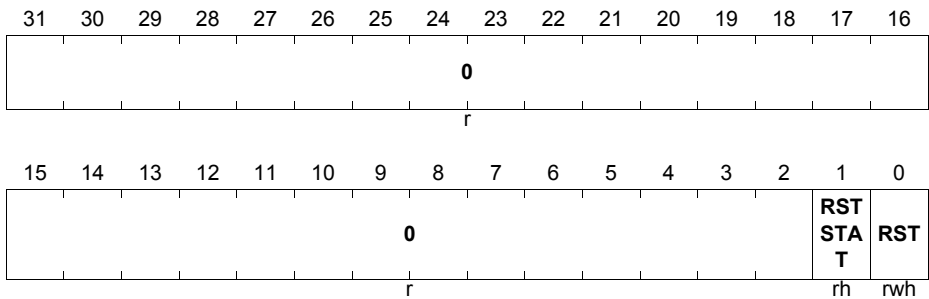
Field	Bits	Type	Description
RES	31:0	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 0 (KRST0)

The Kernel Reset function is used to synchronously reset the AHB module. To activate the kernel reset, it is necessary to set the RST bits by writing with 1_B in both Kernel Reset Registers. The RST bit will be re-set by the adapter with the end of the adapter kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to 1_B in the same clock cycle the RST bit is reset. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLE.CLR register bit.

Note: During the execution of the kernel reset until RSTSTAT is set, write accesses to the module registers will result in an error acknowledge. Adapter registers can still be accessed.

Inter-Integrated Circuit Module (I2C)
I2C0_KRST0
Kernel Reset Register 0
(10014_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (reset to 0 _B) b after the kernel reset was executed.
RSTSTAT	1	rh	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set after the execution of a kernel reset in the same clock cycle both reset bits. 0 _B No kernel reset was executed 1 _B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

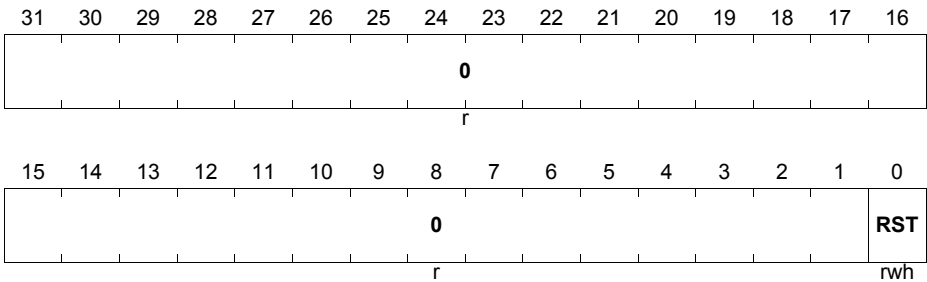
Inter-Integrated Circuit Module (I2C)

I2C0_KRST1

Kernel Reset Register 1

(10018_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set.</p> <p>0_B No kernel reset was requested</p> <p>1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to 0_B) after the kernel reset was executed.</p>
0	[31:1]	r	<p>Reserved</p> <p>Read as 0_B; should be written with 0_B.</p>

Kernel Reset Status Clear Register (KRSTCLR)

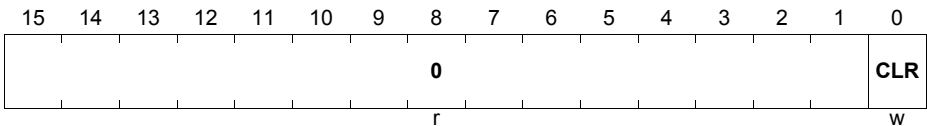
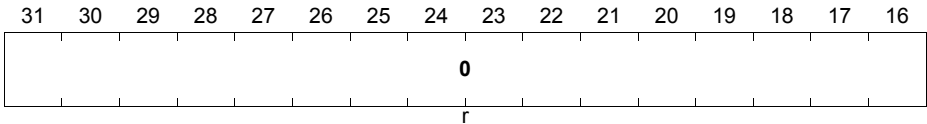
The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

Inter-Integrated Circuit Module (I2C)

I2C0_KRSTCLR

Kernel Reset Status Clear Register (1001C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0 _B .
0	[31:1]	r	Reserved Read as 0 _B ; should be written with 0 _B .

32 Input Output Monitor (IOM)

32.1 Overview

The Input Output Monitor (IOM) module serves as a smart I/O comparison unit, observing and checking for the correct operation of system peripheral outputs that may serve and/or control externally-attached hardware, as well as the correct operation of the hardware itself, including any sensors whose signals may serve as an input to the monitoring function.

The monitoring function may be achieved in a number of configurable ways depending upon the application and the type of safety measure to be deployed. (Examples of such measures are included later-on within this chapter). Triggering via software is also permissible in this manner.

Monitor and reference signals (where needed) are taken from applicable system peripherals (i.e. Capture/Compare Units and General Purpose Timer Units) where they connect to internal padlogic (that feeds/controls I/O Ports), which may or may not be driven to the pads (configuration dependent). In this manner connectivity can be made to a signal that is to be driven externally, and to another signal that may serve as a reference, although may not be driven off-chip. A driven signal may also serve as a reference in order to compare against another that may be provided via a GPIO input, e.g. from a sensor external to the device.

Up to 16 monitoring points can be configured, with the capability to generate a system event that may be driven from one individual, several individual or multiple occurrences of a monitored condition, or a combination of several depending upon configuration.

32.2 Features

- 16 Filter & Prescaler Channels, each connected to a specific pad (Pn_IN), the output of which to serve as a monitor or reference signal.
- 16 Logic Analyzer Modules (LAMs), with each channel comprising a mux to independently select monitor and reference signal inputs, with which to undertake the compare. A local event (i.e. internal to IOM) is generated when parity or disparity occurs between the selected monitor and reference signals. Configurable via register settings.
- 1 EXOR combiner, configurable (via register) to select a range of upto 8 GTM inputs in order to generate a combined signal to serve as a reference.
- Event Combiner Module (ECM), that takes the 16 local event signals (1 from each of the LAM modules) and generates a system event signal (connecting to system Safety Management Unit) from a single, combination or multiple local event(s). Configurable via register settings.

Input Output Monitor (IOM)

- System Peripheral Bus (SPB) interface for configuration and status register interaction.

32.3 Interfaces

The IOM features the following interfaces:

- System Peripheral Bus (SPB) - for the configuration of constituent parts of the IOM and the interrogation of associated status registers/bitfields.
- Multiple port logic/module connectivity, serving as reference or monitor signals. See SoC Integration section within this chapter for device-specific connections to GTM, GPT12, QSPI, CCU, GPIO (for GTM, etc.).
- Safety Management Unit (SMU) - a single global event signal used inform the SMU of a generated event within the IOM.
- System Signals - FPI Clock, GTM Clock, System Reset.

Note: The functionality of the IOM is independent from system debug, and no connectivity or mode of operation exists for this purpose.

Input Output Monitor (IOM)

32.4 Kernel Description

In addition to the Logic Analyzer Modules, Filter & Prescaler Blocks and Event Combiner Module, the IOM also features a System Peripheral Bus (SPB) interface for configuration and status register access. Note: All the configuration registers are protected by the register access protection mechanism (global definition).

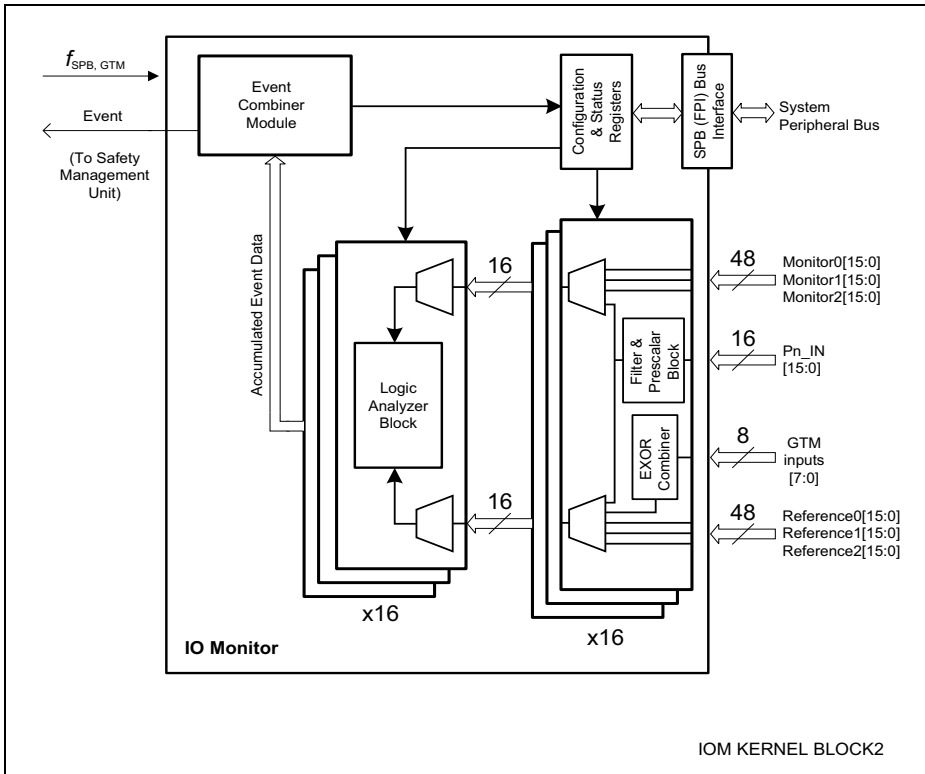


Figure 32-1 IO Monitor Block Diagram

The operational clock for the IOM is derived by combining the SPB and GTM clocks as a logical OR, the resultant frequency of which is therefore determined to be the higher of the two frequencies (assuming a synchronous relationship between the two, both being derived from the same system-level clock). This provides the capability for the IOM to sample GTM-related signals when the SPB (FPI) clock is lower than that of the GTM within the system.

Input Output Monitor (IOM)

32.5 Filter & Prescaler Channel Description

The IOM instances 16 Filter & Prescaler Channels(FPC's) for the purposes of preconditioning and filtering the signals received from the pads (Pn_IN inputs) that may be used as a reference or monitor.

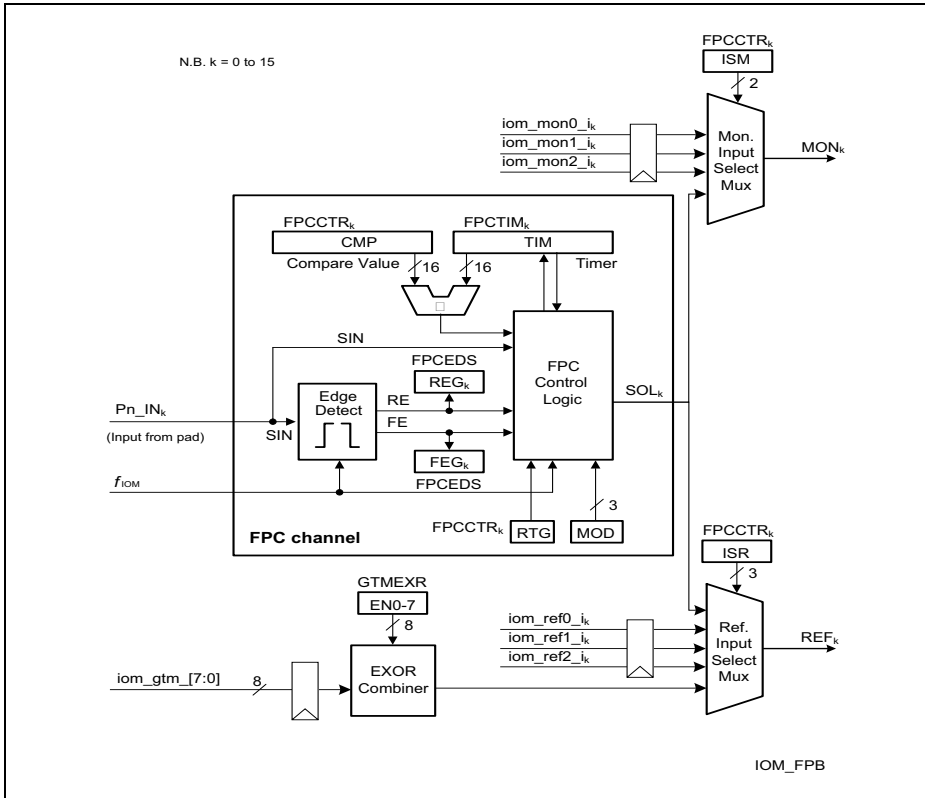


Figure 32-2 Monitor & Reference Selection logic incorporating Filter and Prescaler channel

As shown in Figure 32-2, each FPC channel (cell) is equipped with an signal input multiplexer, a clock multiplexer, an edge detection circuitry, a 16-bit timer, a 16-bit compare register, a 16-bit comparator, and a FPC control circuitry. The edge detection circuitry detects respective edges for the prescaler modes and detects glitches in all other modes.

FPC Registers

The following registers are assigned to the Filter and Prescaler Cells FPC_k ($k = 0-15$):

- FPCESR = Filter and Prescaler Cells Rising and Falling Edge Status Register.
- FPCCTR_k = Filter and Prescaler Cell Control Register k
- FPCTIM_k = Filter and Prescaler Cell Timer Register k

FPC Operating Modes

Each filter and prescaler cell can be individually configured to operate in one of the following operating modes:

- Delayed Debounce Filter Mode on both edges
- Immediate Debounce Filter Mode on both edges
- Rising edge: Immediate Debounce Filter Mode, falling edge: No filtering
- Rising edge: No filtering, falling edge: Immediate Debounce Filter Mode
- Rising edge: Delayed Debounce Filter Mode, falling edge: Immediate Debounce Filter Mode
- Rising edge: Immediate Debounce Filter Mode, falling edge: Delayed Debounce Filter Mode
- Prescaler Mode (triggered by edge detection circuitry on rising edge)
- Prescaler Mode (triggered by edge detection circuitry on falling edge)

The operation mode is selected by bit field FPCCTR_k.MOD.

FPC Input Signal notes

The maximum FPC input signal frequency must be less than or equal to the sampling rate ($f_{FPI}/2$).

FPC Output Signal notes

A level output, representative signal SOL_k, indicating the direction of the detected signal transition, that can be configured to connect to the Logic Analyzer Module_m.

Delayed Debounce Filter Mode

In Delayed Debounce Filter Mode, the signal input SIN is filtered from all signal transitions and glitches with a width smaller than the selected clock period length multiplied by the compare register value.

The input signal SIN is analyzed at the selected filter clock rate of f_{IOM} . If the state of the input sample differs from the current output signal value, the 16-bit timer is increased by one. When the timer register $FPCTIM_k$ is not in its idle state (0000_H) **and** the state of the input sample matches the current output signal value, the 16-bit timer is decremented by one (see [Figure 32-3](#)); if bit $FPCTR_k.RTG$ is set, the timer will be set to idle state again (see [Figure 32-4](#)). A rising or falling edge, occurring on the signal input line SIN when the timer is greater than zero but less than the compare value, sets the corresponding glitch flag $FPCRES.REG$ (on rising edge glitch) or $FPCFES.FEG$ (on falling edge glitch). When the timer matches the 16-bit compare value stored in $FPCTR_k.CMP$ (timer threshold), the level output signal line SOL_k is inverted, and the timer is reset to 0000_H. The rising/falling edge glitch flags must be reset by software.

The filter is by-passed if the compare value $FPCTR_k.CMP$ is programmed to zero (0000_H). In this case, the input signal is directly copied to the output signal.

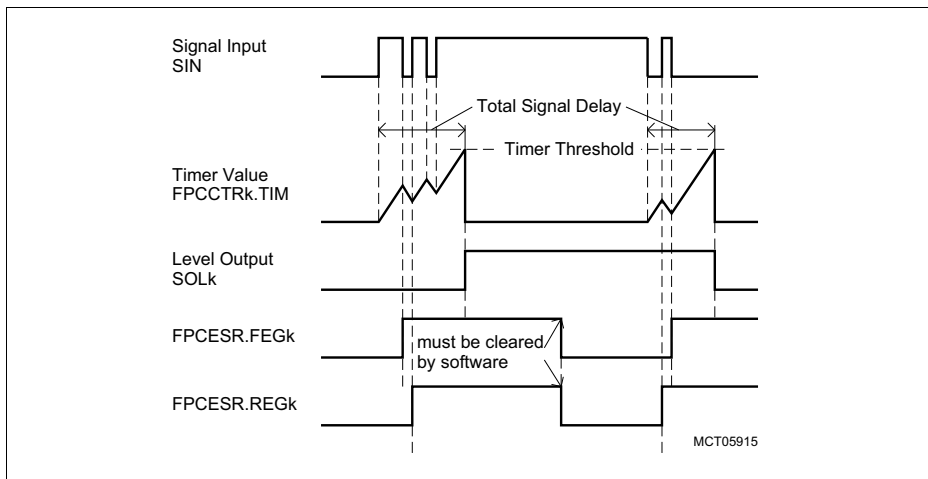


Figure 32-3 FPC Delayed Debounce Filter Algorithm with Timer Decrement

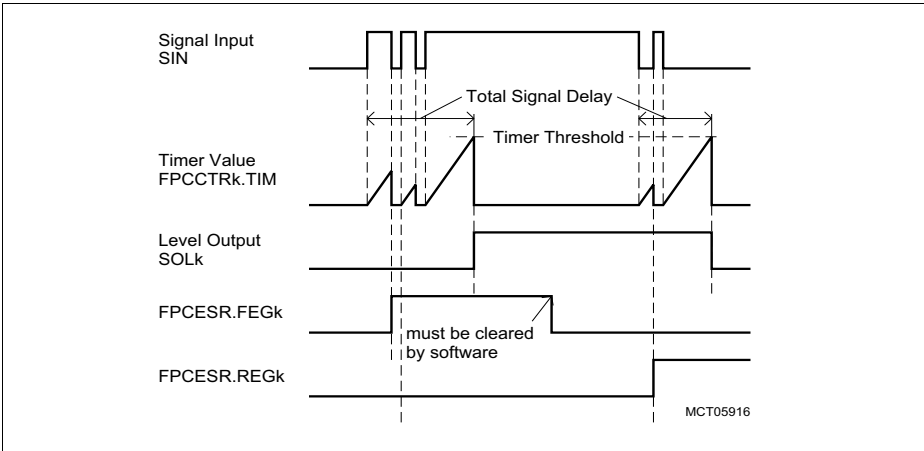


Figure 32-4 FPC Delayed Debounce Filter Algorithm with Timer Reset

The total signal delay from input to output depends on the programmed compare register value, the number of high-frequency pulses (glitches) during the filter operating time, and the timer behaviour in case of a glitch (decrement or reset).

The FPC Delayed Debounce Filter Mode is selected by:

- $FPCCTR_k.MOD = 000_B$

Input Output Monitor (IOM)

Immediate Debounce Filter Mode

In Immediate Debounce Filter Mode, the input signal is filtered from signal transitions and glitches arriving a programmable time after an input signal edge detection (see [Figure 32-5](#)).

The input signal SIN is sampled with f_{FPI} and the input signal SIN edge detection is also performed with f_{FPI} . The further analysis (e.g. filter timer increment, glitch detection) is also achieved with f_{FPI} .

As long as the timer is reset, the FPC control circuitry copies the sampled input value directly to the level output signal line SOLk. When a rising or falling edge occurs on the signal input line SIN and the 16-bit compare value $FPCCTR_k.CMP$ is not zero, the timer is enabled to be increased by the selected clock and the copy mechanism is disabled. When the timer value $FPCTIM_k.TIM$ matches the compare value $FPCCTR_k.CMP$, the timer is reset and the copy mechanism is enabled again. A rising or falling edge, occurring on SIN while the timer is greater than zero but less than the compare value, sets the corresponding glitch flag $FPCRES.REG$ (on rising edge glitch) or $FPCFES.FEG$ (on falling edge glitch). The rising/falling edge glitch flags must be reset by software.

The filter is by-passed if the compare value $FPCCTR_k.CMP$ is programmed to zero (0000_H). In this case, the input signal is directly copied to the output signal without any disable periods.

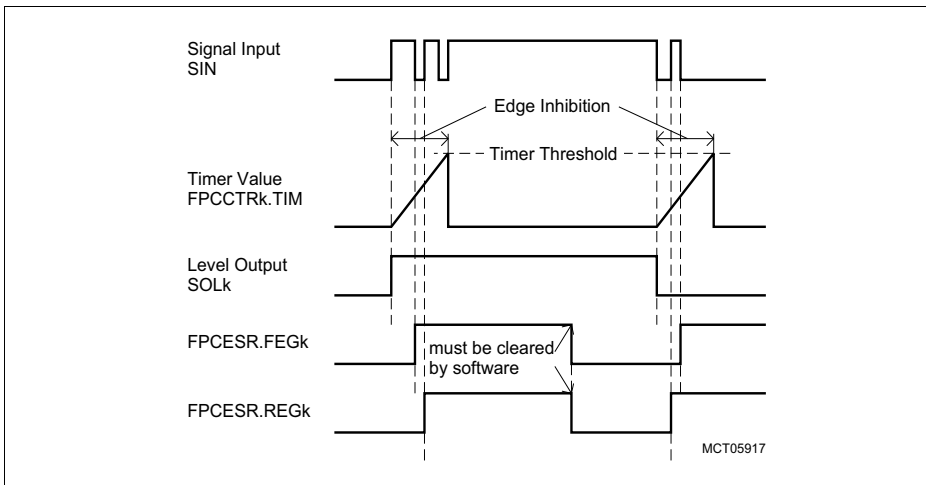


Figure 32-5 FPC Immediate Debounce Filter Algorithm on Both Edges

Note: During the last clock cycle of edge inhibition time (where timer value is equal to the compare value) an input signal glitch will be filtered but the corresponding glitch status flag in register FPCRES/FPCFES is not set.

Input Output Monitor (IOM)

The Immediate Debounce Filter can be enabled only for one edge, either rising or falling. In this case, the signal output follows the signal input value immediately after the timer threshold of the filtered edge is reached, without re-starting the timer (see **Figure 32-6**).

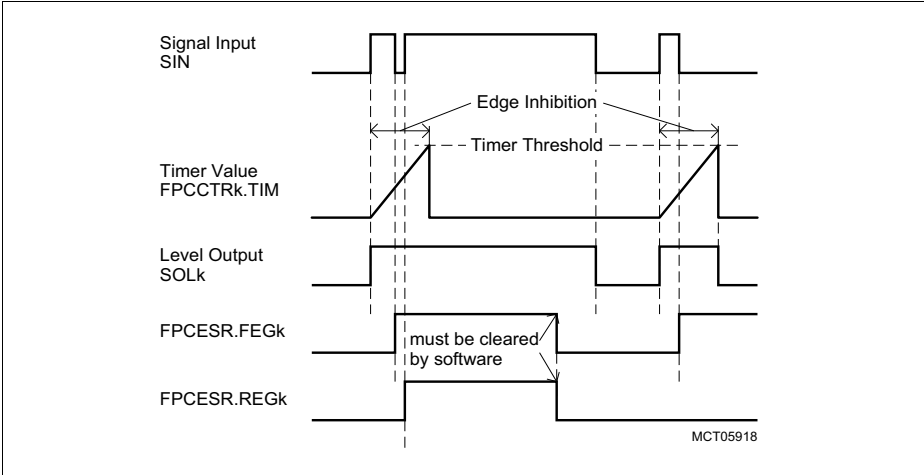


Figure 32-6 FPC Immediate Debounce Filter Algorithm on Rising Edge only

The FPC Immediate Debounce Filter Modes are selected by:

- $FPCCTR_k.MOD = 001_B$: Immediate Debounce Filter Mode on both edges
- $FPCCTR_k.MOD = 010_B$: Immediate Debounce Filter Mode on rising edge only, no filtering on falling edge.
- $FPCCTR_k.MOD = 011_B$: Immediate Debounce Filter Mode on falling edge only, no filtering on rising edge.

Mixed Filter Modes

In the Mixed Filter Modes, one edge of a signal is filtered in the Delayed Debounce Mode, and the other edge is filtered in the Immediate Debounce Mode. The Debounce Mode is switched when the timer threshold is reached. Note that both filter modes use the same timer threshold in this case (see [Figure 32-7](#), demonstrating Delayed Debounce Mode with Timer Decrement on Rising Edge and Immediate Debounce of on Falling Edge).

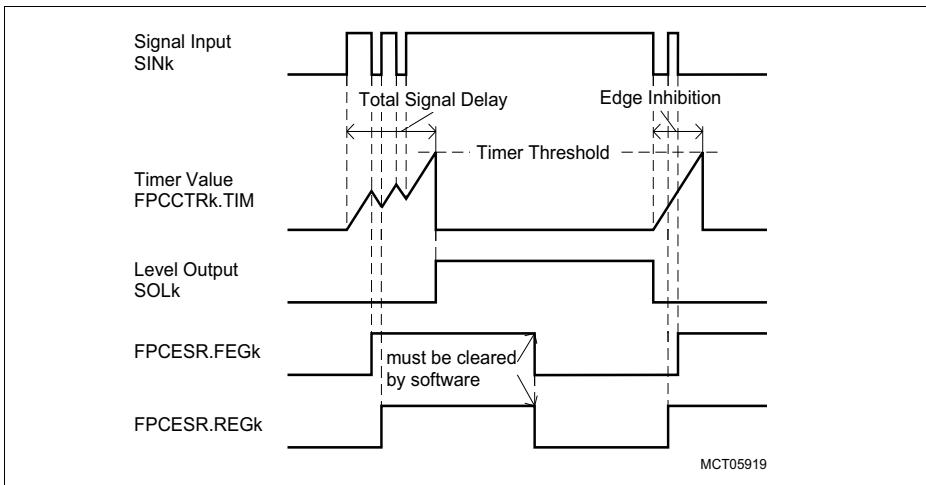


Figure 32-7 FPC Mixed Filter Algorithm

The FPC Mixed Filter Modes are selected by:

- $FPCCTR_k.MOD = 100_B$: Delayed Debounce Filter Mode on rising edge
Immediate Debounce Filter Mode on falling edge
- $FPCCTR_k.MOD = 101_B$: Immediate Debounce Filter Mode on rising edge
Delayed Debounce Filter Mode on falling edge

Prescaler Mode

In Prescaler Mode, the input signal is sampled and analyzed with f_{FPI} . The FPC control circuitry counts each rising (or falling) edge of the input signal. When the timer value matches the compare value:

- one IOM module clock pulse is generated at the level output signal SOL_k
- the timer $FPCTIM_k.TIM$ is reset to 0000_H

Figure 32-8 shows a divide-by-6 operation using the FPC in Prescaler Mode with trigger on rising edge selected.

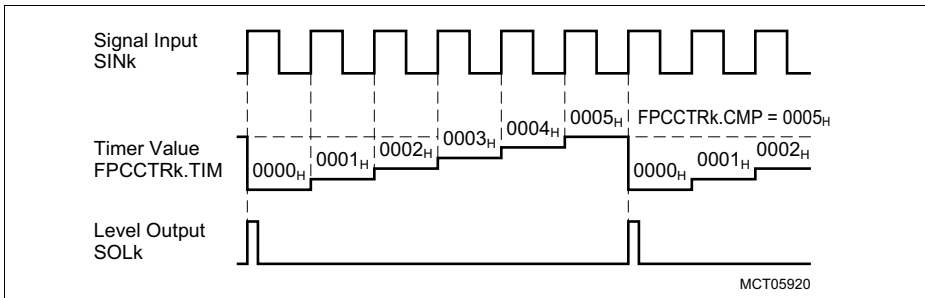


Figure 32-8 FPC Prescaler Mode

For a divide-by-n operation, the compare value $FPCCTR_k.CMP$ must be set to $n - 1$.

The FPC Prescaler Modes are selected by:

- $FPCCTR_k.MOD = 110_B$: Prescaler Mode triggered by edge detection circuitry on rising edge
- $FPCCTR_k.MOD = 111_B$: Prescaler Mode triggered by edge detection circuitry on falling edge

32.6 EXOR Combiner Description

The EXOR Combiner can be configured to accept up to 8 external inputs from GTM module outputs, used to replicate a typical *combined* signal from an automotive multi-phase motor. The output from which can then be used as a reference to monitor such a motor.

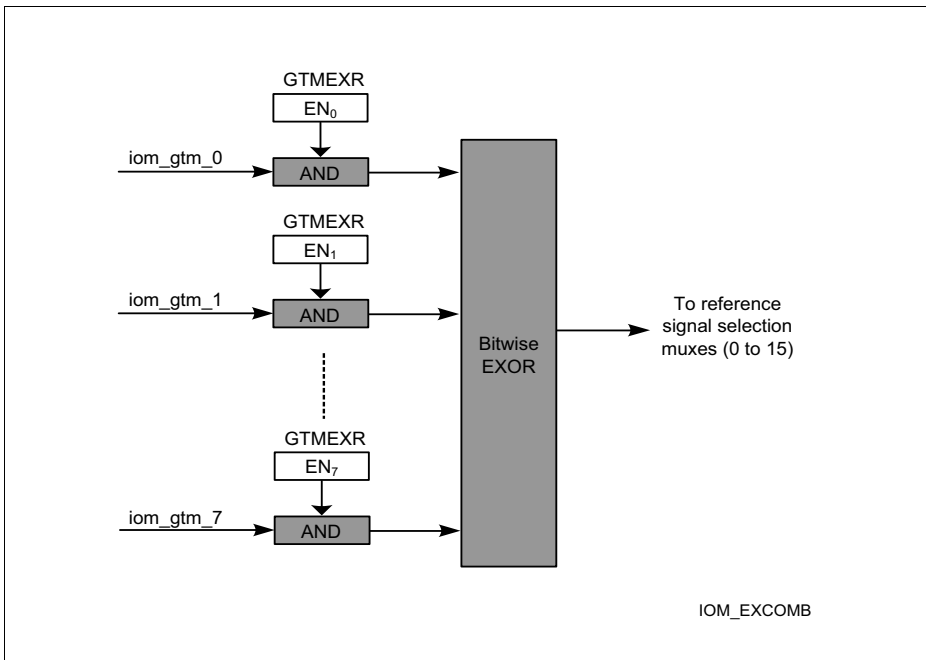


Figure 32-9 EXOR Combiner block diagram

EXOR Combiner Registers

The following registers are assigned to the block:

- GTMEXR = GTM Input EXOR Combiner Selection Configuration Register.

EXOR Combiner Input Signals

- 8 inputs from the GTM module(s).

EXOR Combiner Output Signals

- 1 output signal reflecting the combined logical EXOR function on the associated (and enabled) input(s).

32.7 Logic Analyzer Module (LAM) Description

The IOM instances 16 Logic analyzer Modules (LAM's) for the purposes of monitor and reference signal comparison. Each block accepts one monitor and one reference level signal from the 16 monitor & 16 reference signals available, from the FPC/Input selection block. The behaviour of each LAM is set by configuring the associated register settings.

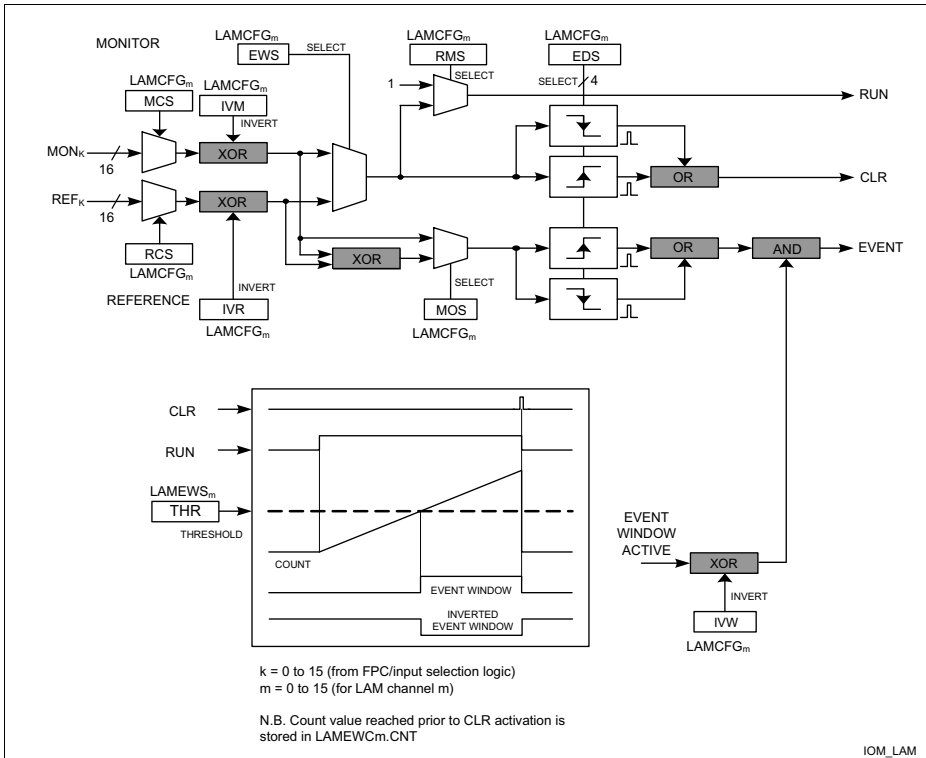


Figure 32-10 Logic Analyzer Module (LAM) block diagram

As shown in Figure 32-9, each Logic Analyzer Module features inputs for the Reference & Monitor points, fed from the requisite Filter & Prescaler block. Depending upon the Logic Analyzer Module configuration register (LAMCFG), and the event window settings register (LAMEWS), the block may be configured to monitor a particular type of signal behaviour, input from sensors external to the device in order to monitor, govern and/or control external application behaviour with particular regard to any given safety requirements.

Input Output Monitor (IOM)

LAM Registers

The following registers are assigned to the Logic Analyzer Module LAM_m (m = 0-15):

- LAMCFG_m = Logic Analyzer Module Configuration Register m
- LAMEWS_m = Logic Analyzer Module Event Window Settings Register m
- LAMEWC_m = Logic Analyzer Module Event Window Count Status Register m

LAM Input Signals

Two inputs exist for each LAM_m:

- Reference signal input, fed from the requisite Filter & Prescaler block.
- Monitor signal input, fed from the requisite Filter & Prescaler block.

Depending upon the required function, both inputs or just the Monitor input may be utilised.

LAM Output Signals

One output is provided by each LAM:

- Event trigger output, configured to be active when a monitored signal falls outside the intended timed and/or periodic behaviour.

32.8 Event Combiner Module (ECM) Description

The IOM instances one Event Combiner Module, which takes the event outputs from each of the 16 Logic Analyzer Modules (LAM's). Some or all of these events can then be combined in a variety of configurable ways in order to generate a single system event.

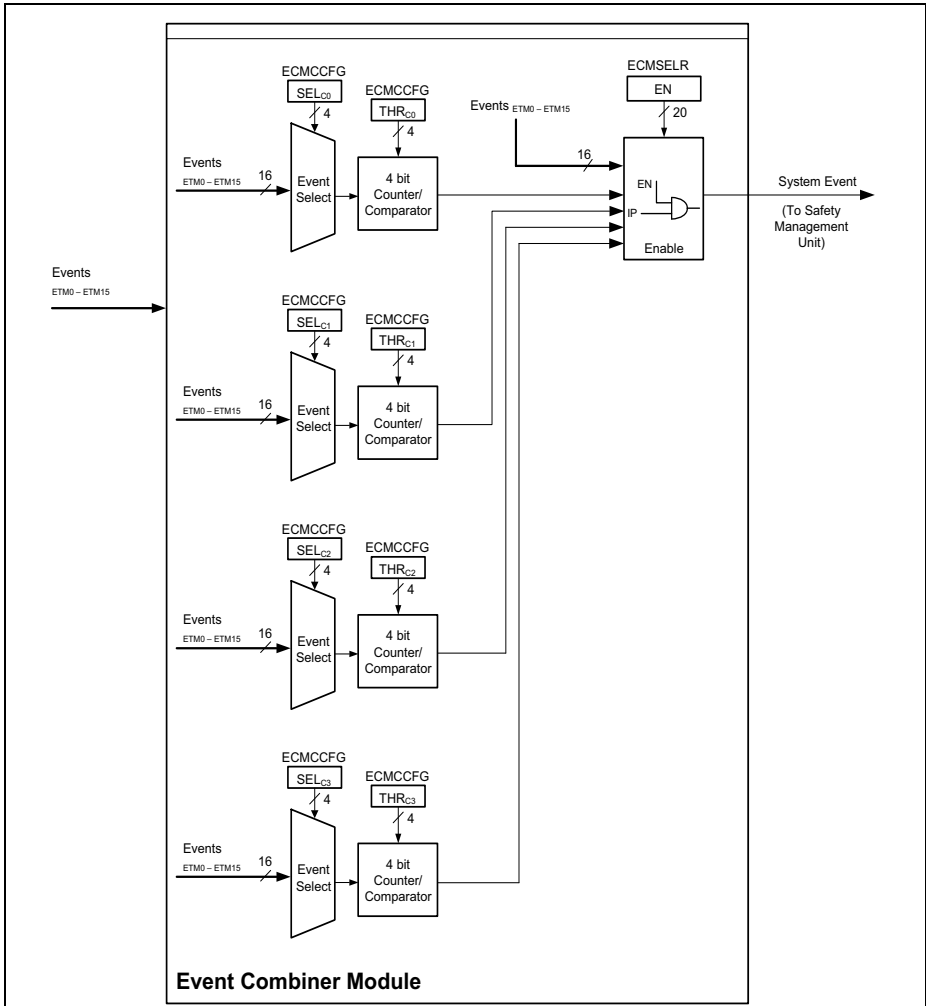


Figure 32-11 Event Combiner Module (ECM) block diagram

Input Output Monitor (IOM)

Four separate 4bit counters with configurable thresholds are also available to be assigned to count multiple occurring events from any of the 16 available LAM's. The output of these counter units (active once a count threshold has been met/surpassed) can also be included (via configuration) within the generation of the system event. (Note: the alarm output will be connected to the Safety Management Unit, SMU.EMM).

The ECM also includes two Event Trigger History (ETMETH0/1) registers, updated with each system event, used to record the status of the 16 LAM event outputs, thereby identifying which triggers were responsible.

In addition, the ECM incorporates a System Peripheral Bus (SPB) interface for writing configuration and reading status registers appertaining to all subblock within the IOM.

ECM Registers

The following registers are assigned to the ECM:

- ECMCCFG = ECM counter configuration register (for the four 4bit local event counters).
- ECMSELR = ECM global event selection register (for the selection of local events, or multiples thereof, required to generate the global event signal).

ECM Input Signals

- From Logic Analyzer Modules (LAM's)
 - Local event triggers from LAM blocks (0 to 15).
- System Peripheral Bus.

ECM Output Signals

One output is generated by each LAM:

- Global event trigger output, intended to be connected to the Safety Management Unit.
- System Peripheral Bus.

Input Output Monitor (IOM)

32.9 IOM Registers

This section describes the kernel registers of the IOM unit.

All read and write accesses to an undefined address location within the IOM address space will result in a bus error on the System Peripheral Bus (SPB).

IOM Kernel Register Overview

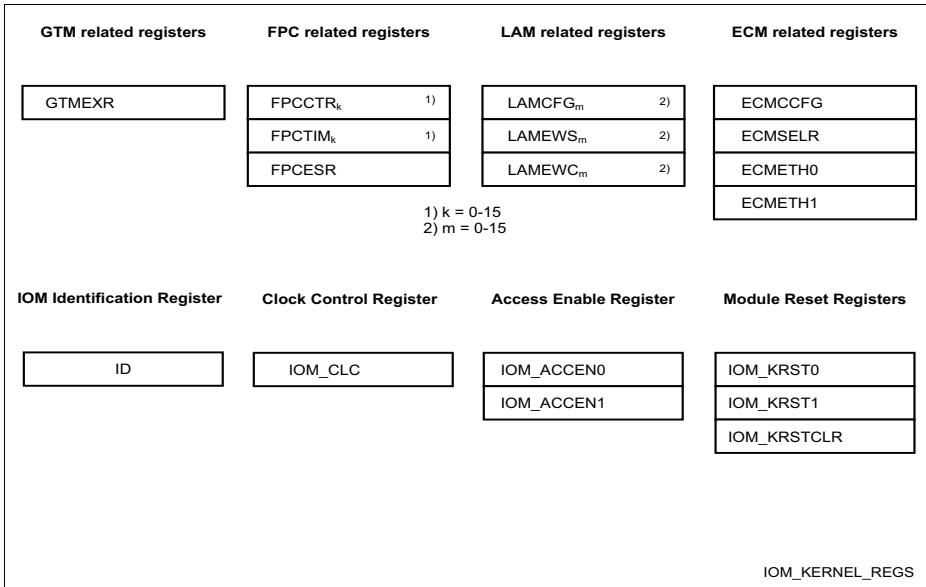


Figure 32-12 IOM Kernel Registers

In the TC26x B-Step, the registers of the IOM units are located in the following address ranges.

Table 32-1 Registers Address Space

Module	Base Address	End Address	Note
IOM	F003 5000 _H	F003 51FF _H	-

Table 32-2 Registers Overview - IOM Kernel Registers

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Reset Class	Description see
			Read	Write		
IOM_CLC	Clock Control Register	000 _H	U, SV	SV, E, P	3	Page 37
ID	IOM Identification Register	008 _H	U, SV	BE	3	Page 20
IOM_KRS TCLR	Reset Status Clear Register	01C _H	U, SV	SV, E, P	3	Page 43
IOM_KRS T1	Reset Control Register 1	020 _H	U, SV	SV, E, P	3	Page 42
IOM_KRS T0	Reset Control Register 0	024 _H	U, SV	SV, E, P	3	Page 40
IOM_ACC EN1	Access Enable Register 1	028 _H	U, SV	SV, SE	3	Page 39
IOM_ACC EN0	Access Enable Register 0	02C _H	U, SV	SV, SE	3	Page 38
ECMCCFG	Event Combiner Module Counter Configuration Register	030 _H	U, SV	U, SV	3	Page 32
ECMSELR	Event Combiner Module Global Event Selection Register	034 _H	U, SV	U, SV	3	Page 34
ECMETH0	Event Trigger History Register 0	038 _H	U, SV	U, SV	3	Page 35
ECMETH1	Event Trigger History Register 1	03C _H	U, SV	U, SV	3	Page 36
GTMEXR	GTM Input EXOR Combiner Selection Register	040 _H	U, SV	U, SV	3	Page 25
FPCESR	Filter and Prescaler Cells Rising & Falling Edge Status Register	078 _H	U, SV	U, SV	3	Page 21
FPCCTR _k	Filter and Prescaler Cell Control Register k (k = 0-15)	080 _H + k * 4)	U, SV	U, SV	3	Page 22

Input Output Monitor (IOM)
Table 32-2 Registers Overview - IOM Kernel Registers

Register Short Name	Register Long Name	Offset Addr.	Access Mode		Reset Class	Description see
			Read	Write		
FPCTIM _k	Filter and Prescaler Cell Timer Register k (k = 0-15)	0C0 _H + k * 4	U, SV	U, SV	3	Page 24
LAMEWC _m	Logic Analyzer Module Event Window Count Status Register m (m = 0-15)	100 _H + m * 4	U, SV	BE	3	Page 27
LAMCFG _m	Logic Analyzer Module Configuration Register m (m = 0-15)	180 _H + m * 4	U, SV	U, SV	3	Page 28
LAMEWS _m	Logic Analyzer Module Event Window Setting Register m (m = 0-15)	1C0 _H + m * 4	U, SV	U, SV	3	Page 31

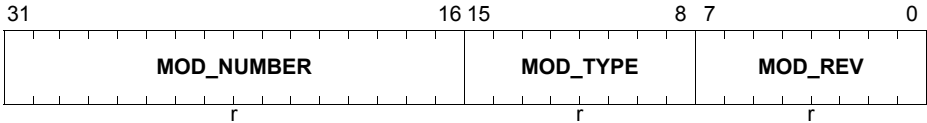
Input Output Monitor (IOM)

32.9.1 IOM Identification Register (IOM_ID)

The IOM Identification Register ID contains read-only information about the module version.

IOM_ID

IOM Identification Register (008_H) Reset Value: 00CC C001_H



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number MOD_REV defines the Module revision number. The value of a module revision starts with 01 _H (first revision).
MOD_TYPE	[15:8]	r	Module Number Value This bit field defines the module as a 32 bit module: C0 _H
MOD_NUM	[31:16]	r	Module Number Value This bit field defines the identification number for the IOM.

32.9.2 Filter & Prescaler Cell (FPC) Registers

IOM Filter and Prescaler Cells Rising & Falling Edge Status Register (IOM_FPCESR)

The Filter and Prescaler Edge Status Register stores the state of detected rising and falling edges from each of the FPC cells.

The register can be written to selectively clear individual bits.

Individual bits are also cleared with a write to the control register (FPCCTRk) or timer register (FPCTIMk).

IOM_FPCESR

IOM Filter and Prescaler Cells Rising & Falling Edge Status Register

 (078_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REG 31	REG 30	REG 29	REG 28	REG 27	REG 26	REG 25	REG 24	REG 23	REG 22	REG 21	REG 20	REG 19	REG 18	REG 17	REG 16
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FEG 15	FEG 14	FEG 13	FEG 12	FEG 11	FEG 10	FEG 9	FEG 8	FEG 7	FEG 6	FEG 5	FEG 4	FEG 3	FEG 2	FEG 1	FEG 0
rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh	rwh

Field	Bits	Type	Description
FEGk (k = 15-0)	k	rwh	Falling Edge Glitch Flag for FPCk 0 _B No falling edge of glitch detected during filtering 1 _B Falling edge of glitch detected during filtering
REGk (k = 15-0)	16+k	rwh	Rising Edge Glitch Flag for FPCk 0 _B No rising edge of glitch detected during filtering 1 _B Rising edge of glitch detected during filtering

Input Output Monitor (IOM)

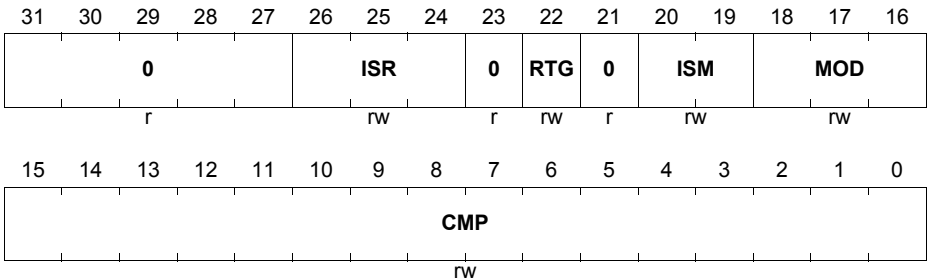
IOM filter & Prescaler Cell Control register (IOM_FPCCTRk)

The IOM Filter & Prescaler Control Register is used to configure the FPC and mux selection logic.

IOM_FPCCTRk (k = 0-15)

IOM Filter and Prescaler Cell Control Register_k
 (080_H+k*4_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CMP	[15:0]	rw	Threshold Value of Filter & Prescaler Cell k CMP is the 16-bit threshold value that is compared with the 16-bit timer value FPCTIMk.TIM.
MOD	[18:16]	rw	Operation Mode Selection for Filter & Prescaler Cell k 000 _B Delayed Debounce Filter Mode on both edges 001 _B Immediate Debounce Filter Mode on both edges 010 _B Rising edge: Immediate Debounce Filter Mode, falling edge: no filtering 011 _B Rising edge: no filtering, falling edge: Immediate Debounce Filter Mode 100 _B Rising edge: Delayed Debounce Filter Mode, falling edge: Immediate Debounce Filter Mode 101 _B Rising edge: Immediate Debounce Filter Mode, falling edge: Delayed Debounce Filter Mode 110 _B Prescaler Mode (triggered on rising edge) 111 _B Prescaler Mode (triggered on falling edge)

Input Output Monitor (IOM)

Field	Bits	Type	Description
ISM	[20:19]	rw	Monitor Input Signal Selection for Filter & Prescaler Cell k IPS determines the signal input used for edge detection. 00 _B Signal input Pn_IN (from portlogic) selected 01 _B Monitor Signal Input 0 selected 10 _B Monitor Signal Input 1 selected 11 _B Monitor Signal Input 2 selected
0	21	r	Reserved Read as 0; should be written with 0.
RTG	22	rw	Reset Timer behaviour for Filter & Prescaler Cell k on Glitch 0 _B Timer for FPCK is decremented on glitch 1 _B Timer for FPCK is cleared on glitch This bit is effective in Delayed Debounce Filter Mode only.
0	23	r	Reserved Read as 0; should be written with 0.
ISR	[26:24]	rw	Reference Input Signal Selection for Filter & Prescaler Cell k ISM determines the input used for the channel k reference signal. 000 _B Signal input Pn_IN (from portlogic) selected 001 _B Reference Signal Input 0selected 010 _B Reference Signal Input 1selected 011 _B Reference Signal Input 2selected 1xx _B GTM XOR Combiner selected
0	[31:27]	r	Reserved Read as 0; should be written with 0.

Input Output Monitor (IOM)

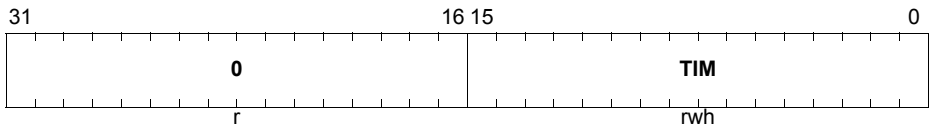
IOM Filter & Prescaler Cell k Timer Register (IOM_FPCTIMk)

The IOM Filter & Prescaler Cell Timer Register is used to set the value of the timer for the required application.

IOM_FPCTIMk (k = 0-15)

IOM Filter and Prescaler Cell kTimer Register_k
(0C0_H+k*4_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TIM	[15:0]	rwh	Timer Value of Filter and Prescaler Cell k N.B. a write of any value to the TIM bitfield will result in it's contents being set to the reset value. This is also undertaken whenever the IOM_FPCCTRk register is written with a new value.
0	[31:16]	r	Reserved Read as 0; should be written with 0.

32.9.3 GTM Input Related Registers

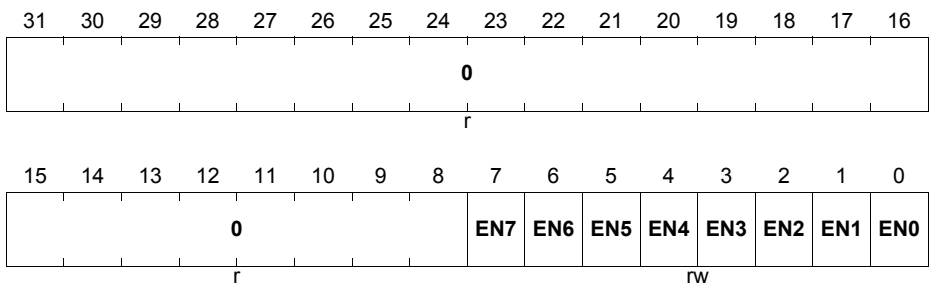
IOM EXOR Combiner GTM signal input Selection Register (IOM_GTMEXR)

Enables the incoming GTM-connected input signals for inclusion within a combined EXOR function.

IOM_GTMEXR

IOM GTM Input EXOR Combiner Selection Register

 (040_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
EN0	0	rw	GTM input 0 selection for EXOR combiner 0 _B Input not selected for EXOR combiner. 1 _B Input selected for EXOR combiner.
EN1	1	rw	GTM input 1 selection for EXOR combiner 0 _B Input not selected for EXOR combiner. 1 _B Input selected for EXOR combiner.
EN2	2	rw	GTM input 2 selection for EXOR combiner 0 _B Input not selected for EXOR combiner. 1 _B Input selected for EXOR combiner.
EN3	3	rw	GTM input 3 selection for EXOR combiner 0 _B Input not selected for EXOR combiner. 1 _B Input selected for EXOR combiner.
EN4	4	rw	GTM input 4 selection for EXOR combiner 0 _B Input not selected for EXOR combiner. 1 _B Input selected for EXOR combiner.
EN5	5	rw	GTM input 5 selection for EXOR combiner 0 _B Input not selected for EXOR combiner. 1 _B Input selected for EXOR combiner.

Input Output Monitor (IOM)

Field	Bits	Type	Description
EN6	6	rw	GTM input 6 selection for EXOR combiner 0 _B Input not selected for EXOR combiner. 1 _B Input selected for EXOR combiner.
EN7	7	rw	GTM input 7 selection for EXOR combiner 0 _B Input not selected for EXOR combiner. 1 _B Input selected for EXOR combiner.
0	[31:8]	r	Reserved Read as 0; should be written with 0.

32.9.4 Logic Analyzer Module (LAM) Registers

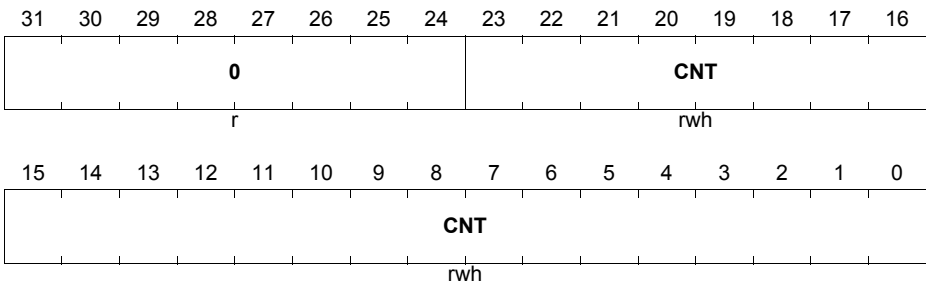
IOM Logic Analyzer Module Event Window Count Status register (IOM_LAMEWCm)

Used to store the window count value reached prior to being cleared in the LAM block once an event has been generated.

Note: A write to this register will result in a bus error, i.e.existing contents unaffected.

IOM_LAMEWCm (m = 0-15)

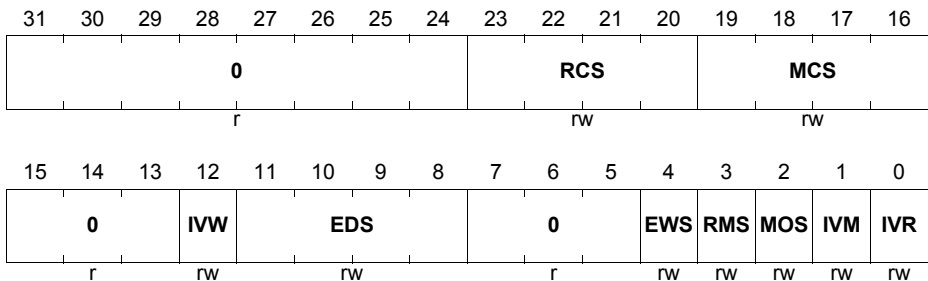
IOM Logic Analyzer Module Event Window Count Status Register_m
 (100_H+m*4_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
CNT	[23:0]	r	Event Window Count Value LAM block m The count value of the event window attained coincident with an event occurring.
0	[31:24]	r	Reserved Read as 0.

Input Output Monitor (IOM)
IOM Logic Analyzer Module Configuration Register (IOM_LAMCFG_m)

Use to configure the application-specific settings for each LAM block, including signal selection muxes at the block inputs.

IOM_LAMCFG_m (m = 0-15)
IOM Logic Analyzer Module Configuration Register_m
(180_H+m*4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
IVR	0	rw	Invert Reference LAM block m This bit field determines whether the reference signal from the FPCrch is inverted or not. 0 _B Don't invert reference signal from FPC. 1 _B Invert reference signal from FPC.
IVM	1	rw	Invert Monitor LAM block m This bit field determines whether the monitor signal from the FPCmch is inverted or not. 0 _B Don't invert monitor signal from FPC. 1 _B Invert monitor signal from FPC.
MOS	2	rw	Monitor Source Select LAM block m This bit field determines whether the monitor signal from the FPCmch is sourced directly or compared (EXOR'd) with the reference signal from the FPCrch for the event compare. 0 _B Monitor signal is sourced directly from FPCmch. 1 _B Monitor signal is EXOR'd with FPCrch.

Input Output Monitor (IOM)

Field	Bits	Type	Description
RMS	3	rw	Runmode Select LAM block m This bit field determines whether the event window generation is free-running or gated with the monitor or reference. 0 _B Event window generation is free-running. 1 _B Event window generation is gated with the monitor or reference signal.
EWS	4	rw	Event Window Select LAM block m This bit field determines whether the event window generation is from the monitor or reference signal. 0 _B Event window generation is determined from the reference signal. 1 _B Event window generation is determined from the monitor signal.
0	[7:5]	r	Reserved Read as 0; should be written with 0.
EDS	[11:8]	rw	Event Window Active Edge Selection LAM block m This bit field determines which active edges of the monitor and reference signals are used for the event window generation. xx00 _B Neither edge used to clear event windowcounter. xx01 _B Positive edge used to clear event windowcounter. xx10 _B Negative edge used to clear event windowcounter. xx11 _B Either edge used to clear event windowcounter. 00xx _B Neither edge used to gate event generation. 01xx _B Positive edge used to gate event generation. 10xx _B Negative edge used to gate event generation. 11xx _B Either edge used to gate event generation.
IVW	12	rw	Invert Event Window LAM block m This bit field determines whether the event window polarity is inverted or not. 0 _B Event window non-inverted. 1 _B Event window inverted.
0	[15:13]	r	Reserved Read as 0; should be written with 0.

Input Output Monitor (IOM)

Field	Bits	Type	Description
MCS	[19:16]	rw	Monitor Input Signal Selection LAM block m This bit field determines which FPC/mux block k monitor output signal is to be used for LAM block m. 0000 _B Monitor signal provided by FPC/mux block 0. 0001 _B Monitor signal provided by FPC/mux block 1. 0010 _B Monitor signal provided by FPC/mux block 2. 0011 _B Monitor signal provided by FPC/mux block 3. 0100 _B Monitor signal provided by FPC/mux block 4. 0101 _B Monitor signal provided by FPC/mux block 5. 0110 _B Monitor signal provided by FPC/mux block 6. 0111 _B Monitor signal provided by FPC/mux block 7. 1000 _B Monitor signal provided by FPC/mux block 8. 1001 _B Monitor signal provided by FPC/mux block 9. 1010 _B Monitor signal provided by FPC/mux block 10. 1011 _B Monitor signal provided by FPC/mux block 11. 1100 _B Monitor signal provided by FPC/mux block 12. 1101 _B Monitor signal provided by FPC/mux block 13. 1110 _B Monitor signal provided by FPC/mux block 14. 1111 _B Monitor signal provided by FPC/mux block 15.
RCS	[23:20]	rw	Reference Input Signal Selection LAM block m This bit field determines which FPC/mux block k reference output signal is to be used for LAM block m. 0000 _B Reference signal provided by FPC/mux block 0. 0001 _B Reference signal provided by FPC/mux block 1. 0010 _B Reference signal provided by FPC/mux block 2. 0011 _B Reference signal provided by FPC/mux block 3. 0100 _B Reference signal provided by FPC/mux block 4. 0101 _B Reference signal provided by FPC/mux block 5. 0110 _B Reference signal provided by FPC/mux block 6. 0111 _B Reference signal provided by FPC/mux block 7. 1000 _B Reference signal provided by FPC/mux block 8. 1001 _B Reference signal provided by FPC/mux block 9. 1010 _B Reference signal provided by FPC/mux block 10. 1011 _B Reference signal provided by FPC/mux block 11. 1100 _B Reference signal provided by FPC/mux block 12. 1101 _B Reference signal provided by FPC/mux block 13. 1110 _B Reference signal provided by FPC/mux block 14. 1111 _B Reference signal provided by FPC/mux block 15.
0	[31:24]	r	Reserved Read as 0; should be written with 0.

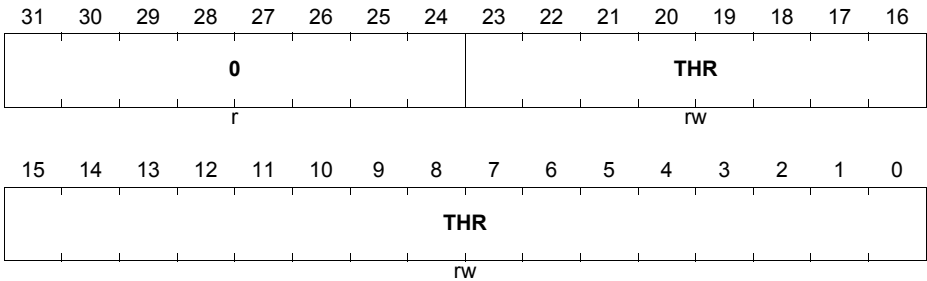
Input Output Monitor (IOM)

IOM Logic Analyzer Module Event Window Configuration Register (IOM_LAMEWSm)

Used to set the threshold value for the event window counter.

IOM_LAMEWSm (m = 0-15)

IOM Logic Analyzer Module Event Window Configuration Register_m
 (1C0_H+m*4_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
THR	[23:0]	rw	Event Window Count Threshold This bit field determines the threshold value for the event window counter from which an event is generated. Note: if THR has a value of 0 then no event will be generated.
0	[31:24]	r	Reserved Read as 0; should be written with 0.

32.9.5 Event Combiner Module (ECM) Registers

Event Combiner Module Counter Configuration register (IOM_ECMCCFG)

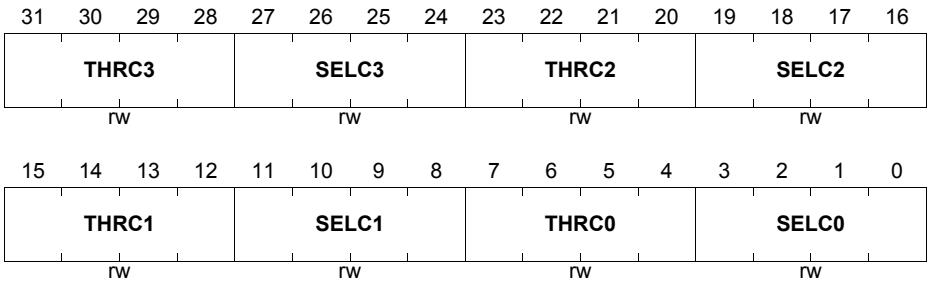
The Event Combiner Configuration register is used to configure each of the four event counter submodules, enabling multiple events from one selected LAM block (for each event counter submodule) to be counted in the generation of a system event.

IOM_ECMCCFG

IOM Event Combiner Module Counter Configuration Register

(030_H)

Reset Value: 0000 0000_H



Input Output Monitor (IOM)

Field	Bits	Type	Description
SELC0, SELC1, SELC2, SELC3	[3:0], [11:8], [19:16], [27:24]	rw	Event Channel Select This bit field determines which channel event output is to be routed to counter _{Cn} . 0000 _B Select channel 0 event output to be counted. 0001 _B Select channel 1 event output to be counted. 0010 _B Select channel 2 event output to be counted. 0011 _B Select channel 3 event output to be counted. 0100 _B Select channel 4 event output to be counted. 0101 _B Select channel 5 event output to be counted. 0110 _B Select channel 6 event output to be counted. 0111 _B Select channel 7 event output to be counted. 1000 _B Select channel 8 event output to be counted. 1001 _B Select channel 9 event output to be counted. 1010 _B Select channel 10 event output to be counted. 1011 _B Select channel 11 event output to be counted. 1100 _B Select channel 12 event output to be counted. 1101 _B Select channel 13 event output to be counted. 1110 _B Select channel 14 event output to be counted. 1111 _B Select channel 15 event output to be counted.
THRC0, THCR1, THCR2, THCR3	[7:4], [15:12], [23:20], [31:28]	rw	Channel Event Counter Threshold This bit field determines the threshold count value. Upon the counter meeting the threshold, the counter event output becomes active high for one clock cycle and the count is reset to zero. 0000 _B Counter disabled (counter event output set to inactive, despite any incoming channel events). 0001 _B Minimum threshold count value. ... 1111 _B Maximum threshold count value.

Input Output Monitor (IOM)

Event Combiner Module Global Event Selection Register (IOM_ECMSELR)

Configured to combine individual and multiple (counted) events from the LAM modules in order to generate a global (system) event.

IOM_ECMSELR

IOM Event Combiner Module Global Event Selection Register

(034_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												CTS	CTS	CTS	CTS
												3	2	1	0
r												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CES	CES	CES	CES	CES	CES	CES	CES	CES	CES	CES	CES	CES	CES	CES	CES
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CESn (n=0-15)	n	rw	<p>Event Combiner Selection</p> <p>The setting of individual bitfields determines the inclusion of the respective channel event in the generation of the global event (OR function).</p> <p>0_B Don't include channel event/event counter output within the global event generation.</p> <p>1_B Include channel event/event counter output within the global event generation.</p>
CTS_n (n=0-3)	n+16	rw	<p>Accumulated (Counted) Event Combiner Selection</p> <p>The setting of individual bitfields determines the inclusion of the respective channel event counter output (1 of 4) in the generation of the global event (AND function).</p> <p>0_B Don't include channel event/event counter output within the global event generation.</p> <p>1_B Include channel event/event counter output within the global event generation.</p>
0	[31:20]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

Input Output Monitor (IOM)
Event Combiner Module Event Trigger History Register (IOM_ECMETH0)

One of two registers (IOM_ECMETH0 and IOM_ECMETH1) that record the status of LAM's 0 to 15 event outputs for the last 4 generated system events.

The generation of a system event will cause the status of the event outputs from LAM blocks 0 to 15 to be recorded in ETA0-15, respectively. Just prior to this happening, the previous status held in ETA0-15 is moved to ETB0-15. Similarly, the previous contents of ETB0-15 and ETC0-15 are moved to ETC0-15 and ETD15-0, respectively. The old contents of ETD15-0, being lost.

At reset (or a register write to either IOM_ECMETH0 or IOM_ECMETH1), all Event Trigger Active flags (ETA, ETB, ETC, ETD) will be cleared.

IOM_ECMETH0
IOM Event Combiner Module Event Trigger History Register 0
(038_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETB	ETB	ETB	ETB	ETB	ETB	ETB	ETB	ETB	ETB	ETB	ETB	ETB	ETB	ETB	ETB
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETA	ETA	ETA	ETA	ETA	ETA	ETA	ETA	ETA	ETA	ETA	ETA	ETA	ETA	ETA	ETA
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

rwh

Field	Bits	Type	Description
ETAn (n=0-15)	n	rwh	LAM 0-15 Event Trigger Activity (last) Contains the status of the event trigger outputs for each of the LAM blocks for the last generated event(s).
ETBn (n=0-15)	n+16	rwh	LAM 0-15 Event Trigger Activity (previous ETA0-15) Contains the previous contents of ETA0-15 prior to being updated.

Input Output Monitor (IOM)

Event Combiner Module Event Trigger History Register (IOM_ECMETH1)

One of two registers (IOM_ECMETH0 and IOM_ECMETH1) that record the status of LAM's 0 to 15 event outputs for the last 4 generated system events.

The generation of a system event will cause the status of the event outputs from LAM blocks 0 to 15 to be recorded in ETA0-15, respectively. Just prior to this happening, the previous status held in ETA0-15 is moved to ETB0-15. Similarly, the previous contents of ETB0-15 and ETC0-15 are moved to ETC0-15 and ETD15-0, respectively. The old contents of ETD15-0, being lost.

At reset (or a register write to either IOM_ECMETH0 or IOM_ECMETH1), all Event Trigger Active flags (ETA, ETB, ETC, ETD) will be cleared.

IOM_ECMETH1

IOM Event Combiner Module Event Trigger History Register 1

(03C_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETD	ETD	ETD	ETD	ETD	ETD	ETD	ETD	ETD	ETD	ETD	ETD	ETD	ETD	ETD	ETD
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

rwh

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETC	ETC	ETC	ETC	ETC	ETC	ETC	ETC	ETC	ETC	ETC	ETC	ETC	ETC	ETC	ETC
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

rwh

Field	Bits	Type	Description
ETCn (n=0-15)	n	rwh	LAM 0-15 Event Trigger Activity (previous ETB0-15) Contains the previous contents of ETB0-15 prior to being updated.
ETDn (n=0-15)	n+16	rwh	LAM 0-15 Event Trigger Activity (previous ETC0-15) Contains the previous contents of ETC0-15 prior to being updated.

Input Output Monitor (IOM)

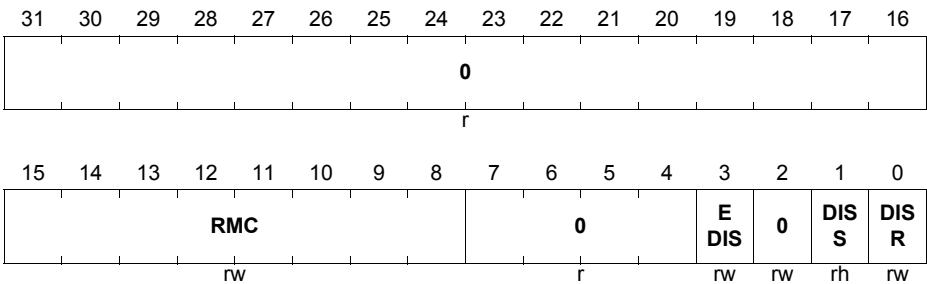
32.9.6 System Registers

IOM Clock Control Register (IOM_CLC)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI_FPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the f_{clk} module clock signal, sleep mode and disable mode for the module.

IOM_CLC

IOM Clock Control Register (000_H) Reset Value: 0000 0003_H



Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
0	2	rw	Reserved Read as 0; should be written with 0.
EDIS	3	rw	Sleep Mode Enable Control Used to control module's sleep mode.
RMC	[15:8]	rw	8-bit Clock Divider Value in RUN Mode
0	[31:16], [7:4]	r	Reserved Read as 0; should be written with 0.

Note: The number of module clock cycles (wait states) which are required by the kernel to execute a read or write access depends on the selected CLC clock frequency, which is selected via bit field RMC in the CLC register. Therefore, increasing

Input Output Monitor (IOM)

CLC.RMC may result in a longer FPI Bus read cycle access time for kernel registers and can also slow down the write throughput to the kernel registers.

IOM Access Enable Register (IOM_ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B, EN1 -> TAG ID 000001_B, ... , EN31 -> TAG ID 011111_B.

IOM_ACCEN0
IOM Access Enable Register 0 (02C_H) Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN 31	EN 30	EN 29	EN 28	EN 27	EN 26	EN 25	EN 24	EN 23	EN 22	EN 21	EN 20	EN 19	EN 18	EN 17	EN 16
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN 15	EN 14	EN 13	EN 12	EN 11	EN 10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Field	Bits	Type	Description
ENn (n = 0-31)	n	r/w	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

Input Output Monitor (IOM)

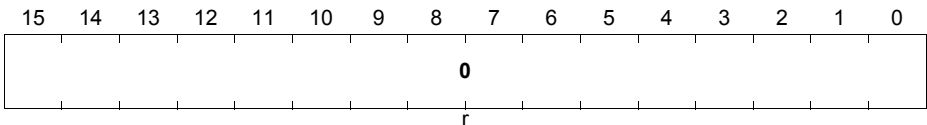
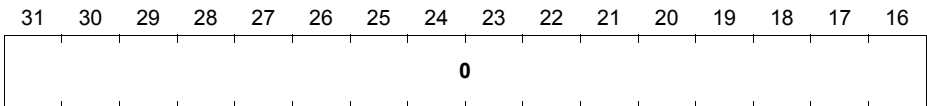
IOM Access Enable Register (IOM_ACCEN1)

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000_B, EN1 -> TAG ID 100001_B, ... , EN31 -> TAG ID 111111_B.

IOM_ACCEN1

IOM Access Enable Register 1 (028_μ) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

IOM Kernel Reset Register 0 (IOM_KRST0)

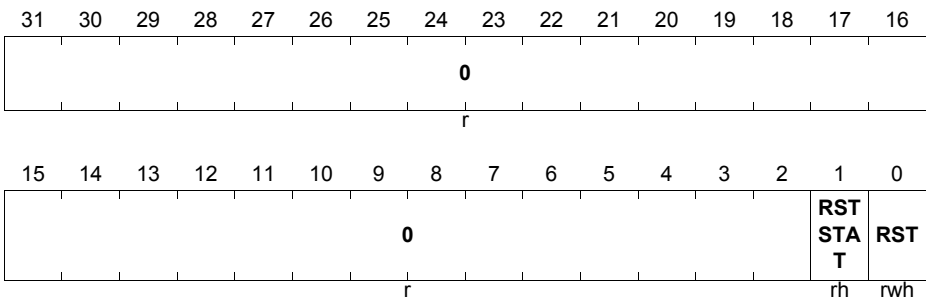
The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLE.CLR register bit.

Note: During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

IOM_KRST0

IOM Kernel Reset Register 0 (024_H) Reset Value: 0000 0000_H



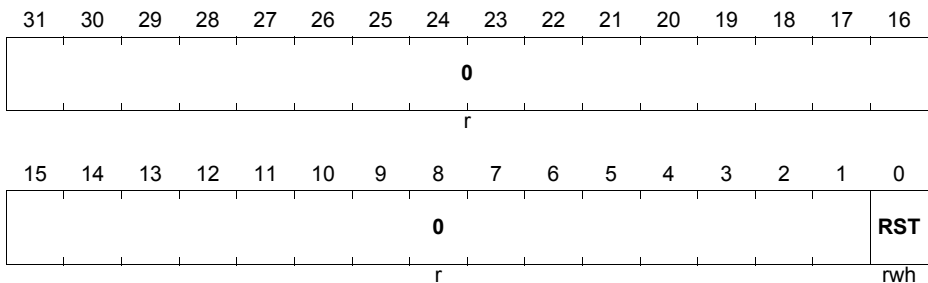
Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set.</p> <p>0_B No kernel reset was requested 1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.</p>

Input Output Monitor (IOM)

Field	Bits	Type	Description
RSTSTAT	1	rh	<p>Kernel Reset Status</p> <p>This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits.</p> <p>0_B No kernel reset was executed 1_B Kernel reset was executed</p> <p>This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.</p>
0	[31:2]	r	<p>Reserved</p> <p>Read as 0; should be written with 0.</p>

IOM Kernel Reset Register 1 (IOM_KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

IOM_KRST1
IOM Kernel Reset Register 1
(020_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Input Output Monitor (IOM)

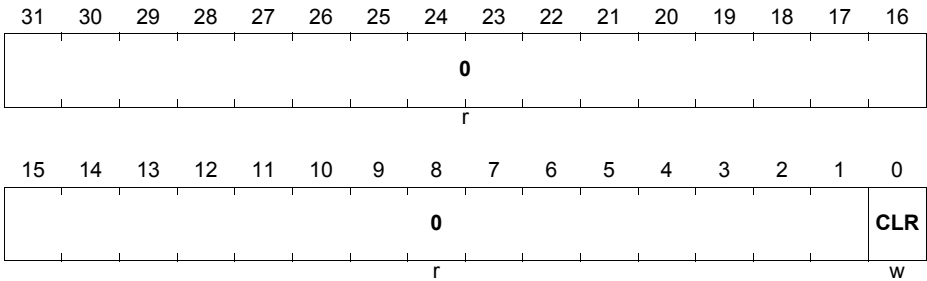
IOM Kernel Reset Status Clear Register (IOM_KRSTCLR)

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

IOM_KRSTCLR

IOM Kernel Reset Status Clear Register(01C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

32.10 SoC Integration

The following tables describes how the IOM Monitor & Reference signals are connected to the portlogic.

inst_iom port naming	GPIO Port	GPIO name	Description	Selection setting
iom_pin_i(0)	P33.0	P33_IN.P0	GPIO pad input	IOM_FPCCTR ₀ ,ISM=00
iom_pin_i(1)	P33.1	P33_IN.P1	GPIO pad input	IOM_FPCCTR ₁ ,ISM=00
iom_pin_i(2)	P33.2	P33_IN.P2	GPIO pad input	IOM_FPCCTR ₂ ,ISM=00
iom_pin_i(3)	P33.3	P33_IN.P3	GPIO pad input	IOM_FPCCTR ₃ ,ISM=00
iom_pin_i(4)	P33.4	P33_IN.P4	GPIO pad input	IOM_FPCCTR ₄ ,ISM=00
iom_pin_i(5)	P33.5	P33_IN.P5	GPIO pad input	IOM_FPCCTR ₅ ,ISM=00
iom_pin_i(6)	P33.6	P33_IN.P6	GPIO pad input	IOM_FPCCTR ₆ ,ISM=00
iom_pin_i(7)	P33.7	P33_IN.P7	GPIO pad input	IOM_FPCCTR ₇ ,ISM=00
iom_pin_i(8)	P33.8	P33_IN.P8	GPIO pad input	IOM_FPCCTR ₈ ,ISM=00
iom_pin_i(9)	P33.9	P33_IN.P9	GPIO pad input	IOM_FPCCTR ₉ ,ISM=00
iom_pin_i(10)	P33.10	P33_IN.P10	GPIO pad input	IOM_FPCCTR ₁₀ ,ISM=00
iom_pin_i(11)	P33.11	P33_IN.P11	GPIO pad input	IOM_FPCCTR ₁₁ ,ISM=00
iom_pin_i(12)	P33.12	P33_IN.P12	GPIO pad input	IOM_FPCCTR ₁₂ ,ISM=00
iom_pin_i(13)	P20.12	P20_IN.P12	GPIO pad input	IOM_FPCCTR ₁₃ ,ISM=00
iom_pin_i(14)	P20.13	P20_IN.P13	GPIO pad input	IOM_FPCCTR ₁₄ ,ISM=00
iom_pin_i(15)	P20.14	P20_IN.P14	GPIO pad input	IOM_FPCCTR ₁₅ ,ISM=00
iom_mon_0_i(0)	P33.0	ALT1	GTM output: TOUT22	IOM_FPCCTR ₀ ,ISM=01
iom_mon_0_i(1)	P33.1	ALT1	GTM output: TOUT23	IOM_FPCCTR ₁ ,ISM=01
iom_mon_0_i(2)	P33.2	ALT1	GTM output: TOUT24	IOM_FPCCTR ₂ ,ISM=01
iom_mon_0_i(3)	P33.3	ALT1	GTM output: TOUT25	IOM_FPCCTR ₃ ,ISM=01
iom_mon_0_i(4)	P33.4	ALT1	GTM output: TOUT26	IOM_FPCCTR ₄ ,ISM=01
iom_mon_0_i(5)	P33.5	ALT1	GTM output: TOUT27	IOM_FPCCTR ₅ ,ISM=01
iom_mon_0_i(6)	P33.6	ALT1	GTM output: TOUT28	IOM_FPCCTR ₆ ,ISM=01
iom_mon_0_i(7)	P33.7	ALT1	GTM output: TOUT29	IOM_FPCCTR ₇ ,ISM=01
iom_mon_0_i(8)	P33.8	ALT1	GTM output: TOUT30	IOM_FPCCTR ₈ ,ISM=01
iom_mon_0_i(9)	P33.9	ALT1	GTM output: TOUT31	IOM_FPCCTR ₉ ,ISM=01
iom_mon_0_i(10)	P33.10	ALT1	GTM output: TOUT32	IOM_FPCCTR ₁₀ ,ISM=01
iom_mon_0_i(11)	P33.11	ALT1	GTM output: TOUT33	IOM_FPCCTR ₁₁ ,ISM=01
iom_mon_0_i(12)	P33.12	ALT1	GTM output: TOUT34	IOM_FPCCTR ₁₂ ,ISM=01
iom_mon_0_i(13)	P20.12	ALT1	GTM output: TOUT68	IOM_FPCCTR ₁₃ ,ISM=01
iom_mon_0_i(14)	P20.13	ALT1	GTM output: TOUT69	IOM_FPCCTR ₁₄ ,ISM=01
iom_mon_0_i(15)	P20.14	ALT1	GTM output: TOUT70	IOM_FPCCTR ₁₅ ,ISM=01
iom_mon_1_i(0)	P15.4	ALT7	CCU60 output: CC62	IOM_FPCCTR ₀ ,ISM=10
iom_mon_1_i(1)	P15.5	ALT7	CCU60 output: CC61	IOM_FPCCTR ₁ ,ISM=10
iom_mon_1_i(2)	P15.6	ALT7	CCU60 output: CC60	IOM_FPCCTR ₂ ,ISM=10
iom_mon_1_i(3)	P15.7	ALT7	CCU60 output: COU760	IOM_FPCCTR ₃ ,ISM=10
iom_mon_1_i(4)	P15.8	ALT7	CCU60 output: COU761	IOM_FPCCTR ₄ ,ISM=10
iom_mon_1_i(5)	P14.0	ALT7	CCU60 output: COU762	IOM_FPCCTR ₅ ,ISM=10
iom_mon_1_i(6)	P14.1	ALT7	CCU60 output: COU763	IOM_FPCCTR ₆ ,ISM=10
iom_mon_1_i(7)	P20.7	ALT7	CCU61 output: COU763	IOM_FPCCTR ₇ ,ISM=10
iom_mon_1_i(8)	P20.8	ALT7	CCU61 output: CC60	IOM_FPCCTR ₈ ,ISM=10
iom_mon_1_i(9)	P20.9	ALT7	CCU61 output: CC61	IOM_FPCCTR ₉ ,ISM=10
iom_mon_1_i(10)	P20.10	ALT7	CCU61 output: CC62	IOM_FPCCTR ₁₀ ,ISM=10
iom_mon_1_i(11)	P20.11	ALT7	CCU61 output: COU760	IOM_FPCCTR ₁₁ ,ISM=10
iom_mon_1_i(12)	P20.12	ALT7	CCU61 output: COU761	IOM_FPCCTR ₁₂ ,ISM=10
iom_mon_1_i(13)	P20.13	ALT7	CCU61 output: COU762	IOM_FPCCTR ₁₃ ,ISM=10
iom_mon_1_i(14)	P02.4	ALT4	PSI5 output: PSITX0	IOM_FPCCTR ₁₄ ,ISM=10
iom_mon_1_i(15)	P02.6	ALT4	PSI5 output: PSITX1	IOM_FPCCTR ₁₅ ,ISM=10
iom_mon_2_i(0)	P20.12	ALT3	QSPI0 output: MRST0	IOM_FPCCTR ₀ ,ISM=11
iom_mon_2_i(1)	P10.1	ALT3	QSPI1 output: MRST1	IOM_FPCCTR ₁ ,ISM=11
iom_mon_2_i(2)	P15.7	ALT3	QSPI2 output: MRST2	IOM_FPCCTR ₂ ,ISM=11
iom_mon_2_i(3)	P10.7	ALT3	QSPI3 output: MRST3	IOM_FPCCTR ₃ ,ISM=11
iom_mon_2_i(4)	P22.1	ALT3	tied to '0'	IOM_FPCCTR ₄ ,ISM=11
iom_mon_2_i(5)	P02.0	ALT5	CAN node 0 output: TXDCAN0	IOM_FPCCTR ₅ ,ISM=11
iom_mon_2_i(6)	P01.3	ALT5	CAN node 1 output: TXDCAN1	IOM_FPCCTR ₆ ,ISM=11
iom_mon_2_i(7)	P02.2	ALT5	CAN node 2 output: TXDCAN2	IOM_FPCCTR ₇ ,ISM=11
iom_mon_2_i(8)	P00.2	ALT5	CAN node 3 output: TXDCAN3	IOM_FPCCTR ₈ ,ISM=11
iom_mon_2_i(9)	P10.2	ALT1	GTM output: TOUT104	IOM_FPCCTR ₉ ,ISM=11
iom_mon_2_i(10)	P10.3	ALT1	GTM output: TOUT105	IOM_FPCCTR ₁₀ ,ISM=11
iom_mon_2_i(11)	P10.4	ALT1	GTM output: TOUT106	IOM_FPCCTR ₁₁ ,ISM=11
iom_mon_2_i(12)	P14.0	ALT2	ASCLIN0 output: ATX0	IOM_FPCCTR ₁₂ ,ISM=11
iom_mon_2_i(13)	P02.2	ALT2	ASCLIN1 output: ATX1	IOM_FPCCTR ₁₃ ,ISM=11
iom_mon_2_i(14)	P02.0	ALT2	ASCLIN2 output: ATX2	IOM_FPCCTR ₁₄ ,ISM=11
iom_mon_2_i(15)	P00.1	ALT2	ASCLIN3 output: ATX3	IOM_FPCCTR ₁₅ ,ISM=11

IOM_PORTLOGIC_CONN_MON

Figure 32-13 Monitor portlogic/block signal connectivity

Input Output Monitor (IOM)

inst_iom port naming	GPIO Port	GPIO name	Description	Selection setting
iom_pin_i(0)	P33.0	P33_IN.P0	GPIO pad input	IOM_FPCCTR ₀ .ISR=000
iom_pin_i(1)	P33.1	P33_IN.P1	GPIO pad input	IOM_FPCCTR ₁ .ISR=000
iom_pin_i(2)	P33.2	P33_IN.P2	GPIO pad input	IOM_FPCCTR ₂ .ISR=000
iom_pin_i(3)	P33.3	P33_IN.P3	GPIO pad input	IOM_FPCCTR ₃ .ISR=000
iom_pin_i(4)	P33.4	P33_IN.P4	GPIO pad input	IOM_FPCCTR ₄ .ISR=000
iom_pin_i(5)	P33.5	P33_IN.P5	GPIO pad input	IOM_FPCCTR ₅ .ISR=000
iom_pin_i(6)	P33.6	P33_IN.P6	GPIO pad input	IOM_FPCCTR ₆ .ISR=000
iom_pin_i(7)	P33.7	P33_IN.P7	GPIO pad input	IOM_FPCCTR ₇ .ISR=000
iom_pin_i(8)	P33.8	P33_IN.P8	GPIO pad input	IOM_FPCCTR ₈ .ISR=000
iom_pin_i(9)	P33.9	P33_IN.P9	GPIO pad input	IOM_FPCCTR ₉ .ISR=000
iom_pin_i(10)	P33.10	P33_IN.P10	GPIO pad input	IOM_FPCCTR ₁₀ .ISR=000
iom_pin_i(11)	P33.11	P33_IN.P11	GPIO pad input	IOM_FPCCTR ₁₁ .ISR=000
iom_pin_i(12)	P33.12	P33_IN.P12	GPIO pad input	IOM_FPCCTR ₁₂ .ISR=000
iom_pin_i(13)	P20.12	P20_IN.P12	GPIO pad input	IOM_FPCCTR ₁₃ .ISR=000
iom_pin_i(14)	P20.13	P20_IN.P13	GPIO pad input	IOM_FPCCTR ₁₄ .ISR=000
iom_pin_i(15)	P20.14	P20_IN.P14	GPIO pad input	IOM_FPCCTR ₁₅ .ISR=000
iom_ref_0_i(0)	P02.0	ALT1	GTM output : TOUT0	IOM_FPCCTR ₀ .ISR=001
iom_ref_0_i(1)	P02.1	ALT1	GTM output : TOUT1	IOM_FPCCTR ₁ .ISR=001
iom_ref_0_i(2)	P02.2	ALT1	GTM output : TOUT2	IOM_FPCCTR ₂ .ISR=001
iom_ref_0_i(3)	P02.3	ALT1	GTM output : TOUT3	IOM_FPCCTR ₃ .ISR=001
iom_ref_0_i(4)	P02.4	ALT1	GTM output : TOUT4	IOM_FPCCTR ₄ .ISR=001
iom_ref_0_i(5)	P02.5	ALT1	GTM output : TOUT5	IOM_FPCCTR ₅ .ISR=001
iom_ref_0_i(6)	P02.6	ALT1	GTM output : TOUT6	IOM_FPCCTR ₆ .ISR=001
iom_ref_0_i(7)	P02.7	ALT1	GTM output : TOUT7	IOM_FPCCTR ₇ .ISR=001
iom_ref_0_i(8)	P02.8	ALT1	GTM output : TOUT8	IOM_FPCCTR ₈ .ISR=001
iom_ref_0_i(9)	P00.0	ALT1	GTM output : TOUT9	IOM_FPCCTR ₉ .ISR=001
iom_ref_0_i(10)	P00.1	ALT1	GTM output : TOUT10	IOM_FPCCTR ₁₀ .ISR=001
iom_ref_0_i(11)	P00.2	ALT1	GTM output : TOUT11	IOM_FPCCTR ₁₁ .ISR=001
iom_ref_0_i(12)	P00.3	ALT1	GTM output : TOUT12	IOM_FPCCTR ₁₂ .ISR=001
iom_ref_0_i(13)	P00.4	ALT1	GTM output : TOUT13	IOM_FPCCTR ₁₃ .ISR=001
iom_ref_0_i(14)	P00.5	ALT1	GTM output : TOUT14	IOM_FPCCTR ₁₄ .ISR=001
iom_ref_0_i(15)	P00.6	ALT1	GTM output : TOUT15	IOM_FPCCTR ₁₅ .ISR=001
iom_ref_1_i(0)	P00.0	ALT7	CCU60 output : COUT63	IOM_FPCCTR ₀ .ISR=010
iom_ref_1_i(1)	P02.5	ALT7	CCU60 output : COUT62	IOM_FPCCTR ₁ .ISR=010
iom_ref_1_i(2)	P02.3	ALT7	CCU60 output : COUT61	IOM_FPCCTR ₂ .ISR=010
iom_ref_1_i(3)	P02.1	ALT7	CCU60 output : COUT60	IOM_FPCCTR ₃ .ISR=010
iom_ref_1_i(4)	P02.4	ALT7	CCU60 output : CC62	IOM_FPCCTR ₄ .ISR=010
iom_ref_1_i(5)	P02.2	ALT7	CCU60 output : CC61	IOM_FPCCTR ₅ .ISR=010
iom_ref_1_i(6)	P02.0	ALT7	CCU60 output : CC60	IOM_FPCCTR ₆ .ISR=010
iom_ref_1_i(7)	P00.10	ALT7	CCU61 output : COUT63	IOM_FPCCTR ₇ .ISR=010
iom_ref_1_i(8)	P00.6	ALT7	CCU61 output : COUT62	IOM_FPCCTR ₈ .ISR=010
iom_ref_1_i(9)	P00.4	ALT7	CCU61 output : COUT61	IOM_FPCCTR ₉ .ISR=010
iom_ref_1_i(10)	P00.2	ALT7	CCU61 output : COUT60	IOM_FPCCTR ₁₀ .ISR=010
iom_ref_1_i(11)	P00.5	ALT7	CCU61 output : CC62	IOM_FPCCTR ₁₁ .ISR=010
iom_ref_1_i(12)	P00.3	ALT7	CCU61 output : CC61	IOM_FPCCTR ₁₂ .ISR=010
iom_ref_1_i(13)	P00.1	ALT7	CCU61 output : CC60	IOM_FPCCTR ₁₃ .ISR=010
iom_ref_1_i(14)	P02.4	ALT4	PSI5 output : PSITX0	IOM_FPCCTR ₁₄ .ISR=010
iom_ref_1_i(15)	P02.8	ALT4	PSI5 output : PSITX2	IOM_FPCCTR ₁₅ .ISR=010
iom_ref_2_i(0)	P20.12	ALT3	QSPI0 output : MRST0	IOM_FPCCTR ₀ .ISR=011
iom_ref_2_i(1)	P10.1	ALT3	QSPI1 output : MRST1	IOM_FPCCTR ₁ .ISR=011
iom_ref_2_i(2)	P15.7	ALT3	QSPI2 output : MRST2	IOM_FPCCTR ₂ .ISR=011
iom_ref_2_i(3)	P10.7	ALT3	QSPI3 output : MRST3	IOM_FPCCTR ₃ .ISR=011
iom_ref_2_i(4)	P22.1	ALT3	tied to '0'	IOM_FPCCTR ₄ .ISR=011
iom_ref_2_i(5)	P02.0	ALT5	CAN node 0 output : TXDCAN0	IOM_FPCCTR ₅ .ISR=011
iom_ref_2_i(6)	P01.3	ALT5	CAN node 1 output : TXDCAN1	IOM_FPCCTR ₆ .ISR=011
iom_ref_2_i(7)	P02.2	ALT5	CAN node 2 output : TXDCAN2	IOM_FPCCTR ₇ .ISR=011
iom_ref_2_i(8)	P00.2	ALT5	CAN node 3 output : TXDCAN3	IOM_FPCCTR ₈ .ISR=011
iom_ref_2_i(9)	P10.5	ALT1	GTM output : TOUT107	IOM_FPCCTR ₉ .ISR=011
iom_ref_2_i(10)	P10.6	ALT1	GTM output : TOUT108	IOM_FPCCTR ₁₀ .ISR=011
iom_ref_2_i(11)	P10.7	ALT1	GTM output : TOUT109	IOM_FPCCTR ₁₁ .ISR=011
iom_ref_2_i(12)	P14.0	ALT2	ASCLIN0 output : ATX0	IOM_FPCCTR ₁₂ .ISR=011
iom_ref_2_i(13)	P02.2	ALT2	ASCLIN1 output : ATX1	IOM_FPCCTR ₁₃ .ISR=011
iom_ref_2_i(14)	P02.0	ALT2	ASCLIN2 output : ATX2	IOM_FPCCTR ₁₄ .ISR=011
iom_ref_2_i(15)	P00.1	ALT2	ASCLIN3 output : ATX3	IOM_FPCCTR ₁₅ .ISR=011
-	-	-	EXOR Combiner o/p	IOM_FPCCTR _k .ISR=1xx where k = channel number (0 to 15)

IOM_PORTLOGIC_CONN_REF

Figure 32-14 Reference portlogic/block signal connectivity

Input Output Monitor (IOM)

inst_iom port naming	GPIO Port	GPIO name	Description	Inclusion setting
iom_gtm_i(0)	P33.0	ALT1	GTM output : TOUT22	IOM_GTMEXR.EN0=1 (set to 0 to exclude)
iom_gtm_i(1)	P33.1	ALT1	GTM output : TOUT23	IOM_GTMEXR.EN1=1 (set to 0 to exclude)
iom_gtm_i(2)	P33.2	ALT1	GTM output : TOUT24	IOM_GTMEXR.EN2=1 (set to 0 to exclude)
iom_gtm_i(3)	P33.3	ALT1	GTM output : TOUT25	IOM_GTMEXR.EN3=1 (set to 0 to exclude)
iom_gtm_i(4)	P33.4	ALT1	GTM output : TOUT26	IOM_GTMEXR.EN4=1 (set to 0 to exclude)
iom_gtm_i(5)	P33.5	ALT1	GTM output : TOUT27	IOM_GTMEXR.EN5=1 (set to 0 to exclude)
iom_gtm_i(6)	P33.6	ALT1	GTM output : TOUT28	IOM_GTMEXR.EN6=1 (set to 0 to exclude)
iom_gtm_i(7)	P33.7	ALT1	GTM output : TOUT29	IOM_GTMEXR.EN7=1 (set to 0 to exclude)

IOM_PORTLOGIC_CONN_EXOR

Figure 32-15 EXOR combiner portlogic/block signal connectivity

32.11 Example Monitor/Safety Measures

For each channel the following measurements shall be possible:

- Duty cycle measurement
- Set-up and/or hold time measurement
- Delay window (min,max) with respect to rising or falling edge

The following waveforms provide more examples of typical signal monitoring functionality that could be deployed using the IOM. Some typical configuration settings are also provided on some of the examples.

32.11.1 Example 1 - Pulse or duty cycle too short

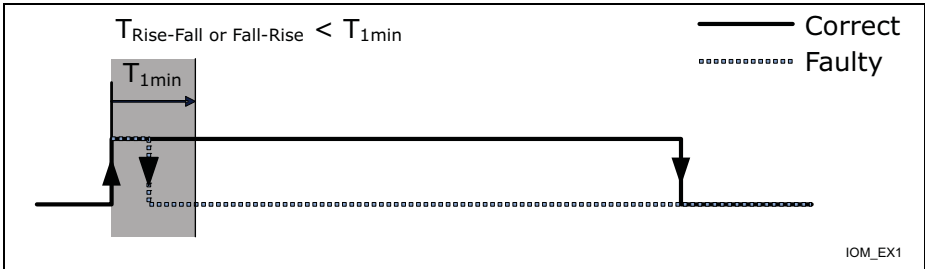


Figure 32-16 example 1 - pulse or duty cycle too short

Example settings for LAM block registers:

LAMCFG.IVR: 0x0 ; don't care

LAMCFG.IVM: 0x0 ; don't invert monitor signal (or invert for non-duty part of period)

LAMCFG.MOS: 0x0 ; monitor signal used for event generation

LAMCFG.RMS: 0x0 ; free-running

LAMCFG.EWS: 0x1 ; select monitor signal for event window

LAMCFG.EDS: 0x9 ; CLR on positive edge, EVT on negative edge

LAMCFG.IVW : 0x1 ; invert window, capture events under the threshold

LAMCFG.MCS/RCS ; select according to device configuration

LAMEWS.THR ; select appropriate threshold (minimum duty cycle length required. If duty cycle is shorter than this value then EVT will trigger)

32.11.2 Example 2 - Pulse or duty cycle too long

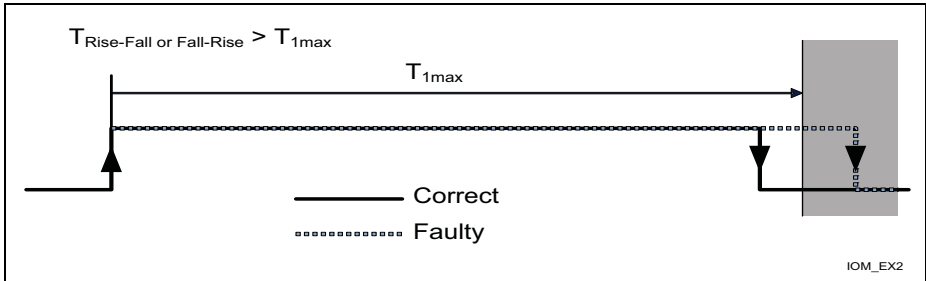


Figure 32-17 example 2 - pulse or duty cycle too long

Example settings for LAM block registers:

LAMCFG.IVR: 0x0 ; don't care

LAMCFG.IVM: 0x0 ; don't invert monitor signal (or invert for non-duty part of period)

LAMCFG.MOS: 0x0 ; monitor signal used for event generation

LAMCFG.RMS: 0x0 ; free-running

LAMCFG.EWS: 0x1 ; select monitor signal for event window

LAMCFG.EDS: 0x9 ; CLR on positive edge, EVT on negative edge

LAMCFG.IVW : 0x0 ; don't invert window, capture events beyond the threshold

LAMCFG.MCS/RCS ; select according to device configuration

LAMEWS.THR ; select appropriate threshold (maximum duty cycle length required. If duty cycle is longer than this value then EVT will trigger)

32.11.3 Example 3 - Period too short

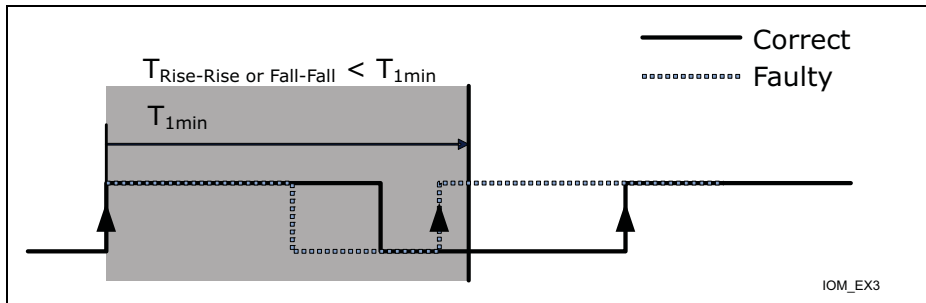


Figure 32-18 example 3 - period too short

Example settings for LAM block registers:

LAMCFG.IVR: 0x0 ; don't care

LAMCFG.IVM: 0x0 ; don't invert monitor signal

LAMCFG.MOS: 0x0 ; monitor signal used for event generation

LAMCFG.RMS: 0x0 ; free-running

LAMCFG.EWS: 0x1 ; select monitor signal for event window

LAMCFG.EDS: 0x5 ; CLR on positive edge, EVT on positive edge

LAMCFG.IVW : 0x1 ; invert window, capture events under the threshold

LAMCFG.MCS/RCS ; select according to device configuration

LAMEWS.THR ; select appropriate threshold (minimum period length required. If duty cycle is shorter than this value then EVT will trigger)

32.11.4 Example 4 - Period too long

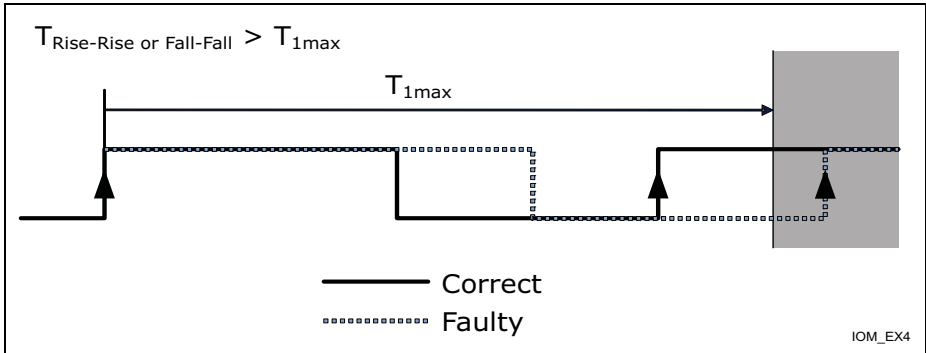


Figure 32-19 example 4 - period too long

Example settings for LAM block registers:

LAMCFG.IVR: 0x0 ; don't care

LAMCFG.IVM: 0x0 ; don't invert monitor signal

LAMCFG.MOS: 0x0 ; monitor signal used for event generation

LAMCFG.RMS: 0x0 ; free-running

LAMCFG.EWS: 0x1 ; select monitor signal for event window

LAMCFG.EDS: 0x5 ; CLR on positive edge, EVT on positive edge

LAMCFG.IVW : 0x0 ; don't invert window, capture events beyond the threshold

LAMCFG.MCS/RCS ; select according to device configuration

LAMEWS.THR ; select appropriate threshold (maximum period length required. If duty cycle is longer than this value then EVT will trigger)

32.11.5 Example 5 - Diagnosis of Command and Feedback - acceptable propagation window and/or signal consistency check

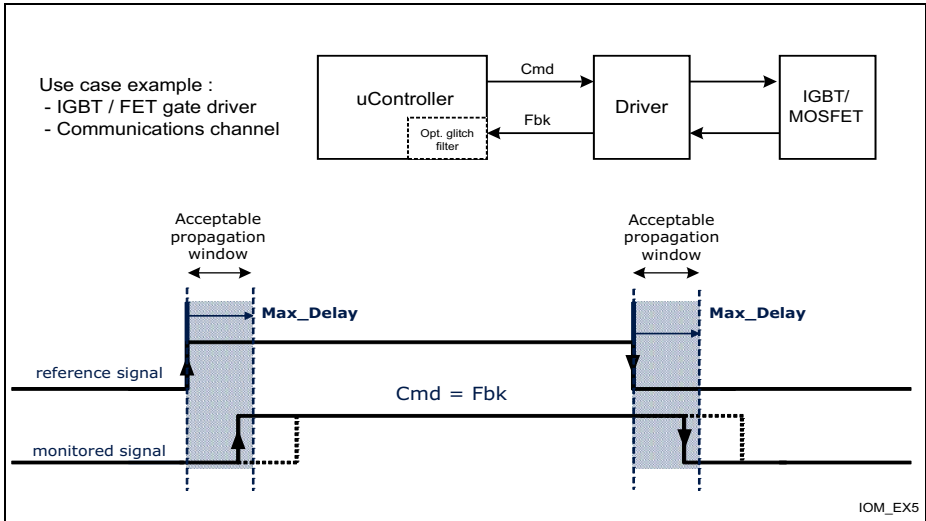


Figure 32-20 example 5 - Diagnosis of Command & Feedback

Using the driven signal (off-chip) as a reference, using a feedback signal (input to device) as the monitor. Checking that the monitored signal shows activity within an acceptable propagation window, and/or that the feedback signal shows the same behaviour as the reference (Command = Feedback).

Example settings for LAM block registers:

LAMCFG.IVR: 0x0 ; don't invert reference signal

LAMCFG.IVM: 0x0 ; don't invert monitor signal

LAMCFG.MOS: 0x1 ; (MON xor REF) signal used for event generation

LAMCFG.RMS: 0x0 ; free-running

LAMCFG.EWS: 0x0 ; select reference signal for event window

LAMCFG.EDS: 0xB ; CLR on both edges of REF, EVT on falling edge (of xor'd signal)

LAMCFG.IWW: 0x0 ; don't invert window, capture events beyond the threshold

LAMCFG.MCS/RCS ; select according to device configuration

LAMEWS.THR ; set to max delay allowed

32.11.6 Example 6 - Diagnosis of Set-up and Hold times

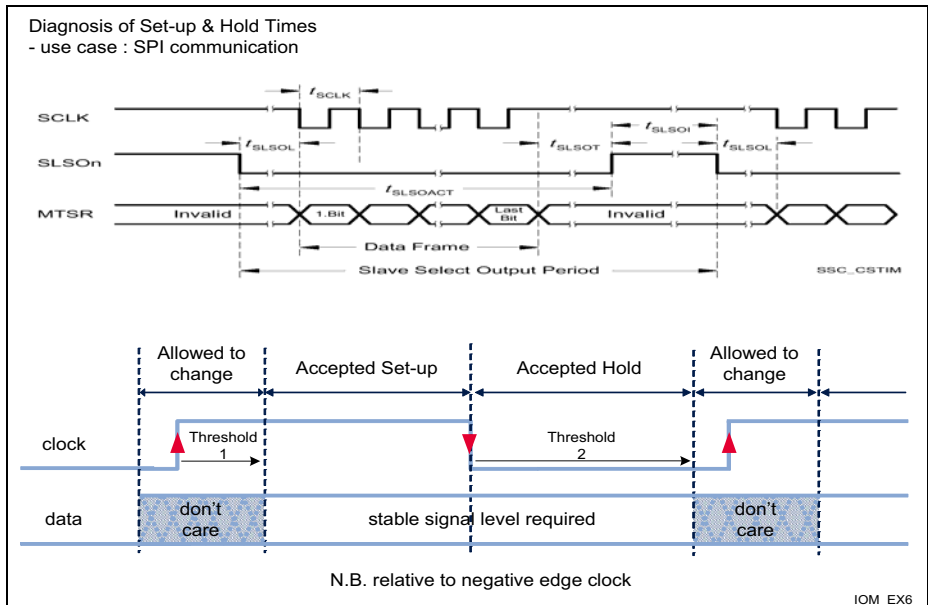


Figure 32-21 example 6 - Diagnosis of Set-up & Hold times

Ensuring that data remains stable within the Set-up and Hold windows, generating an event otherwise, and/or ensuring that data only changes within the specified windows. Note that 2 LAM channels will be required for this purpose, one for set-up, one for hold.

Example settings for LAM block registers for Set-up :

LAMCFG.IVR: 0x0 ; don't invert reference signal

LAMCFG.IVM: 0x0 ; don't invert monitor signal

LAMCFG.MOS: 0x0 ; monitor signal used for event generation

LAMCFG.RMS: 0x1 ; run is gated

LAMCFG.EWS: 0x0 ; select reference signal for event window

LAMCFG.EDS: 0xF ; CLR on both edges, EVT on both edges

LAMCFG.IVW : 0x0 ; don't invert window, capture events beyond the threshold

LAMCFG.MCS/RCS ; select according to device configuration

LAMEWS.THR ; Acceptable Set-up (ref Threshold 1 on waveforms shown)

Input Output Monitor (IOM)

Example settings for LAM block registers for Hold:

LAMCFG.IVR: 0x0 ; invert reference signal (use for gating)

LAMCFG.IVM: 0x0 ; don't invert monitor signal

LAMCFG.MOS: 0x0 ; monitor signal used for event generation

LAMCFG.RMS: 0x1 ; event window is gated

LAMCFG.EWS: 0x0 ; select reference signal for event window

LAMCFG.EDS: 0xD ; CLR on positive edge (inverted neg. edge), EVT on both edges

LAMCFG.IVW : 0x1 ; invert window, capture events below the threshold

LAMCFG.MCS/RCS ; select according to device configuration

LAMEWS.THR ; Acceptable Hold (ref Threshold 2 on waveforms shown)

33 Peripheral Sensor Interface (PSI5)

This chapter describes the PSI5 Interface of the TC27x. It contains the following sections:

- Functional description of the PSI5 kernel (see **“Overview” on Page 33-2**)
- PSI5 kernel register descriptions (see **“PSI5 Kernel Registers” on Page 33-22**)
- TC27x implementation-specific details and registers of the PSI5 module (port connections and control, interrupt control, address decoding, and clock control) see **“PSI5 Module Implementation” on Page 33-141**

33.1 PSI5 Kernel Description

Figure 33-1 shows a global view of the PSI5 interface.

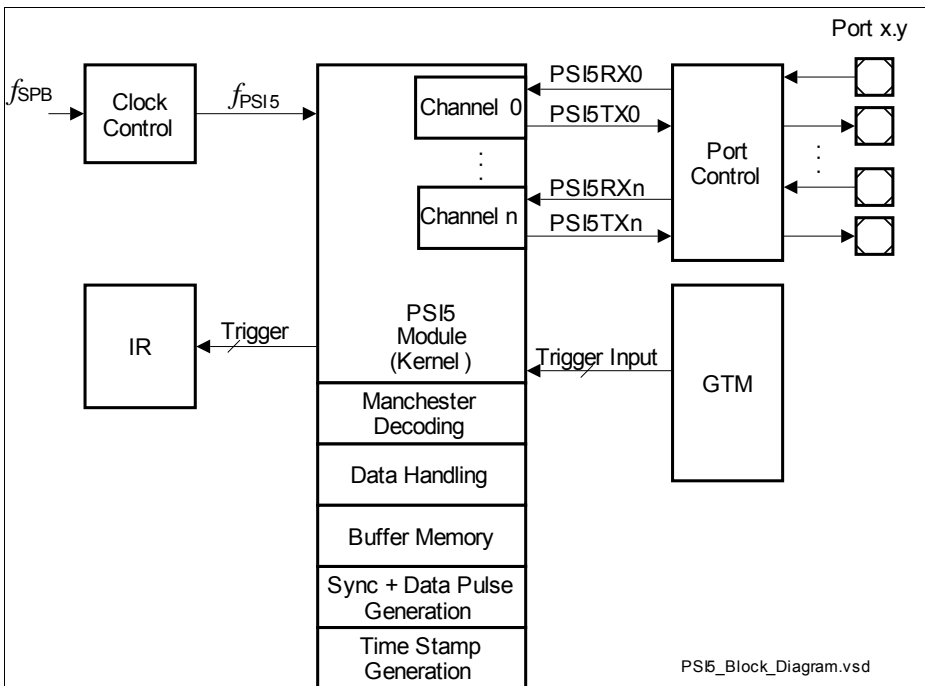


Figure 33-1 General Block Diagram of the PSI5 Interface

The PSI5 module communicates with the external world via one I/O line for each channel. The PSI5RXx lines are the receive data input signals. If the optional unidirectional mode is used, the signals PSI5TXx are transmitted on separate GPIO ports. Receive and transmit path are always routed to two different ports.

Peripheral Sensor Interface (PSI5)

33.1.1 Overview

The PSI5 IP-module performs communication according to the PSI5 specification V1.3.

The Peripheral Sensor Interface is an interface for automotive sensor applications. PSI5 is an open standard based on existing sensor interfaces for peripheral airbag sensors, already proven in millions of airbag systems.

This module supports many additional features as discussed today in the standard organization.

The PSI5 interface provides a current loop based serial communication link typically used to connect airbag sensors or other peripheral devices.

While the physical layer is done externally, this module manages protocol handling and data representation to the application. Note that there is no on chip phy, i.e. the current to voltage and voltage to sync pulse translation is done externally.

Receive data on a PSI5 channel can be set up according to the underlying application. In particular the number of bits forming one value is configured.

The message storage consists of a 32-bit buffer register for each channel and an additional 32-bit register containing the 24-bit time stamp and additional status bits. These 64 bits per frame are additionally stored in a buffer of 32 lines per channel.

In Host to sensor communication mode, the module can provide regular sync pulses and select between internal and external timer/trigger resources. It supports as well CPU to sensor communication, e.g. for setting up the sensor and retrieving sensor status information.

The register set of the PSI5 module can be accessed directly by the CPU for configuration, data read out and status query.

The PSI5 module supports the following features:

Features

- Conformance with PSI5 protocol specification V1.3
- Data rates of 125 kbit/s and 189 kbit/s supported
- 2 PSI5 channels working independently in parallel
- Supports 6 sensor slots per channel
- Asynchronous and synchronous data transmission modes (control by micro controller in synchronous mode)
- Decoding manchester protocol
- Error recognition in manchester code
- Configurable data word length 8, 10, 16, 20, 24 bit according to standard (PSI5 V1.3)
- Support of non standard V1.3 frame length: 11... 33 bit
- CRC check of received sensor data implemented but CRC code transparent
- Support of extended serial data messaging according to SENT SAE J2716 JAN 2010

Peripheral Sensor Interface (PSI5)

- 24-Bit time stamp (resolution: 1µs)
- Storage of up to 32 frames per channel with time stamp
- FIFO access and management
- Buffer overrun detection
- Buffer Memory Status Overview Registers
- Support of ECU to Sensor communication
- Three 64-bit downstream data registers for data input, data preparation and data output
- Downstream data transmission by variation of pulse length (2.0)
- Downstream data transmission by leaving out sync pulses (V1.3)
- Staggering Sync Pulses of all channels to avoid too many channels sending sync pulses in parallel
- Generation of 3 or 6 bit CRC for downstream data
- Start sequence and CRC generator for downstream data
- One sum interrupt for general events
- One sum interrupt on selectable errors
- Sticky interrupt flags, error interrupt optional (default disabled)
- Optional output inversion for use of external open drain transistor
- Optional input inversion for use of external open transistor for level shifting

Configurability

- Number of PSI5 channels working independently in parallel (1 - 5)
- Storage of up to 32 frames per channel with time stamp (w/wo)
 - FIFO access and management
 - Buffer overrun detection
 - Buffer Memory Status Overview, Set and Clear Registers

33.2 General Operation

PSI5 is intended for use in applications where high resolution sensor data needs to be communicated from a sensor to an Engine Control Unit (ECU). It is intended as a replacement for the lower resolution methods of 10 bit A/D converters and PWM and as a simpler low cost alternative to CAN or LIN. The implementation assumes that the sensor is a smart sensor containing a microprocessor or dedicated logic device (ASIC) to create the signal.

Figure 33-2 shows a typical TC27x application in which a PSI5 interface reads a sensor device.

Peripheral Sensor Interface (PSI5)

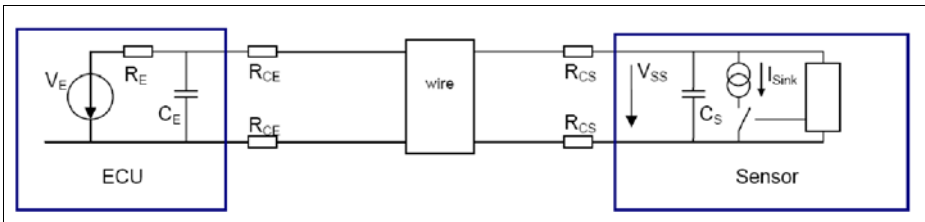


Figure 33-2 PSI5 to External Device Connection

PSI5 communication is

- asynchronous, unidirectional from sensor to controller without any synchronization
- synchronous, bidirectional with a sync pulse from the ECU triggering messages
- synchronous, bidirectional, sync pulses additionally coding data from ECU to sensor

The sensor signal is transmitted as a series of current pulses in Manchester coding.

The sync pulses are transmitted by increasing the voltage.

Figure 33-3 shows a typical TC27x application in which a PSI5 ECU reads multiple PSI5 sensor devices on a bus.

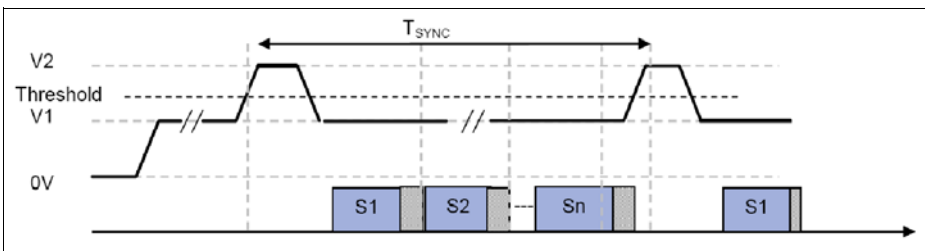


Figure 33-3 PSI5 Frames with Sync pulses

33.3 Definitions

SENT: Single Edge Nibble Transmission

Nibble: Four Bit value between 0 and 15 = half a Byte = one character in hex (0 to F)

ASIC: Application Specific Integrated Circuit

CAN: Controller Area Network

LIN: Local Interconnect Network

ISO: International Organization for Standardization

ECU: Electronic Control Unit

FSM: Finite State Machine

SOF: Start of Frame

Peripheral Sensor Interface (PSI5)

EOF: End of Frame

STS: Start Sequence

AD: Address Data (e.g. of ECU to Sensor Frame)

SD: Send Data (Payload of ECU to Sensor Frame)

CRC: Cyclic Redundancy Check

33.4 PSI5 Operation

The Module supports two modes:

Standard PSI5 Mode supports communication fully compliant to V1.3

Extended PSI5 Mode supporting additional bit fields and longer frames. (See below)

In standard V1.3 mode, a specific start condition and a start sequence switch the sensor in ECU to Sensor communication mode. In this mode, the sync pulses are no longer used as triggers for data frames but are interpreted as data. A pulse represents a 1 and a missing pulse represents a 0. This requires isochronous pulses.

In extended Mode (Power Train sub standard), the pulse width is modulated. A standard pulse represents a 0, a longer pulse a 1. This way continuous data transmission can be achieved.

33.5 Frame Formats and Definitions

This section describes the frame formats and definitions of the PSI5 protocol.

33.5.1 PSI5 V1.3 Frame

Figure 33-4 shows the layout and definitions of a standard PSI5 frame. Note that the PSI5 standard specifies that the least significant bit is sent out first. See standard for CRC and Parity definition.

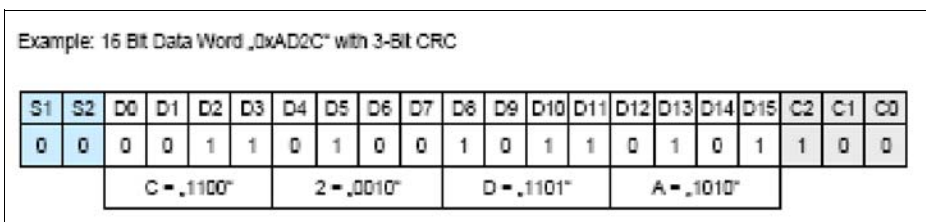


Figure 33-4 Standard PSI5 Frame

33.5.2 Extended PSI5 Frame (non standard)

Figure 33-5 shows the layout and definitions of an extended PSI5 frame as defined in V2.0 for automotive power train application. Note that the PSI5 standard specifies that the least significant bit is sent out first.

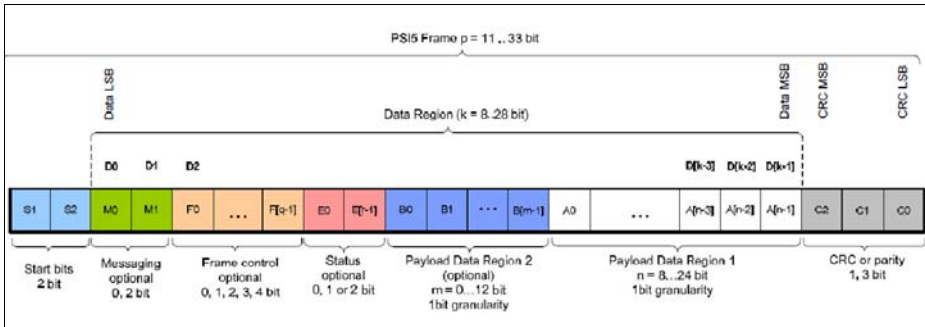


Figure 33-5 Extended PSI5 Frame

Enhanced PSI5 Sensor to ECU Message CRC

The transmission of PSI5 data is checked by the following protection modes. The transmission error detection must be selectable:

- 1) 1-bit even parity
- 2) 3-bit CRC

The applied generator polynomial of the CRC is $g(x) = 1 + x + x^3$ with a binary start value (seed) "111". The transmitter shall extend the data bits by three zeros (as MSBs). This augmented data word shall be fed (LSB first) into the shift registers of the CRC check. Start bits shall be ignored in this check. When the last zero of the augmentation is pending on the input adder, the shift registers contain the CRC checksum. These three check bits shall be transmitted in reverse order (MSB first: C2, C1, C0)

In this implementation the number of elements is configurable and thus longer or shorter than 8 .. 28 bits!

33.5.3 Extended Serial Data Encoding (“Slow Channel”)

The serial data is transmitted bit wise per frame in bits [1:0] of the Messaging bit field of 18 consecutive messages from the transmitter.

33.5.4 Extended Serial Data Frame

Serial data (“Slow Channel”) is communicated in a 18-bit sequence as shown in **Figure 33-6**. The frame start of a serial message is indicated by the unique pattern “01111110” in bit M1. The first “1” in a series of six ones (after a “0”) indicates the first set of a Messaging bits (M1 and M0) of a serial frame. Messaging bit M1 of serial data frames 1 - 6 are set to “1”. Messaging bit M1 of serial communication nibbles 7, 13 and 18 are set to “0”. 18 consecutive PSI5 messages must be successfully transmitted (no errors) for the serial value to be received. The CRC generation is different from the CRC generation on the data nibbles. (See SENT Standard SAE_J2716_201001)

The serial message frame contains 21 bits of message data and a 6-bit frame-check sequence. Two different configurations can be chosen:

- 12-bit data and 8-bit message ID
- 16-bit data and 4-bit message ID

A configuration bit (serial data bit M1, serial communication Messaging bits of frame No. 8) determines the configuration of the enhanced serial message frame. It determines how the PSI5 module automatically interprets the serial data.

- Configuration bit = 0: 12-bit data and 8-bit message ID
- Configuration bit = 1: 16-bit data and 4-bit message ID

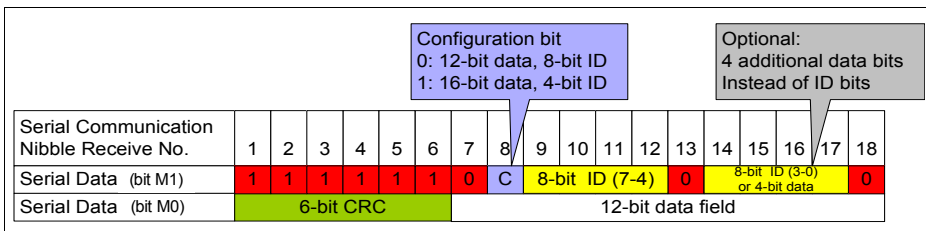


Figure 33-6 Extended Serial Data Frame

Figure 33-7 shows in detail the frame structure for Extended Serial Data Frames with Configuration Bit = 0.

Peripheral Sensor Interface (PSI5)

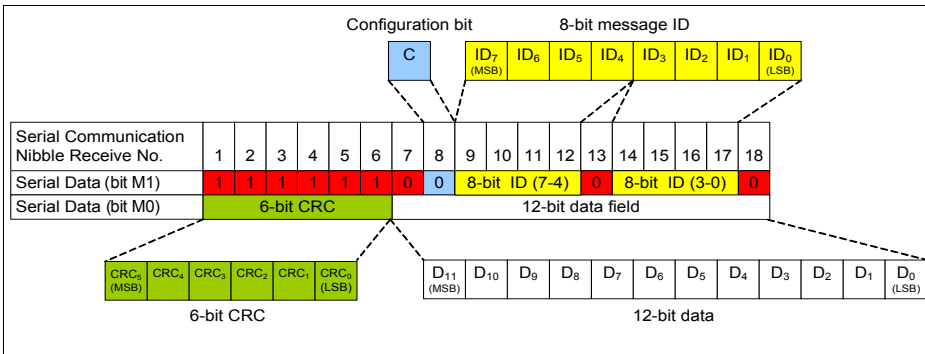


Figure 33-7 Configuration Bit = 0

Figure 33-8 shows in detail the frame structure for Extended Serial Data Frames with Configuration Bit = 1.

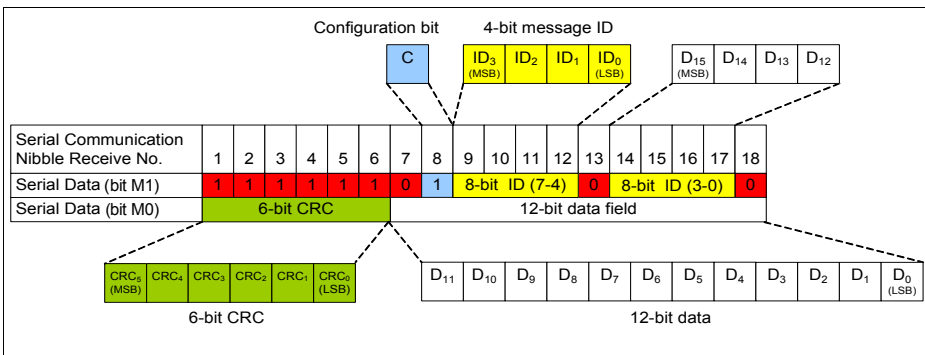


Figure 33-8 Configuration Bit = 1

33.6 Sync Pulses

Sync Pulses are used for two different purposes:

- Triggering a data frame for data acquisition from a sensor or
- ECU to sensor communication.

33.6.1 Synchronous Transmission

In the “synchronous” mode, the sensor (slave) starts to transfer a complete data frame only after a sync pulse is forced by the master on the OUT pin. An external phy translates this pulse into a voltage increase. The sensor then initiates a measurement and starts to calculate the new output data value. The data follows in a standard PSI5 frame, starting

Peripheral Sensor Interface (PSI5)

with the 2 start bits, data and Parity or CRC . The timing diagram in [Figure 33-9](#) visualizes a synchronous transmission

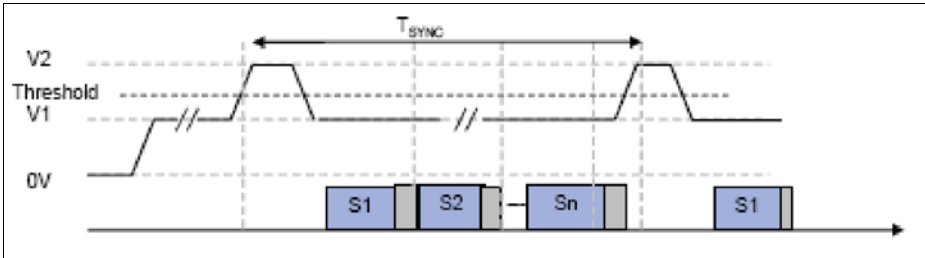


Figure 33-9 Synchronous Transmission

33.6.2 ECU to Sensor Communication

Two versions are supported: legacy PSI5 V1.3 (see [Figure 33-10](#)) and revised version (see [Figure 33-11](#) and [Figure 33-12](#)) .

The legacy version

uses sync pulses as start condition and start bits to signal the start of the ECU to Sensor Communication. From this start on, the sync pulses are interpreted by the sensor as data. See standard for CRC and Parity definition.

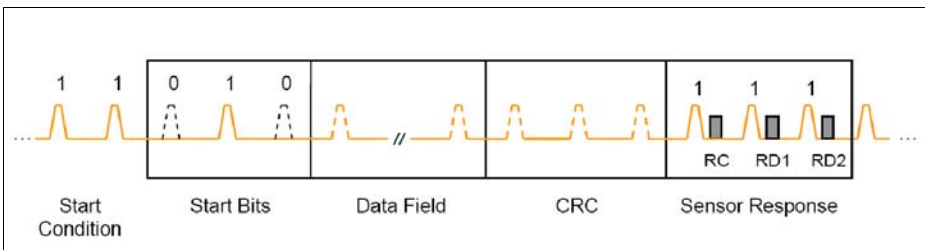


Figure 33-10 ECU to Sensor (legacy)

The revised version

allows for modulation of the sync pulse duration. This way bidirectional communication is possible continuously. Every ECU-to-Sensor frame starts with a nine-bit frame-start-sequence [0 1 1 1 1 1 1 0]. Synchronization is possible at the beginning of each frame.

Sync Bits

A consecutive train of seven ones within the transmitted data could be misinterpreted as a frame start condition. Therefore a “sync bit” (logical 0) is inserted at every seventh bit position (after every six bits of the sensor address and the payload data). With respect

Peripheral Sensor Interface (PSI5)

to the data transmission, these sync bits are treated like stuffing bits, which are removed at the receiver side.

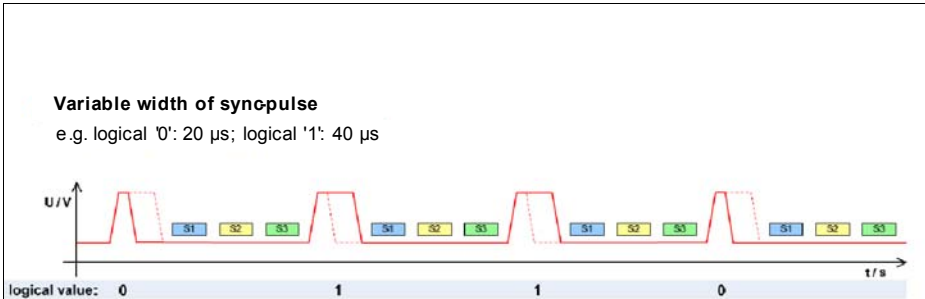


Figure 33-11 ECU to Sensor, Bit transmission (revised)

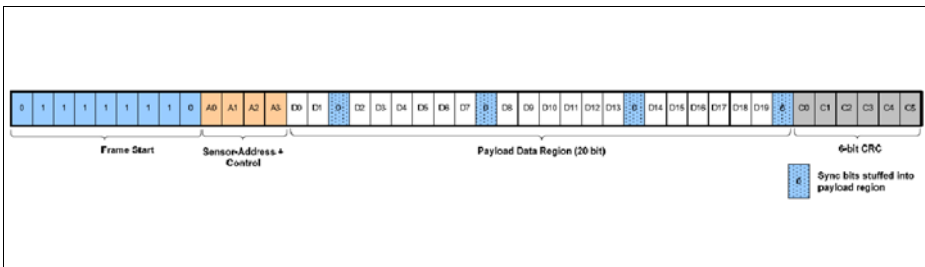


Figure 33-12 ECU to Sensor, Frame format (revised)

Enhanced ECU to Sensor Message CRC

The ECU-to-Sensor Frame has a 6-bit CRC.

The generator polynomial of the CRC is

$$g(x)=x^6 + x^4 +x^3 +1$$

with a binary CRC initialization value "010101".

The transmitter extends the data bits by six zeros (as MSBs).

This augmented data word shall be fed (LSB first) into the shift registers of the CRC check. Start bits and sync bits are ignored in this check.

When the last zero of the augmentation is pending on the input adder, the shift registers contain the CRC checksum.

These six check bits shall be transmitted LSB first [C0, C1 .. C5].

In this implementation the number of elements is configurable and thus longer or shorter than 24 bits (standard is 4 address and 20 data bits)!

33.7 Manchester Decoding

Decoding of the Manchester protocol:

- The Manchester decoder uses a nominal 4 MHz / 6.048 MHz sampling clock (32 samples per bit).
- A sampling counter, running at the sampling clock rate controls the sampling of the incoming Manchester encoded data.
- The Manchester signal transition occurring in the middle of the bit time resets the counter, making the counter count from 0-31 (nominally) between successive mid-bit times.
- Detection of the mid-bit transitions uses three consecutive samples (011 = rising edge, 100 = falling edge).
- The maximum value of the 6 bit counter shall be 41, at which point a Manchester error is detected due to lack of a transition, and the counter rolls over to 0. ("Idle Time")
- The value of a Manchester encoded input data bit is determined by proper logic levels before and after a transition and proper timing of the transition.
- Bit-value determination nominally takes place at the 1/4 and 3/4 bit times, with the mid-bit transition occurring at the 1/2 bit time. This is nominally at sampling counter values of 8 (1/4 bit time) and 24 (3/4 bit time).
- Logic levels are determined using a 2 of 3 majority vote of consecutive samples (samples 23, 24, 25 and 7, 8, 9)
 - The combinations 111, 011, 101, 110 indicate a 1
 - The combinations 000, 001, 010, 100 indicate a 0

Manchester-2 is used by standard PSI5 V1.3:

- Start bits = '00'
- Falling edge corresponds to logical '1'
- Rising edge corresponds to logical '0'
- Least significant bit (LSB) is sent out first

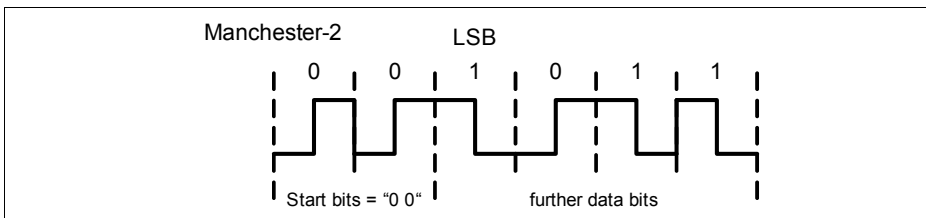


Figure 33-13 Manchester-2

Sampling Details

- Recognition of Frame Start
 - Manchester Code requires a deterministic Start of Frame
 - Frame Separation: An "Idle Time" of at least one Bit time is required

Peripheral Sensor Interface (PSI5)

- “Idle Time” is the time without edges on the line that is required to safely determine that no frame is being transceived (1 Bit time + tolerance = > 41 samples)
 The Idle Counter starts counting
 - on a falling edge, after sync pulse or after BOT, if the channel is switched on
 - if the line is low
 - up to 41 and stops and
 - activates the idle flag
 - until the line shows a rising edge, this clears the idle counter
- After this idle time, a determined start bit sequence is required for synchronization of the phase of receiver and transmitter ('0 0' in case of the „Manchester 2“-code employed)
- If no idle time occurs at the end of a frame, bits after the configured length are dropped.
- If a 2nd frame comes too early (collision = no idle time) the 2nd frame is dropped. The decoder waits for an “Idle Time” for re synchronization.
- Idle time is particularly required in case of Manchester Errors and frame collisions for a save re synchronization. Otherwise, two successive zeros in a damaged frame / a second colliding frame could lead to reception of wrong data. Note that back to back transfers of correct frames are forbidden by standard (T_gap required).
- This idle time is required as well after a regular End Of Frame, Sync Pulse or a Blank Out Time to avoid inadvertent synchronization inside frames under transmission.
- Bit sampling
 - After the start of the protocol is detected and an edge on the line signal is detected and the counter is greater 25 the counter is set to '0'
 - If the counter is 41 the counter is set to '0' and stops
 - The counter will be incremented each clock cycle
 - When the sample counter has the value 9, the level is stored in register cnt9_val
 - When the sample counter has the value 25, the level is stored in register cnt25_val
 - This register is reset if no frame is detected. Otherwise a wrong start bit detection could occur.
- Bit evaluation
 - When the sample counter value is 16, the decision is done for the new sample value:
 - Check if the last edge was detected correctly i.e.: edge detected between sample counter value 25 and sample counter value 41
 - If true:
 - Cnt_25_lev = 0 & cnt_9_lev = 1 => new sample = '0'
 - Cnt_25_lev = 1 & cnt_9_lev = 0 => new sample = '1'
 - If the edge check fails or cnt_25_lev = cnt_9_lev, Manchester Error is assumed.

Peripheral Sensor Interface (PSI5)

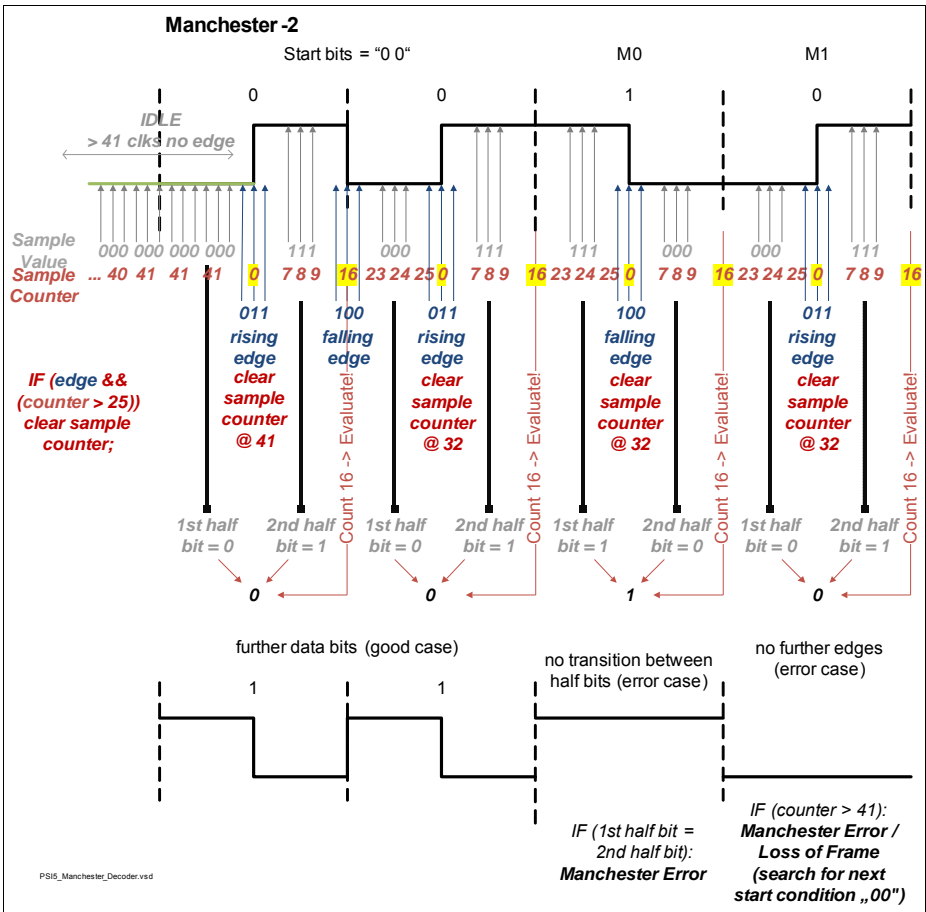


Figure 33-14 Manchester Decoder

Definition of Manchester Error (ME)

- Two half bits have the same value => ME, the decoder waits for an "Idle Time"
- > 41 sample times no edge => ME
- The ME is no dedicated interrupt but will lead to various error responses depending on where it occurs. See [Table 33-1](#).

Table 33-1 Error Responses due to ME

Manchester Error (ME) Case	Response	Comment
First ME in start bits	Discard frame	NFI (if FEC is set) and empty frame stored (if FEC and VBS is set), Time Stamp = end of slot where SOF happened. This is true even if these start bits should start before WDL and end after WDL
Start bits cut off by BOT or Sync Pulse (frame starts without leading idle time)	Discard frame	NFI (if FEC is set) and empty frame stored (if FEC and VBS is set)
First ME in Messaging bits	If MSG is set: MEI	Frame is stored, bits after error are lost ("0")
	If MSG cleared: NBI	
First ME in data, parity or CRC / Frame ends too early	NBI	Frame is stored, bits after error are lost ("0")
Frame ends too late (longer than PDL)	no NBI	Bits after PDL are cut off and EOF is detected. Frame is stored.
Collision (Idle Time missing)	Discard 2nd frame	If the 1st frame was too late and running over WDL or If the 2nd frame was too early and running over WDL then for 2nd frame, NFI (if FEC is set) and empty frame stored (if FEC and VBS is set), Time Stamp = end of 2nd slot

In all cases, the receiver waits for an idle time followed by the start bits "00" before a new frame is detected.

33.8 Bit Rate Generation

The bit rate of each channel is individually selectable from the 2 standard bit rates 125 kHz and 189 kHz. The two referring over sampling frequencies are generated centrally.

This chapter shows in detail how the bit rate for each channel is adjusted.

Peripheral Sensor Interface (PSI5)

In the first stage, a global fractional divider serves as pre divider. Its intermediate frequency $f_{fracdiv}$ can be set up so that it is handy as input frequency for the different PSI5 channels. Usually this can be left unused.

For each of the two bit rates there is an own fractional divider. Each channel can select one out of the two clock signals f_{125} and f_{189} .

For details on the principles of a fractional divider, please refer to the SCU chapter of TC27x section "Clock Control".

The PSI5 module provides 5 clock signals centrally (Figure 33-15):

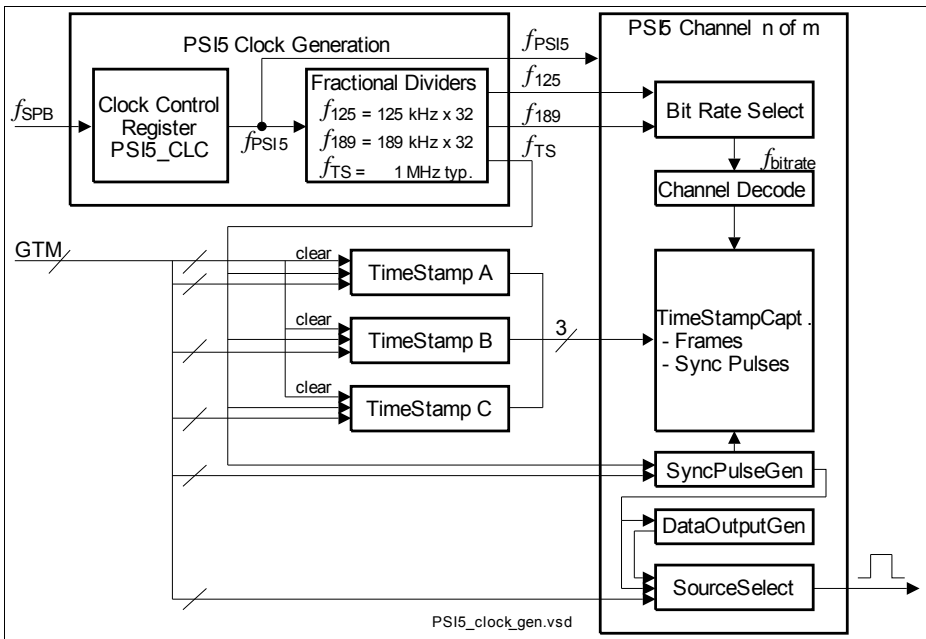


Figure 33-15 PSI5 Module Clock Generation

- f_{PSI5}

This is the module clock that is used inside the PSI5 kernel for control purposes such as clocking of control logic and register operations. The frequency of f_{PSI5} is always identical to the system clock frequency f_{SPB} . The clock control register PSI5_CLC makes it possible to enable/disable f_{PSI5} under certain conditions.
- $f_{fracdiv}$

This clock is the module clock that is used inside the PSI5 kernel for bit rate generation of the serial channels. The fractional divider register PSI5_FDR controls the frequency of $f_{fracdiv}$. Usually not required and set to 1.

Peripheral Sensor Interface (PSI5)

- f_{125}
This clock is the over sampling clock that can be selected for use inside a PSI5 channel as the bit rate frequency. The fractional divider register PSI5_FDRL controls the frequency of f_{125} . The clock drives the manchester decoder and needs to be 32 times faster than the nominal bus frequency
- f_{189}
This clock is the over sampling clock that can be selected for use inside a PSI5 channel as the bit rate frequency. The fractional divider register PSI5_FDRH controls the frequency of f_{189} . The clock drives the manchester decoder and needs to be 32 times faster than the nominal bus frequency.
- f_{TS}
This clock is used to drive the central time stamp counter. It can be selected to drive the global sync pulse time base and inside the PSI5 channels to drive the local watch dog timers. The fractional divider register PSI5_FDRT controls the frequency of f_{TS} . This is usually adjusted to 1 MHz / 1 μ s period time.

The following two formulas define the frequency of f_{fracdiv} :

$$f_{\text{fracdiv}} = f_{\text{PSI5}} / (1024 - \text{PSI5_FDR.STEP}); \text{FDR.DM} = 01\text{B} \quad (33.1)$$

$$f_{\text{fracdiv}} = f_{\text{PSI5}} \times \text{PSI5_FDR.STEP} / 1024 \text{ with STEP} = 0 \dots 1023; \text{FDR.DM} = 10\text{B} \quad (33.2)$$

The following two formulas define the frequency of f_{125} .

$$f_{125} = f_{\text{fracdiv}} / (1024 - \text{PSI5_FDRL.STEP}); \text{FDRL.DM} = 01\text{B} \quad (33.3)$$

$$f_{125} = f_{\text{fracdiv}} \times \text{PSI5_FDRL.STEP} / 1024 \text{ with STEP} = 0 \dots 1023; \text{FDRL.DM} = 10\text{B} \quad (33.4)$$

The following two formulas define the frequency of f_{189} .

$$f_{189} = f_{\text{fracdiv}} / (1024 - \text{PSI5_FDRH.STEP}); \text{FDRH.DM} = 01\text{B} \quad (33.5)$$

$$f_{189} = f_{\text{fracdiv}} \times \text{PSI5_FDRH.STEP} / 1024 \text{ with STEP} = 0 \dots 1023; \text{FDRH.DM} = 10\text{B} \quad (33.6)$$

The following two formulas define the frequency of f_{TS} :

$$f_{TS} = f_{\text{PSI5}} / (1024 - \text{PSI5_FDRT.STEP}); \text{FDRT.DM} = 01\text{B} \quad (33.7)$$

$$f_{TS} = f_{\text{PSI5}} \times \text{PSI5_FDRT.STEP} / 1024 \text{ with STEP} = 0 \dots 1023; \text{FDRT.DM} = 10\text{B} \quad (33.8)$$

33.9 Digital Glitch Filter

Very slow slopes and signal noise can lead to fast transitions of the input signal. These unwanted transitions are suppressed by a digital glitch filter similar to a Filter and Prescaler Cell (FPC) of Infineons GPTA® in Delayed De-bounce Filter Mode with up and down (no reset).

It is built for filtering very fast transitions only. The filter calculates the integral of the signal. If the integral reaches a programmable saturation point, the signal change is notified to the pulse measurement unit. Thus it helps to find the exact pulse length for said slow slopes in noisy environment.

The glitch filter is clocked with f_{PSI5} . If the state of the input sample differs from the current output signal value, the internal counter is incremented by one. When the state of the input sample matches the current output signal value and the timer is not in idle, the timer is decremented by one. When the timer matches the compare value stored in IOCRx.DEPTH, the level of the output signal line is inverted. The timer will be reset and set to idle state again.

The depth of the filter can be programmed to a value between 1 and 15. Typically a depth of 3 to 5 T_{PSI5} is sufficient. By default it is cleared. If IOCRx.DEPTH is cleared the filter is inactive/bypassed. Nevertheless, the input signal is sampled with f_{PSI5} .

The internal signal after the filter will change value not before the new value was sampled DEPTH times. If during this period a spike occurs, it takes $2 \times T_{PSI5}$ times longer for the internal signal to change value. The filter's principal implies a delay of the signal by $(DEPTH \times T_{PSI5})$.

Upon detection of glitch during rising or falling edge, IOCRx.REG or IOCRx.FEG is set. The rising / falling edge glitch flags must be reset by software.

Figure 33-16 shows the digital glitch filter:

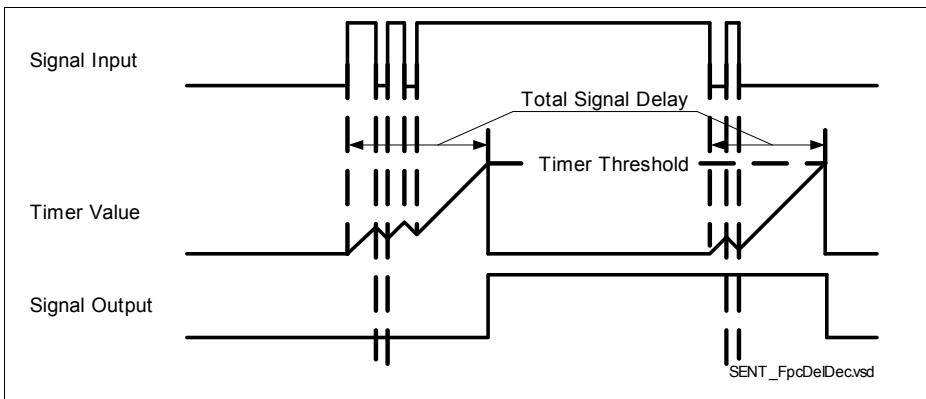


Figure 33-16 Digital Glitch Filter

33.10 Time Stamp Generation

Three time bases can be selected for Time Stamping:

Any of TSRA, TSRB and TSRC provide one time stamp counter. Each of them can be configured to be driven by either f_{TS} or by an external timer, i.e. one of the GTM inputs. The frequency of f_{TS} is controllable by the fractional divider in register FDRT.

All of them can be cleared by a selectable GTM signal or by SW, single or synchronously.

Two times are captured for each channel:

The last sync pulse sent on this channel is stored in the sync pulse time stamp capture register SPTSCx.

The last start of frame (SOF) on this channel is stored in the SOF capture register SFTSCx.

For each channel the capture time (SPTSC or SFTSC) can be selected, that will be stored in the Receive Data Register and the Receive Data Memory.

Figure 33-17 shows the time stamp generation:

Peripheral Sensor Interface (PSI5)

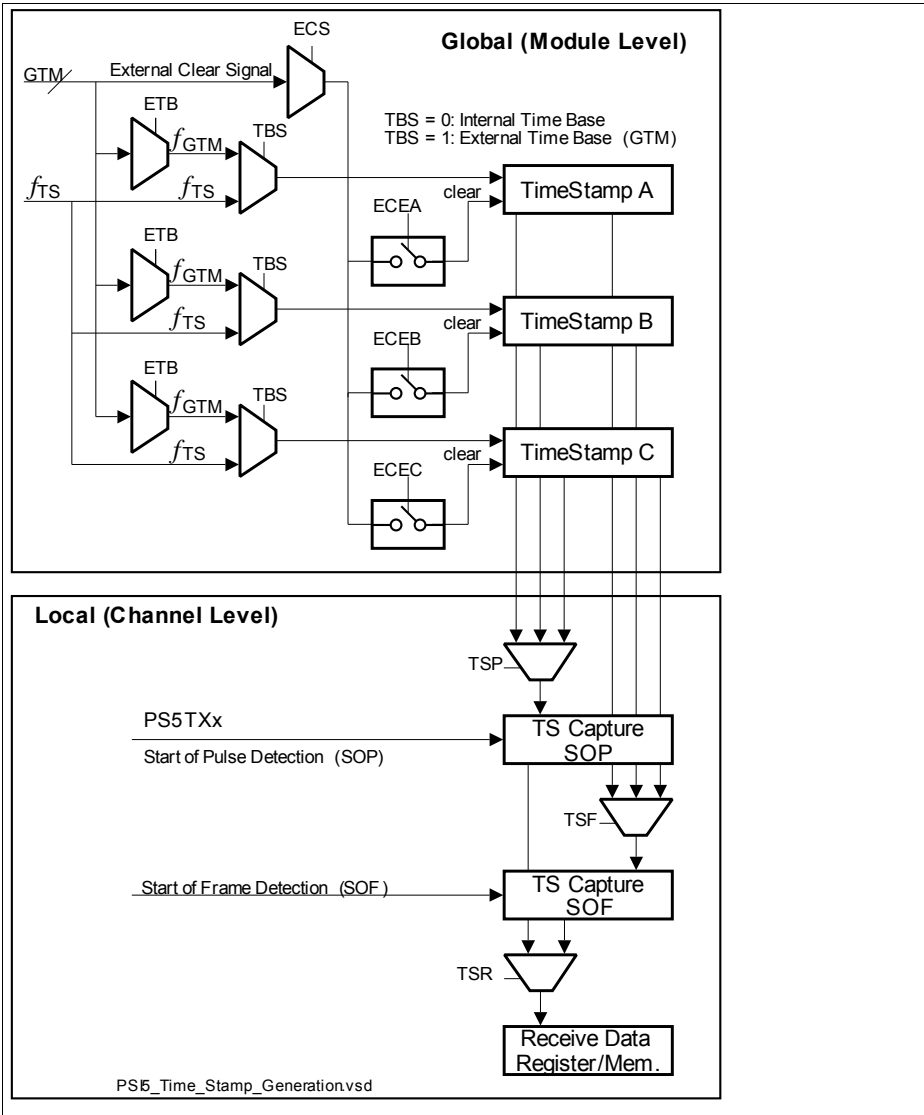


Figure 33-17 Time Stamp Generation

33.11 Error Detection Capabilities

Each PSI5 channel can detect and signal the following error conditions:

Protocol Level:

- No frame received / Manchester coding error in start bits (NFI)
- Messaging Bits with Manchester coding Error (MEI)
- Number of bits too small / Manchester coding error but messaging bits ok (NBI)
- Checksum error (CRCI)
- Frame not sent in expected time slot (TEI)
- Serial Communication: Wrong Status and Communication bit field (WSI)
- Serial Communication: CRC error (SCRI)

Transfer Management Level:

- Receive Data Buffer Overrun (RBI)
- Receive Data Memory Overrun (RMI)
- FIFO Warning Level (FWI)
- Receive Memory Underrun (RUI)
- ECU to Sensor Data Buffer Underrun (TBI)

33.12 Interrupts

8 Interrupt sources are available for the PSI5 module. For each trigger source of a channel x one interrupt can be selected in register INPx.

RDI indicates a receive data interrupt. It is activated when a received frame is moved to a Receive Data Register RDR. This happens always if at least the start bits were received without Manchester coding error.

RSI indicates a receive frame success interrupt, i.e. non of the interrupts NFI, MEI, NBI, CRCI, TEI, RBI, RMI was detected. The referring interrupt must be enabled in GCR to be considered for RSI. This allows e.g. to use only the Receive data register RDRL/Hx and ignore the Receive Data Memory RDML/Hx. Both RDI and RSI will be issued together in normal use cases where reception is correct.

RBI indicates a receive buffer overrun interrupt. It is activated when a new frame is transferred to a Receive Data Register RDR while the old value was still not read by the host ("overwrite"), i.e. the kernel wants to set any of the two interrupts RSI and RDI and finds any of these two interrupts already set.

RMI is issued if no memory buffer is available. RMI is set after a frame has been received or a No Frame Received (NFI) message is to be stored while RDIOVx.RDI[RFCx.WRP] is set. I.e. the kernel wants to set flag RDixWRP and finds any of these two flags already set. The old data is overwritten by the new data in the buffer memory.

FWI is set if the configured warning level of the FIFOx was reached. (See RFCx)

Peripheral Sensor Interface (PSI5)

TPI, TSI and TOI indicate a transmit interrupt. They are activated when data is moved from a SDR to SSR (TPI), from SSR to SOR (TSI) and from SOR to the pulse generator (TOI).

TPOI, TSOI and TOOI indicate a transfer (send) register overrun interrupt. They are set if data is written to one of the registers in the transmission chain while the referring register is locked (occupied). The data written is ignored in this case.

In addition the protocol error interrupts are available:

TEI, NBI, MEI, CRCI. If one of the protocol interrupts is activated, data is to be treated as invalid according to standard V1.3. Note that the wrong data is still available in all buffers. In particular, on MEI the wrong data is still fed into the SENT serial data analyzer (slow channel).

NFI is set, if a frame is missing in a slot configured by the watch dog timer. The module can be configured to store a complete message anyhow with flag NFI set. The slot count and the time stamp can be helpful for debugging. This provides a method for easy data handling, e.g. by using DMA transfers only without checking single interrupt bits.

WSI, SDI, SCRI treat the interrupts referring to the Status and Communication nibble.

For acceleration of the interrupt service routine, a Register INTOV is implemented that shows if an interrupt class (RSI, RDI, RBI, TDI, TBI, ERRI, SDI, FWI) is active on any of the channels. It holds a flag for each class which ORs the states from all channels of each node pointer in interrupt node pointer registers INPx. This flag is automatically set if there is an interrupt pending for the node pointer which is enabled. It is automatically reset, if no more enabled interrupt is pending for interrupt class.

In addition, there are 8 overview registers per channel holding the flags NBI, CRCI, TEI, RMI, RSI, RDI, NFI, MEI for each buffer line: NBIOVx, CRCIOVx, TEIOVx, RMIOVx, RSIOVx; RDIOVx, NFIOVx, MEIOVx; x = [1 .. 0]. E.g. NBIOV0 holds 32 bits, one for each buffer line of channel 0 reflecting if NBI is set in this buffer line.

The interrupt request or the corresponding interrupt set bit (in register INTSETA/B) can trigger the interrupt generation at the selected interrupt node. The service request pulse is generated independently from the interrupt flag in register INTSTATA/Bx. The interrupt flag can be cleared by software by writing to the corresponding bit in register INTCLRA/B.

If more than one interrupt source is connected to the same interrupt node (in register INPx), the requests are combined to one common line.

33.13 Trigger Outputs

Any interrupt source can be used as trigger output TRIGO outside the module. Each TRIGO is connected to the interrupt router. See [Table 33-4 “Service Request Lines of PSI5” on Page 33-142](#).

33.14 PSI5 Kernel Registers

This section describes the kernel registers of the PSI5 module. All PSI5 kernel register names described in this section will be referenced in other parts of the TC27x User's Manual by the module name prefix "PSI5_" for the PSI5 interface.

All registers in the PSI5 address spaces are reset with the application reset (definition see SCU section "Reset Operation").

The complete and detailed address map of the PSI5 module is described in [Table 33-2](#) on [Page 33-22](#).

w can take the values 0 ... 6 (watch dog limit values)

x can take the values 0 ... 1 (PSI5 channels)

y can take the values 0 ... 31

z can take the values 0 ... 5

Table 33-2 Registers Address Space - PSI5 Kernel Registers

Module	Base Address	End Address	Note
PSI5	F000 5000 _H	F000 5AFF _H	–

Table 33-3 Registers Overview - PSI5 Kernel Registers

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
Module Control Registers					
PSI5_CLC	Clock Control Register	000 _H	SV, U	SV, E, P	Page 33-29
-	reserved	004 _H	BE	BE	
PSI5_ID	Module Identification Register	008 _H	SV, U	BE	Page 33-28
PSI5_FDR	Module Fractional Divider	00C _H	SV, U	SV, E, P	Page 33-35
Global Registers					
PSI5_FDRL	Fractional Divider Register for lower bit rate	010 _H	SV, U	SV, E, P	Page 33-37
PSI5_FDRH	Fractional Divider Register for higher bit rate	014 _H	SV, U	SV, E, P	Page 33-38

Peripheral Sensor Interface (PSI5)
Table 33-3 Registers Overview - PSI5 Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
PSI5_FDRT	Fractional Divider Register for Time Stamp counter	018 _H	SV, U	SV, E, P	Page 33-39
PSI5_TSRA	Module Time Stamp Register A	01C _H	SV, U	SV, E, P	Page 33-41
PSI5_TSRB	Module Time Stamp Register B	020 _H	SV, U	SV, E, P	Page 33-43
PSI5_TSRC	Module Time Stamp Register C	024 _H	SV, U	SV, E, P	Page 33-45
-	reserved	028 _H	BE	BE	
PSI5_GCR	Global Control Register	02C _H	SV, U	SV, E, P	Page 33-47
Channel Registers					
PSI5_IOC Rx	Input and Output Control Register x	030 _H + x*36*4	SV, U	SV, E, P	Page 33-49
Receive Control Registers					
PSI5_RCRAx	Receiver Control Register A x	034 _H + x*36*4	SV, U	SV, E, P	Page 33-52
PSI5_RCRBx	Receiver Control Register B x	038 _H + x*36*4	SV, U	SV, E, P	Page 33-53
PSI5_RCRCx	Receiver Control Register C x	03C _H + x*36*4	SV, U	SV, E, P	Page 33-55
PSI5_WDTxw	Watch Dog Timer Register xw (x=0..1)(w=0..6)	040 _H + x*36*4 +w*4	SV, U	SV, E, P	Page 33-58
Receive Data and Status Registers					
PSI5_RSRx	Receive Status Register x	05C _H + x*36*4	SV, U	BE	Page 33-60
PSI5_SDSxz	Serial Data and Status Register of channel x frame slot z	060 _H +x *36*4+ z*4 _H	SV, U	BE	Page 33-61
PSI5_SPTSCx	Start of Pulse Time Stamp Capture Register x	078 _H + x*36*4	SV, U	BE	Page 33-62

Peripheral Sensor Interface (PSI5)
Table 33-3 Registers Overview - PSI5 Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
PSI5_SFTSCx	Start of Frame Time Stamp Capture Register x	07C _H + x*36*4	SV, U	BE	Page 33-63
PSI5_RDRLx	Receive Data Register lower bits x	080 _H + x*36*4	SV, U	BE	Page 33-64
PSI5_RDRHx	Receive Data Register higher bits x	084 _H + x*36*4	SV, U	BE	Page 33-65
Receive Data Memory					
PSI5_RDMLxy	Receive Data Buffer of channel x, line y	600 _H + x*256 +y*8	SV, U	BE	Page 33-69
PSI5_RDMHxy	Receive Data Buffer of channel x, line y	604 _H + x*256 +y*8	SV, U	BE	Page 33-70
-	reserved	900H- AFFH	BE	BE	
Receive Data Memory Control and Status Registers					
PSI5_RFCx	Receive FIFO Control of channel x	3E4 _H + x*4	SV, U	SV, U, P	Page 33-71
PSI5_RDFx	Receive Data FIFO of channel x	3F8 _H + x*4	SV, U	BE	Page 33-72
PSI5_RSIOVx	RSI Overview Register of channel x	40C _H + x*4	SV, U	BE	Page 33-74
PSI5_RMIOVx	RMI Overview Register of channel x	420 _H + x*4	SV, U	BE	Page 33-75
PSI5_NBIOVx	NBI Overview Register of channel x	434 _H + x*4	SV, U	BE	Page 33-76
PSI5_TEIOVx	TEI Overview Register of channel x	448 _H + x*4	SV, U	BE	Page 33-77
PSI5_CRCIOVx	RSI Overview Register of channel x	45C _H + x*4	SV, U	BE	Page 33-78
PSI5_RDIOVx	RDI Overview Register of channel x	470 _H + x*4	SV, U	BE	Page 33-79

Peripheral Sensor Interface (PSI5)
Table 33-3 Registers Overview - PSI5 Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
PSI5_NFIOVx	NFI Overview Register of channel x	484 _H + x*4	SV, U	BE	Page 33-80
PSI5_MEIOVx	MEI Overview Register of channel x	498 _H + x*4	SV, U	BE	Page 33-81
PSI5_RSISEx	RSI Set Register of channel x	4AC _H + x*4	nBE	SV, U, P	Page 33-82
PSI5_RMISEx	RMI Set Register of channel x	4C0 _H + x*4	nBE	SV, U, P	Page 33-83
PSI5_NBISEx	NBI Set Register of channel x	4D4 _H + x*4	nBE	SV, U, P	Page 33-84
PSI5_TEISEx	TEI Set Register of channel x	4E8 _H + x*4	nBE	SV, U, P	Page 33-85
PSI5_CRCISEx	RSI Set Register of channel x	4FC _H + x*4	nBE	SV, U, P	Page 33-86
PSI5_RDISEx	RDI Set Register of channel x	510 _H + x*4	nBE	SV, U, P	Page 33-87
PSI5_NFISEx	NFI Set Register of channel x	524 _H + x*4	nBE	SV, U, P	Page 33-88
PSI5_MEISEx	MEI Set Register of channel x	538 _H + x*4	nBE	SV, U, P	Page 33-89
PSI5_RSICLRx	RSI Clear Register of channel x	54C _H + x*4	nBE	SV, U, P	Page 33-90
PSI5_RMICLRx	RMI Clear Register of channel x	560 _H + x*4	nBE	SV, U, P	Page 33-91
PSI5_NBICLRx	NBI Clear Register of channel x	574 _H + x*4	nBE	SV, U, P	Page 33-92
PSI5_TEICLRx	TEI Clear Register of channel x	588 _H + x*4	nBE	SV, U, P	Page 33-93
PSI5_CRCICLRx	RSI Clear Register of channel x	59C _H + x*4	nBE	SV, U, P	Page 33-94
PSI5_RDICLRx	RDI Clear Register of channel x	5B0 _H + x*4	nBE	SV, U, P	Page 33-95

Peripheral Sensor Interface (PSI5)
Table 33-3 Registers Overview - PSI5 Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
PSI5_NFICLRx	NFI Clear Register of channel x	5C4 _H +x*4	nBE	SV, U, P	Page 33-96
PSI5_MEICLRx	MEI Clear Register of channel x	5D8 _H +x*4	nBE	SV, U, P	Page 33-97
-	reserved	5EC _H -5FF _H	BE	BE	
Sync Pulse Control Registers					
PSI5_PGCx	Pulse Generation Control Register x	088 _H +x*36*4	SV, U	SV, E, P	Page 33-99
PSI5_CTVx	Channel Trigger Value Register	08C _H +x*36*4	SV, U	SV, E, P	Page 33-104
Send Registers					
PSI5_SCRx	Send Control Register x	090 _H +x*36*4	SV, U	SV, E, P	Page 33-106
PSI5_SDLRx	Send Data Register Low x	094 _H +x*36*4	SV, U	SV, U, P	Page 33-111
PSI5_SDRHx	Send Data Register H x	098 _H +x*36*4	SV, U	SV, U, P	Page 33-111
PSI5_SSLRx	Send Shift Register Low x	09C _H +x*36*4	SV, U	SV, U, P	Page 33-112
PSI5_SSRHx	Send Shift Register High x	0A0 _H +x*36*4	SV, U	SV, U, P	Page 33-113
PSI5_SORLx	Send Output Register Low x	0A4 _H +x*36*4	SV, U	SV, U, P	Page 33-113
PSI5_SORHx	Send Output Reg. High x	0A8 _H +x*36*4	SV, U	SV, U, P	Page 33-114
-	reserved	0AC _H +x*36*4	BE	BE	
-	reserved	0B0 _H +x*36*4	BE	BE	
-	reserved	0B4 _H +x*36*4	BE	BE	

Peripheral Sensor Interface (PSI5)
Table 33-3 Registers Overview - PSI5 Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
Interrupt Status and Control Registers					
PSI5_INTOV	Interrupt Overview Reg.	2F8 _H	SV, U	BE	Page 33-11 5
PSI5_INTSTATAx	Interrupt Status Reg. A x	310 _H +x*4	SV, U	BE	Page 33-11 7
PSI5_INTSTATBx	Interrupt Status Reg. B x	324 _H +x*4	SV, U	BE	Page 33-12 4
PSI5_INTSETAx	Interrupt Set Register A x	338 _H +x*4	nBE	SV, E, P	Page 33-12 6
PSI5_INTSETBx	Interrupt Set Register B x	34C _H +x*4	nBE	SV, E, P	Page 33-12 8
PSI5_INTCLRAX	Interrupt Clear Reg. A x	360 _H +x*4	nBE	SV, U, P	Page 33-13 0
PSI5_INTCLRBx	Interrupt Clear Reg. B x	374 _H +x*4	nBE	SV, U, P	Page 33-13 2
PSI5_INTENAx	Interrupt Enable Reg. A x	388 _H +x*4	SV, U	SV, E, P	Page 33-13 4
PSI5_INTENBx	Interrupt Enable Reg. Bx	39CH+x*4	SV, U	SV, E, P	Page 33-13 7
PSI5_INPx	Int. Node Pointer Reg. x	2FCH+x*4 _H	SV, U	SV, E, P	Page 33-13 8
BPI Kernel Registers					
OCS	OCDS Contr. + Stat. Reg.	3CC _H	U, SV	SV, P	Page 33-30
PSI5_ACCEN0	Access Enable Register 0	3D0 _H	U, SV	SV, SE	Page 33-31
PSI5_ACCEN1	Access Enable Register 1	3D4 _H	U, SV	SV, SE	Page 33-32
PSI5_KRST0	Reset Control Register 0	3D8 _H	U, SV	SV, P	Page 33-33
PSI5_KRST1	Reset Control Register 1	3DC _H	U, SV	SV, P	Page 33-34
PSI5_KRSTCLR	Reset Status Clear Reg.	3E0 _H	U, SV	SV, P	Page 33-35

1) The absolute register address is calculated as follows:

Module Base Address ([Table 33-2](#)) + Offset Address (shown in this column)

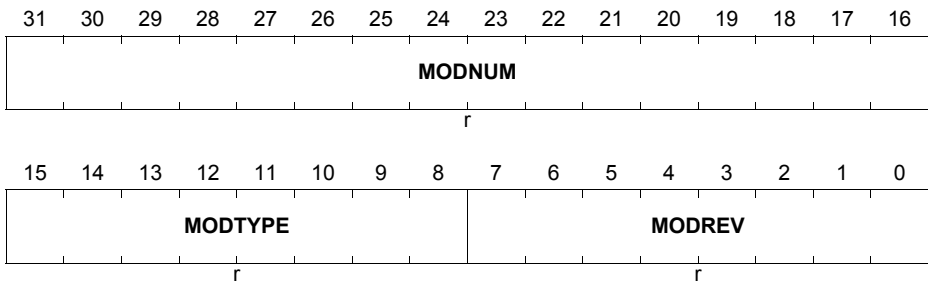
33.14.1 Module Control

Module Identification Register

The PSI5 Module Identification Register ID contains read-only information about the module version.

PSI5_ID

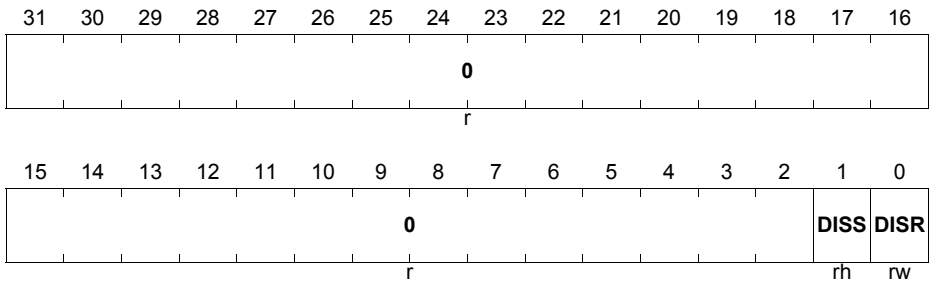
Module Identification Register (008_H) **Reset Value: 00C3 C0XX_H**



Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MODTYPE	[15:8]	r	Module Type This bit field defines the module as a 32-bit module: C0 _H
MODNUM	[31:16]	r	Module Number Value This bit field defines the module identification number for the PSI5: 00C3 _H

Clock Control Register (CLC)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI_FPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the module clock, sleep mode and disable mode for the module.

Peripheral Sensor Interface (PSI5)
PSI5_CLC
Clock Control Register
(000_H)
Reset Value: 0000 0003_H


Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Note: After a hardware reset operation, the f_{PSI5} clock is switched off and the PSI5 module is disabled (DISS set).

OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset. The OCS register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS register value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32 bit wide only and requires Supervisor Mode.

Peripheral Sensor Interface (PSI5)
PSI5_OCS
OCDS Control and Status
(3CC_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUS STA	SUS_P	SUS				0							
r		rh	w	rw				r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								r							

Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately. The kernel continues when hard suspend is left. 2 _H Soft suspend option A (Suspend after end of current send or receive transfers, if any are ongoing) others, reserved
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:0], [31:30]	r	Reserved Read as 0; must be written with 0.

Access Enable Register (ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID.

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

Peripheral Sensor Interface (PSI5)

The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 00000B, EN1 -> TAG ID 000001B, ... , EN31 -> TAG ID 011111B.

PSI5_ACCEN0
Access Enable Register 0
(3D0_H)
Reset Value: FFFF FFFF_H

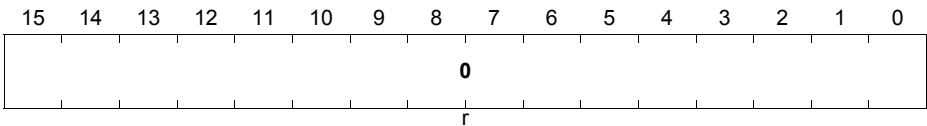
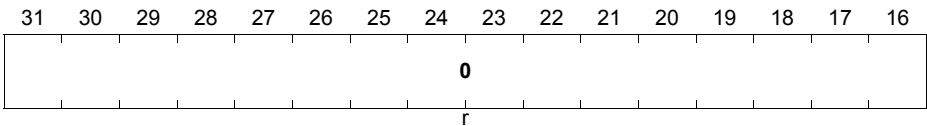
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN3	EN3	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN1	EN1	EN1	EN1	EN1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN1	EN1	EN1	EN1	EN1	EN1	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
5	4	3	2	1	0										
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register (ACCEN1)

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000B, EN1 -> TAG ID 100001B, ... , EN31 -> TAG ID 111111B.

Peripheral Sensor Interface (PSI5)
PSI5_ACCEN1
Access Enable Register 1
(3D4_H)
Reset Value: 0000 0000_H


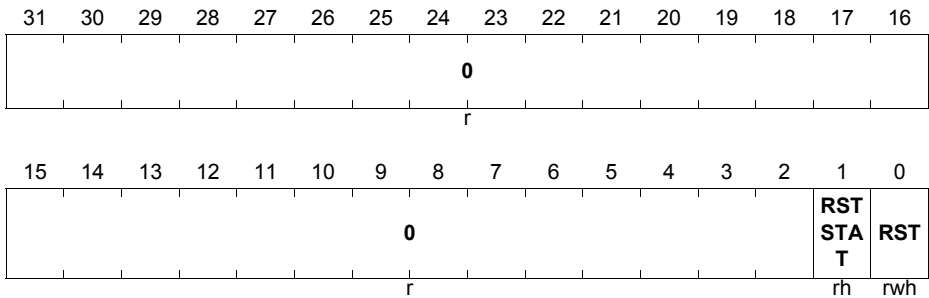
Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLE.CLR register bit.

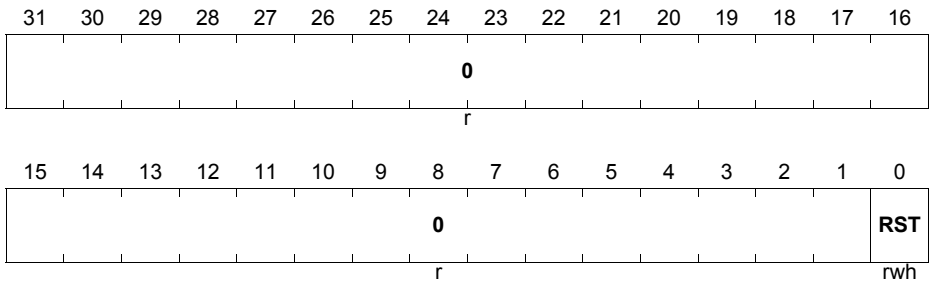
During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

Peripheral Sensor Interface (PSI5)
PSI5_KRST0
Kernel Reset Register 0
(3D8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
RSTSTAT	1	rh	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits. 0 _B No kernel reset was executed 1 _B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

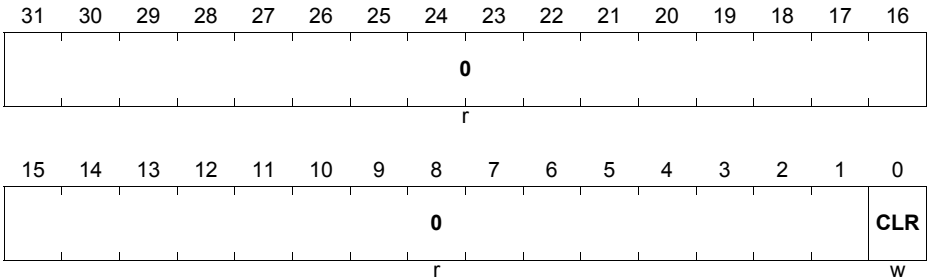
The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Peripheral Sensor Interface (PSI5)
PSI5_KRST1
Kernel Reset Register 1
(3DC_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Status Clear Register (KRSTCLR)

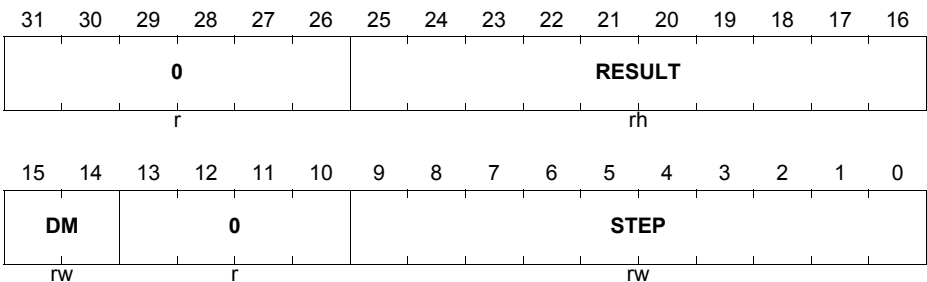
The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

Peripheral Sensor Interface (PSI5)
PSI5_KRSTCLR
Kernel Reset Status Clear Register (3E0_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Fractional Divider Register

 The Fractional Divider Register controls the input clock f_{fracdiv} .

PSI5_FDR
PSI5 Fractional Divider Register (00C_H)
Reset Value: 0000 0000_H


Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
STEP	[9:0]	rw	Step Value Reload or addition value for RESULT.
DM	[15:14]	rw	Divider Mode DM selects normal or fractional divider mode. 00 _B Fractional divider is switched off; no output clock is generated. The Reset External Divider signal is 1. RESULT is not updated (default after System Reset). 01 _B Normal Divider Mode selected. 10 _B Fractional Divider Mode selected. 11 _B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	[25:16]	rh	Result Value Bit field for the addition result.
0	[13:10], [31:26]	r	Reserved Read as 0; should be written with 0.

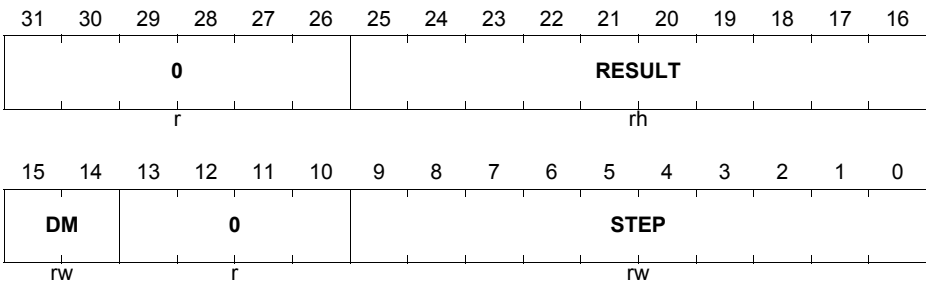
Peripheral Sensor Interface (PSI5)

Fractional Divider Register for Lower Bit Rate

The Fractional Divider Register for lower Bit Rate contains the pre divider that defines the time resolution of the f_{125} . It divides f_{fracdiv} by a factor as given in [Equation \(33.3\)](#) and [Equation \(33.4\)](#) and provides f_{125} to all channels.

FDRL

Fractional Divider Register for Lower Bit Rate (010_H) Reset Value: 0000 0000_H

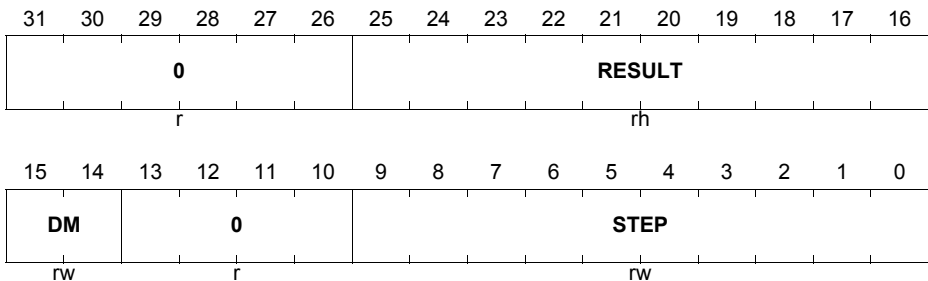


Field	Bits	Type	Description
STEP	[9:0]	rw	Step Value Reload or addition value for RESULT.
DM	[15:14]	rw	Divider Mode DM selects normal or fractional divider mode. 00 _B Fractional divider is switched off; no output clock is generated. 01 _B Normal Divider Mode selected. 10 _B Fractional Divider Mode selected. 11 _B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	[25:16]	rh	Result Value Bit field for the addition result.
0	[13:10], [31:26]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface (PSI5)

Fractional Divider Register for Higher Bit Rate

The Fractional Divider Register for Higher Bit Rate contains the pre divider that defines the time resolution of f_{189} . It divides $f_{fracdiv}$ by a factor as given in [Equation \(33.5\)](#) and [Equation \(33.6\)](#) and provides f_{189} to all channels.

FDRH
Fractional Divider Register for Higher Bit Rate (014_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
STEP	[9:0]	rw	Step Value Reload or addition value for RESULT.
DM	[15:14]	rw	Divider Mode DM selects normal or fractional divider mode. 00 _B Fractional divider is switched off; no output clock is generated. 01 _B Normal Divider Mode selected. 10 _B Fractional Divider Mode selected. 11 _B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	[25:16]	rh	Result Value Bit field for the addition result.
0	[13:10] , [31:26]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface (PSI5)
Fractional Divider Register for Time Stamp

The PSI5 Module Time Stamp Predivider Register contains the pre divider that defines the time resolution of the Time Stamp Registers TSRA/B/C and the Sync Pulse Time Base Counter SBC. It divides f_{PSI5} by a factor as given in [Equation \(33.7\)](#) and [Equation \(33.8\)](#). It contains as well the bits for reset control of the time stamp counters.

FDRT
Fractional Divider Register for Time Stamp (018_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECE C	ECE B	ECE A	ECS			RESULT									
rw	rw	rw	rw			rh									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DM		0			STEP										
rw		r			rw										

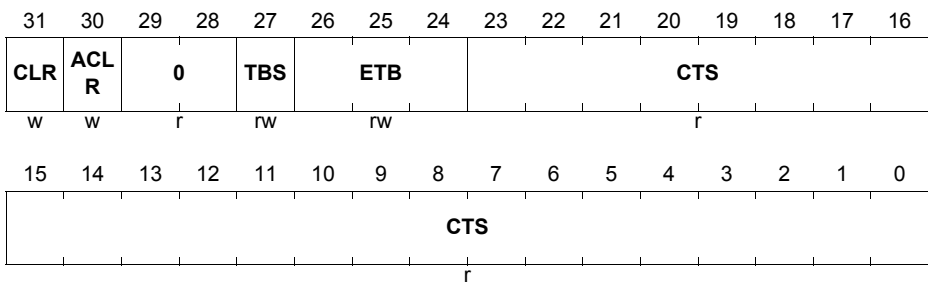
Field	Bits	Type	Description
STEP	[9:0]	rw	Step Value Reload or addition value for RESULT.
DM	[15:14]	rw	Divider Mode DM selects normal or fractional divider mode. 00 _B Fractional divider is switched off; no output clock is generated. 01 _B Normal Divider Mode selected. 10 _B Fractional Divider Mode selected. 11 _B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	[25:16]	rh	Result Value Bit field for the addition result.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
ECS	[28:26]	rw	External Time Stamp Clear Source Select Selects the external trigger line that clears the global time stamp counters TSRA/B/C.CTS if this is enabled by ECEA/ECEB/ECEC. Channel must be disabled if changed (GCR.CEN = 0). 000 _B TRIG0 001 _B TRIG1 ... _B ... 101 _B TRIG5 110 _B reserved, no trigger selected 111 _B reserved, no trigger selected
ECEA	29	rw	External Time Stamp Clear Enable A Enables the external trigger line selected by ECS to clear the global time stamp counter TSRA.CTS on rising edge of the external trigger. 0 _B disabled 1 _B enabled
ECEB	30	rw	External Time Stamp Clear Enable B Enables the external trigger line selected by ECS to clear the global time stamp counter TSRB.CTS on rising edge of the external trigger. 0 _B disabled 1 _B enabled
ECEC	31	rw	External Time Stamp Clear Enable C Enables the external trigger line selected by ECS to clear the global time stamp counter TSRC.CTS on rising edge of the external trigger. 0 _B disabled 1 _B enabled
0	[13:10]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface (PSI5)
Module Time Stamp Register A

This PSI5 Module Time Stamp Register contains read-only information about the current time given in clock cycles of f_{TS} or f_{GTM} since last reset while module was clocked. It also contains the control bits for selection of input clock and reset of this counter.

TSRA
Time Stamp Register A
(01C_H)
Reset Value: 0000 0000_H


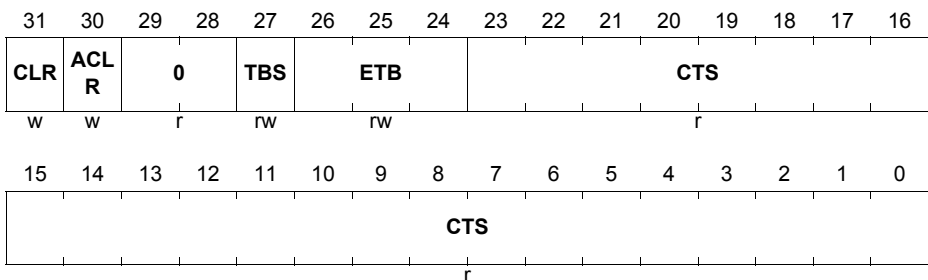
Field	Bits	Type	Description
CTS	[23:0]	r	Current Time Stamp for the Module This bit field shows the current time stamp.
ETB	[26:24]	rw	External Time Base Select Selects the external clock line for counter CTS. Channel must be disabled if changed (GCR.CEN = 0). 000 _B TRIG0 001 _B TRIG1 ... _B ... 101 _B TRIG5 110 _B reserved, no trigger selected 111 _B reserved, no trigger selected
TBS	27	rw	Time Base Select This bit selects the clock source for CTS 0 _B Internal, CTS counts in clock cycles of f_{TS} 1 _B External, CTS counts in clock cycles of f_{GTM}
ACL	30	w	Clear All Current Time Stamp Counters If set, this bit clears TSRA/B/C.CTS. TSRA/B/C.CTS count on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
CLR	31	w	Clear Current Time Stamp for the Module If set, this bit clears CTS. CTS counts on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.
0	[29:28]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface (PSI5)
Module Time Stamp Register B

This PSI5 Module Time Stamp Register contains read-only information about the current time given in clock cycles of f_{TS} or f_{GTM} since last reset while module was clocked. It also contains the control bits for selection of input clock and reset of this counter.

TSRB
Time Stamp Register B
(020_H)
Reset Value: 0000 0000_H


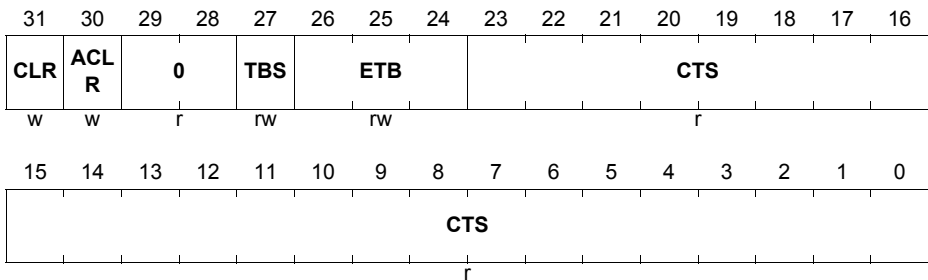
Field	Bits	Type	Description
CTS	[23:0]	r	Current Time Stamp for the Module This bit field shows the current time stamp.
ETB	[26:24]	rw	External Time Base Select Selects the external clock line for counter CTS. Channel must be disabled if changed (GCR.CEN = 0). 000 _B TRIG0 001 _B TRIG1 ... _B ... 101 _B TRIG5 110 _B reserved, no trigger selected 111 _B reserved, no trigger selected
TBS	27	rw	Time Base Select This bit selects the clock source for CTS 0 _B Internal, CTS counts in clock cycles of f_{TS} 1 _B External, CTS counts in clock cycles of f_{GTM}
ACL	30	w	Clear All Current Time Stamp Counters If set, this bit clears TSRA/B/C.CTS. TSRA/B/C.CTS count on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
CLR	31	w	Clear Current Time Stamp for the Module If set, this bit clears CTS. CTS counts on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.
0	[29:28]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface (PSI5)
Module Time Stamp Register C

This PSI5 Module Time Stamp Register contains read-only information about the current time given in clock cycles of f_{TS} or f_{GTM} since last reset while module was clocked. It also contains the control bits for selection of input clock and reset of this counter.

TSRC
Time Stamp Register C
(024_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CTS	[23:0]	r	Current Time Stamp for the Module This bit field shows the current time stamp.
ETB	[26:24]	rw	External Time Base Select Selects the external clock line for counter CTS. Channel must be disabled if changed (GCR.CEN = 0). 000 _B TRIG0 001 _B TRIG1 ... _B ... 101 _B TRIG5 110 _B reserved, no trigger selected 111 _B reserved, no trigger selected
TBS	27	rw	Time Base Select This bit selects the clock source for CTS 0 _B Internal, CTS counts in clock cycles of f_{TS} 1 _B External, CTS counts in clock cycles of f_{GTM}
ACL	30	w	Clear All Current Time Stamp Counters If set, this bit clears TSRA/B/C.CTS. TSRA/B/C.CTS count on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
CLR	31	w	Clear Current Time Stamp for the Module If set, this bit clears CTS. CTS counts on, starting from 0. This bit is automatically cleared by HW and will always be read as zero.
0	[29:28]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface (PSI5)
Global Control Register

The Global Control Register defines module wide settings.

The first 5 bits define, which error flags are regarded at RSI. RSI indicates that the referring frame is free of the selected errors. Each of these errors can be selected: NFI, TEI, NBI, MEI, CRCI.

GCR
Global Control Register
(02C_H)
Reset Value: 0000 001F_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											CEN 4	CEN 3	CEN 2	CEN 1	CEN 0
r											rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		ETC 4	ETC 3	ETC 2	ETC 1	ETC 0	0				TEI	NFI	MEI	NBI	CRCI
r		rw	rw	rw	rw	rw	r				rw	rw	rw	rw	rw

Field	Bits	Type	Description
CRCI	0	rw	CRCI is selected if bit is set.
NBI	1	rw	NBI is selected if bit is set.
MEI	2	rw	MEI is selected if bit is set.
NFI	3	rw	NFI is selected if bit is set.
TEI	4	rw	TEI is selected if bit is set.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
ETCx (x = 0-4)	8+x	rw	Enable Channel Trigger Counter CTVx.CTC This bit enables CTVx.CTC. The bits ETC0 ..x can be set with one write access to synchronously start all counters. This is required for proper sync pulse staggering. If set, CTCx counts on, starting from its current value. CTCx can be written only if ETCx is cleared (stopped). In TC27x only ETC0 .. 1 are available. All other must be written with 0, and always read 0.
CENx (x = 0-4)	16+x	rw	Enable Channel This bit enables PSI5 Channel x. If cleared, all internal state machines of the receiver and the sender are forced to default idle state while all registers can be read and written. Used for configuration of a channel.
0	[31:21], [15:13], [7:5]	r	Reserved Read as 0; should be written with 0.

33.14.2 Input and Output Control

Input and Output Control Register Functions

The Input and Output Control Register IOCR_x determines for the PSI5 channel x:

for the receiver:

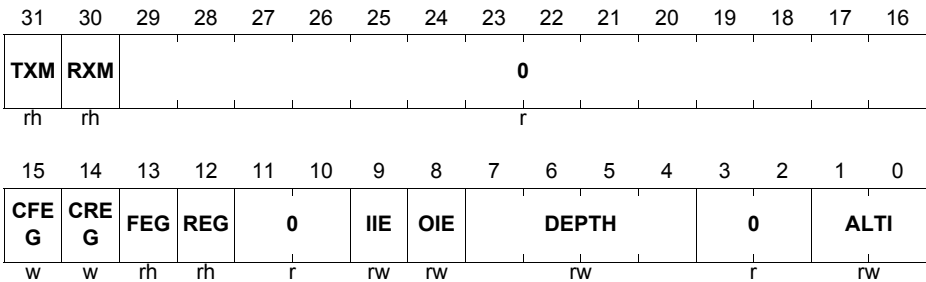
- the alternate input
- the filter depth
- the input signal polarity

for the transmitter

- the output signal polarity

IOCR_x (x = 0-1)

Input and Output Control Register x(030_H+x*36*4) Reset Value: 0000 0000_H



Field	Bits	Type	Description
ALTI	[1:0]	rw	Alternate Input Select Selects the alternate input for channel x: 00 _B Alternate Input 0 selected 01 _B Alternate Input 1 selected 10 _B Alternate Input 2 selected 11 _B Alternate Input 3 selected

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
DEPTH	[7:4]	rw	Digital Glitch Filter Depth DEPTH determines the number of port input samples clocked with f_{PSI5} that are taken into account for the calculation of the floating average. The higher DEPTH is chosen to be, the longer the glitches that are suppressed and the longer the delay of the input signal introduced by this filter. 0000 _B off, default 0001 _B 1 T_{PSI5} 0010 _B 2 0011 _B 3 ... _B ... 1111 _B 15
OIE	8	rw	Output Inverter Enable Channel x Selects the Pulse Polarity of the output of channel x 0 _B Pulse polarity is not inverted 1 _B Pulse polarity is inverted
IIE	9	rw	Input Inverter Enable Channel x Selects the Pulse Polarity of the input of channel x 0 _B Pulse polarity is not inverted 1 _B Pulse polarity is inverted
REG	12	rh	Rising Edge Glitch Flag for Channel x Shows the status of the glitch detection of channel x 0 _B No Glitch detected on rising edge 1 _B Glitch detected on rising edge REG is cleared by setting CREG.
FEG	13	rh	Falling Edge Glitch Flag for Channel x Shows the status of the glitch detection of channel x 0 _B No Glitch detected on falling edge 1 _B Glitch detected on falling edge FEG is cleared by setting CFEG.
CREG	14	w	Clear Rising Edge Glitch Flag for Channel x Clears the status flag REG 0 _B REG is not cleared 1 _B REG is cleared CREG always read zero.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
CFEG	15	w	Clear Falling Edge Glitch Flag for Channel x Clears the status flag FEG 0 _B FEG is not cleared 1 _B FEG is cleared CFEG always read zero.
RXM	30	rh	Receive Monitor for Channel x Shows the status of the receive signal of channel x after glitch filtering and inverted as specified by IIE. 0 _B Current signal is low. 1 _B Current signal is high.
TXM	31	rh	Transmit Monitor for Channel x Shows the status of the transmit signal of channel x inverted as specified by OIE. 0 _B Current signal is low. 1 _B Current signal is high.
0	[3:2], [11:10], [29:16]	r	Reserved Read as 0; should be written with 0.

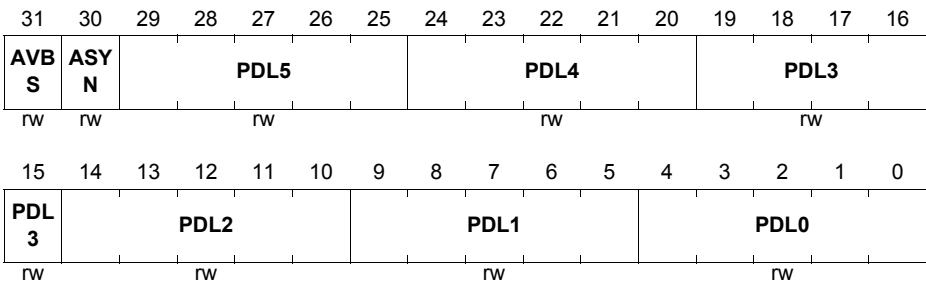
33.14.3 Receiver Control Registers

Receiver Control Register A

The Receiver Control Registers RCRA contains control bits/bit fields that are related to the PSI5 receiver operation. It configures the payload length of the up to 6 frames in the up to 6 slots.

RCRAx (x = 0-1)

Receiver Control Register A x **(034_H+x*36*4)** **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
PDLy (y = 0-5)	[5*y+4: 5*y]	rw	Payload Data Length PDL determines the number of data bits per frame that the PSI5 channel x is setup for in slot y. PDL does not include the start bits and the Parity/CRC bits. PDL includes the Messaging bits, the Frame Control bits and the Status bits. 0000 _B 0 bit data 0001 _B 1 bit data 00010 _B 2 bit data 00011 _B 3 bit data 11100 _B 28 bit data 11101 _B 0 bit data 11111 _B 0 bit data

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
ASYN	30	rw	Asynchronous Mode If set, <ul style="list-style-type: none"> the watch dog timers for the PSI5 channel x, slot WDL1 .. 6 are disabled If WDL0 is not cleared TEI is issued if the distance between two frames is longer than specified in WDL0 Slot Counter SC is incremented with each received frame (works as frame counter) with roll over after 6. No Sync Pulses generated, TX path is inactive 0_B off (default) 1_B on
AVBS	31	rw	Verbose Mode for Asynchronous Mode If set, and ASYN is set and WDL0 is > 0 an empty frame is stored in RDRL/Hx and the referring RDML/H each time, the watch dog timer for the PSI5 channel x without reception of a frame. 0_B off (default) 1_B on

Receiver Control Register B

The Receiver Control Registers RCRBx contains control bits/bit fields that are related to the PSI5 receiver operation. It configures up to 6 frame types in the up to 6 slots.

RCRBx (x = 0-1)

Receiver Control Register B x **(038_H+x*36*4)** **Reset Value: 0000 0000_H**

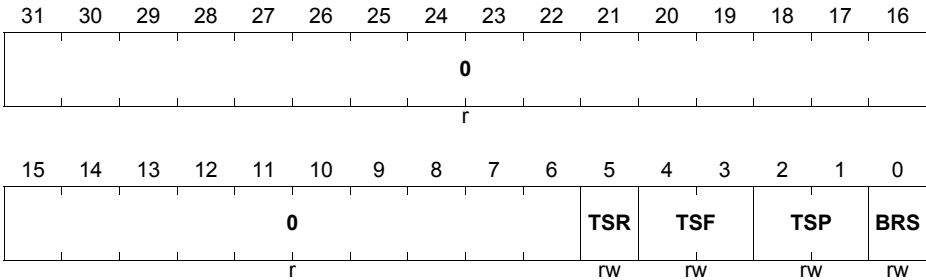
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								VBS	FEC	CRC	MSG	VBS	FEC	CRC	MSG
								5	5	5	5	4	4	4	4
r								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VBS	FEC	CRC	MSG	VBS	FEC	CRC	MSG	VBS	FEC	CRC	MSG	VBS	FEC	CRC	MSG
3	3	3	3	2	2	2	2	1	1	1	1	0	0	0	0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
MSGy (y = 0-5)	4*y	rw	Messaging Bits If set, the 2 Messaging bits are configured for the PSI5 channel x in slot y. If set, Extended Serial Message processing on bits D[1:0] is activated. (18 frames, 4 or 8 bit ID, 12 or 16 bit data, 6 bit CRC) If Messaging bits are transmitted they are always presented in RDRx independently from status of MSGy. I.e. RDRx always stores exactly the data that was received. 0 _B No Messaging bits (default) 1 _B 2 Messaging bits
CRCy (y = 0-5)	4*y+1	rw	CRC or Parity Selection If set, a 3 bit CRC checksum is expected for the PSI5 channel x in slot y. Else, 1 bit Parity is assumed. 0 _B 1 Parity Bit is configured (default) 1 _B 3 CRC bits are configured
FECy (y = 0-5)	4*y+2	rw	Frame Expectation Control If set, a frame is expected for the PSI5 channel x in slot y. A No Frame Received error interrupt NFI is issued each time, the watch dog timer for the PSI5 channel x, slot y expires without reception of a SOF or EOF. 0 _B No Message expected (default) 1 _B A Message is expected, SC is checked
VBSy (y = 0-5)	4*y+3	rw	Verbose Mode If set, and a message is expected for the PSI5 channel x in slot y (FECy) an empty frame is stored in RDRL/Hx and the referring RDML/H each time, the watch dog timer for the PSI5 channel x, slot y expires without reception of a frame. 0 _B off (default) 1 _B on
0	[31:24]	r	Reserved Read as 0; should be written with 0.

Receiver Control Register C

The Receiver Control Registers RCRCx contains control bits/bit fields that are related to the PSI5 receiver operation. Bit Rate and Time Stamp Sources are configured here.

Peripheral Sensor Interface (PSI5)
RCRCx (x = 0-1)
Receiver Control Register C x (03C_H+x*36*4) Reset Value: 0000 0000_H


Field	Bits	Type	Description
BRS	0	rw	Baud Rate Select 0 _B f_{125} selected for channel x 1 _B f_{189} selected for channel x
TSP	[2:1]	rw	Time Stamp Select for Pulses 00 _B TSRA.CTS is captured in SPTSC 01 _B TSRB.CTS is captured in SPTSC 10 _B TSRC.CTS is captured in SPTSC 11 _B not defined. Do not use
TSF	[4:3]	rw	Time Stamp Select for Start of Frame (SOF) 00 _B TSRA.CTS is captured in SFTSC 01 _B TSRB.CTS is captured in SFTSC 10 _B TSRC.CTS is captured in SFTSC 11 _B not defined. Do not use
TSR	5	rw	Time Stamp Select for Receive Data Registers 0 _B SPTSC is stored in RDRHx 1 _B SFTSC is stored in RDRHx
0	[31:6]	r	Reserved Read as 0; should be written with 0.

Watch Dog Timer Register

Synchronous Mode

Selected with RCRAx.ASYN = 0.

The Watch Dog Timer Registers WDTxy contain 7 time limits defining 6 frame slots.

Peripheral Sensor Interface (PSI5)

The internal Watch Dog timer is started automatically at the start of a sync pulse. The value entered here defines the time in multiples of TTS (defined in FDRT) from the last start of a sync pulse on channel x.

The internal Watch Dog Timer is compared to WDL_x. A match triggers:

- the automatic incrementing of slot counter RDRH.SC_x
- starting from WDL_x1 on: check if a frame with the corresponding SC y has been received or if such a frame is still being received since last WDL_x expired. If this is not the case but a frame was expected in this slot y (configured so in RCRB_x.FEC_y) “No Frame Received” Interrupt NFI is issued. If configured so (in RCRB_x.VBS_y), an empty frame (all data fields are ‘0’) is stored in RDRL/H_x and the referring RDML/H.
- Frames that start before a slot boundary and end after this boundary but are too short and thus discarded will result in an NFI / empty frame as well (if configured).
 - If no frame was received before the SOF in this slot,
SC = slot where SOF happened.
Time Stamp = end of slot where SOF happened.
 - If a good frame was received in this slot before a second SOF crosses WDL, the parameters for the next slot are taken over already and if no good frame is received in this next slot:
SC = next slot (SC where SOF happened + 1)
Time Stamp = end of next slot (SC where SOF happened + 1).
 - If a good frame was received in this slot before a second SOF crosses WDL, the parameters for the next slot are taken over already and if a good frame is received in this next slot: no NFI is issued.
- (The remarks about the Time Stamps assume that RCRC_x.TSR selects SFTSC_x.)
- A frame starting before a slot y and ending in slot y will not prevent NFI for slot y.

Detection of Frame Boundaries:

- SOF is the first rising edge of the first start bit after an “Idle Time”, a time stamp is triggered and SC is captured for the frame at this point in time
- EOF is determined based on the configured frame length (RCRAX.PDL_y and RCRC_x.CRC_y). An unexpected “Idle Time” or Manchester Error will result in earlier EOF detection.
- For each frame, the receiver checks if slot counter SC at SOF is equal to SC at EOF. If not, the interrupt TEI is issued. FEC is not relevant for TEI.

If WDL_x is cleared no check is performed but RDRH.SC is still incremented when it is its turn in the sequence of the WDLs. Thus such unconfigured WDL_x must appear only at the end. If an unconfigured WDL_x appears between configured WDL_x, the slot count will be confused.

If a frame is received in a slot where no frame is expected, no dedicated flag is set but the frame is received with the parameters (PDL, SC and on) given for this slot. SW will detect this as SC for this frame is wrong.

Asynchronous Mode

Peripheral Sensor Interface (PSI5)

Selected with RCRAx.ASYN = 1.

The watch dog timers for the PSI5 channel x, slot WDL1 .. 6 are disabled. If WDL0 is not cleared TEI is issued if the distance between two frames is longer than specified in WDL0. Slot Counter SC is incremented with each received frame (works as frame counter) with roll over after 6. If AVBS (verbose mode) is set too, an empty frame is stored in RDRL/Hx and the referring RDML/H.

Channel Configuration at Slot Boundaries / Take over of Slot Parameters

The receiver takes over the parameters for a frame at the beginning of a slot
- with 3 exceptions:

Exception 1)

During SLOT0 (this is the slot after the last configured slot), after the sync pulse, the configuration of SLOT1 is already used. An internal shadow slot count value makes sure, the messaging bits are routed to slow channel for SLOT1. This way, a frame for SLOT1 that is too early will be received correctly, the messaging bits are sorted correctly. The time violation is still visible by TEI = 1 as such a frame would still start in a different slot (SLOT0) than it ends (SLOT1). SC = 1 will be stored in the receive buffer/memory.

Exception 2)

Frames that come too early could destroy the slow channel of the preceding slot. To prevent this, during SLOT1 .. SLOT5, the configuration of the next slot is loaded immediately after EOF was received. Note that an SOF must be seen first in the same slot before an EOF is looked for. This way, the HW prevents the destruction of a good frame by a preceding frame that is too late. An internal shadow slot count value makes sure, the messaging bits are routed to the slow channel for the following SLOT. This way, a frame for e.g. SLOT3 that is too early will be received correctly, the messaging bits are sorted correctly in SLOT3 and not in SLOT2 where the SOF of this frame is. The time violation is still visible by TEI = 1 as such a frame would still start in a different slot (SLOT2) than it ends (SLOT3). SC = 3 will be stored in the receive buffer/memory.

Exception 3)

If a frame does not end at the configured end of a slot, the new parameters of the following slot are taken over by the receiver only after the EOF was detected. This allows to receive a frame properly that is too late.

Note that orphan frames, i.e. frames that come with a timely shift of one or more slots can not be detected as such. Thus they will be stored with the SC as they appear.

Notes on Frames coming too early

Peripheral Sensor Interface (PSI5)

A frame may come too early and thus start in a slot that is configured for no frame (FEC = 0).

If the user wants to make sure that this frame is stored in RDR/RDM, the slot with FEC = 0 must be configured in the same way as its follower.

At least, the length field PDL must be set to its maximum value (28) to allow for reception of all frame lengths. This allows to capture such a frame with referring error bits. (TEI, NBI, CRC in most cases)

If this is not done and PDL is (by default) = 0 the following happens:

CASE1: SOF is too early and thus in the slot with FEC = 0, due to PDL = 0 also EOF is detected in the slot with FEC = 0.

The Frame is stored in RDR/RDM with Time Stamp, 0 bits of data (as configured in PDL), Slot Count of the too early slot (with FEC =0), CRCI is generated as the FSM assumes that no CRC can be checked ok on no data, NBI is issued.

CASE2: SOF is too early and thus in the slot with FEC = 0, although PDL = 0 EOF is detected in the slot FOLLOWING the slot with FEC = 0 (because CRC transmission end here, i.e. the SOF was very close to the end of the slot)

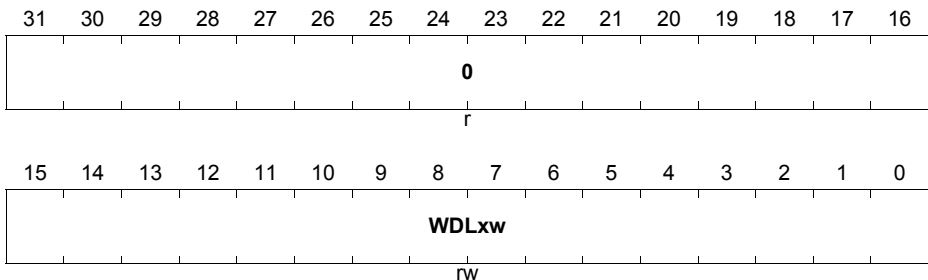
The Frame is stored in RDR/RDM with Time Stamp, 0 bits of data (as configured in PDL), Slot Count of the too early slot (with FEC =0), CRCI is generated as the FSM assumes that no CRC can be checked ok on no data, NBI and TEI bits are set.

WDT0w (w = 0-6)

Watch Dog Timer Register 0w **(040_H+w*4)** **Reset Value: 0000 0000_H**

WDT1w (w = 0-6)

Watch Dog Timer Register 1w **(0D0_H+w*4)** **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
WDLxw	[15:0]	rw	Watch Dog Timer Limit for channel x limit w.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
0	[31:16]	r	Reserved Read as 0; should be written with 0.

Usage of the Watch Dog Timer Limits:

The example below shows some use cases.

F1: "Frame too early": Slot Count SC at SOF is 0 and 1 at EOF. Timing Error is issued (TEI). F1 is stored with configuration of Slot 1 and SC = 1.

WDL2: Slot Count SC incremented by HW, no "No Receive Frame" Interrupt as no frame expected (FEC=0)

F3: "Frame too late": Slot Count SC at SOF is 3 and 4 at EOF. Timing Error is issued (TEI). The configuration for Slot 3 is kept until the end of F3 so that it is not destroyed. Note that the EOF of Frame 3 does not lead to take over the configuration of Slot 5! But the EOF of F4 does. This way a F5 coming too early would be received correctly.

WDL5: "Frame missing": No SOF or EOF received since WDL4. A frame was expected (FEC=1), No Receive Frame Interrupt is issued.

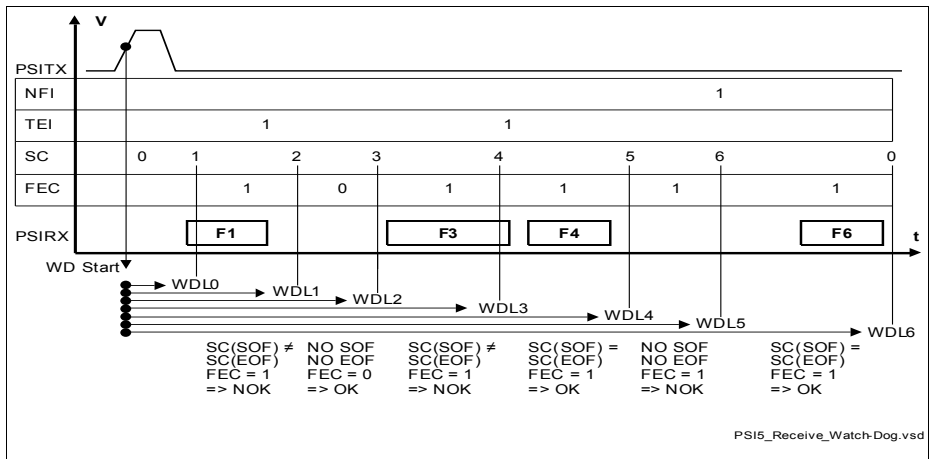


Figure 33-18 Watch Dog Timer Limits

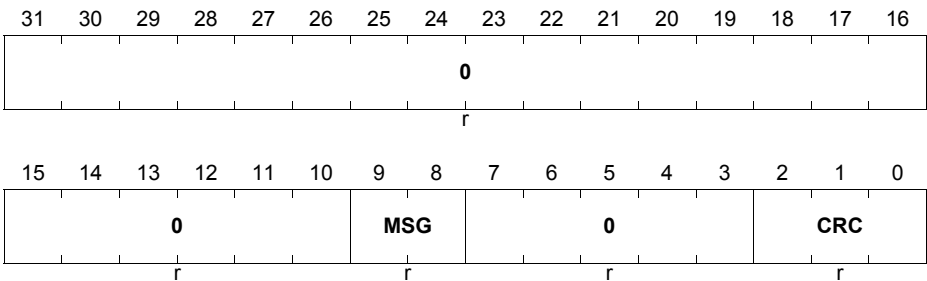
33.14.4 Receive Data and Status Registers

Receiver Status Register

The Receive Status Register provides the status information of channel x.

RSRx (x = 0-1)

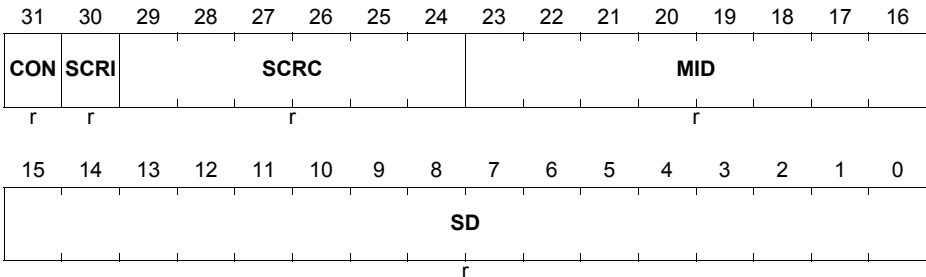
Receive Status Register x (05C_H+x*36*4) Reset Value: 0000 0000_H



Field	Bits	Type	Description
CRC	[2:0]	r	CRC of last frame. CRC0 is on bit position 0.
MSG	[9:8]	r	Messaging Bits of last frame. MSG0 is on bit position 8.
0	[7:3], [31:10]	r	Reserved Read as 0; should be written with 0.

Serial Data and Status Register (“Slow Channel”)

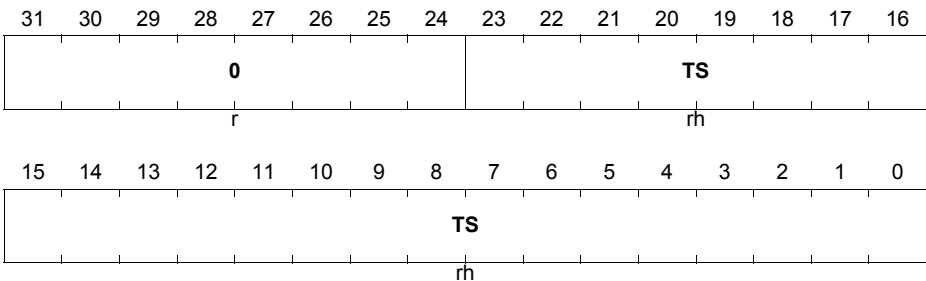
The Serial (Receive) Data and Status Register provides the data and status information of channel x slot y. The data is presented in bit field SD independent from the SCRC check result in SCRI.

Peripheral Sensor Interface (PSI5)
SDS0z (z = 0-5)
Serial Data and Status Register 0z (060H+z*4)
Reset Value: 0000 0000_H
SDS1z (z = 0-5)
Serial Data and Status Register 1z (0F0H+z*4)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
SD	[15:0]	r	Serial Data of last serial data frame on channel x slot y. SD0 is on bit position 0. If SDS.CON is not set, bits [15:12] are zero.
MID	[23:16]	r	Message ID of last serial data frame. ID0 is on bit position 16. If SDS.CON is set, bits [23:20] are zero.
SCRC	[29:24]	r	SCRC CRC of last serial data frame. CRC0 is on position 24.
SCRI	30	r	CRC of Serial Message failed Interrupt Flag. This bit is set if the CRC of the serial message fails. This includes a check of the Messaging bit field for correct 0 values of bit 1 in frames 7, 13 and 18. See INTSTATx.
CON	31	r	Configuration bit of last serial frame. 0 _B 12-bit data and 8-bit message ID 1 _B 16-bit data and 4-bit message ID

Peripheral Sensor Interface (PSI5)
Start of Pulse Time Stamp Capture Register SPTSCx

Each time a sync pulse is sent on channel x, the time stamp is captured in this register. RCRC.TSP selects, if TSRA, B or C is used. RCRC.TSR determines if SPTSCx or SFTSCx is stored in RDRHx.

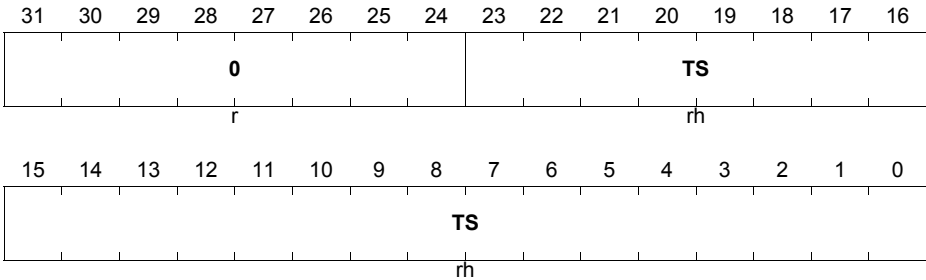
SPTSCx (x = 0-1)
SOP TS Capture Register SPTSCx(078_H+x*36*4)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
TS	[23:0]	rh	Time Stamp of the last sync pulse sent on channel x. The Time Stamp is taken at the rising edge of the pulse.
0	[31:24]	r	Reserved Read as 0; should be written with 0.

Start of Frame Time Stamp Capture Register SFTSCx

Each time a start of frame is received on channel x, the time stamp is captured in this register. The first rising edge of the first start bit is captured. Note that a preceding "Idle Time" is required for start bit detection. In case the two start bits should not be received successfully (i.e. without manchester coding errors) the frame is completely dropped and the latest captured value is overwritten with the next SOF or at the referring WDLy for "empty messages" if configured in RCRBx.VBSy. RCRC.TSF selects, if TSRA, B or C is used. RCRCx.TSR determines if SPTSCx or SFTSCx is stored in RDRHx.

Peripheral Sensor Interface (PSI5)

SFTSCx (x = 0-1)
SOF TS Capture Register SFTSCx(07C_H+x*36*4)
Reset Value: 0000 0000_H


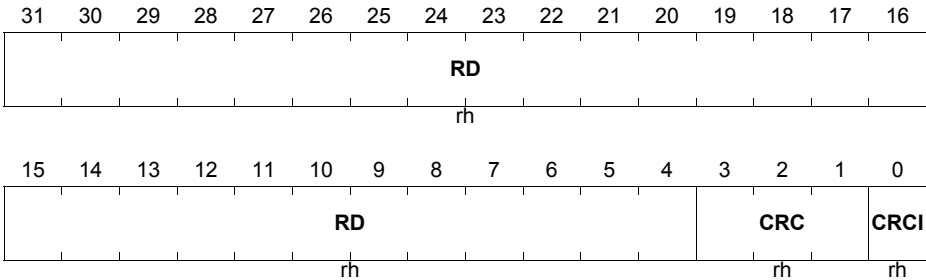
Field	Bits	Type	Description
TS	[23:0]	rh	Time Stamp of the last frame received on channel x. The Time Stamp is taken at the rising edge of the first start bit S1 that was qualified by the receiver.
0	[31:24]	r	Reserved Read as 0; should be written with 0.

Receive Data Registers RDRLx

The Receive Data Register RDRLx for channel x shows the data content of a received data frame.

The internal receive buffer is always cleared (0x0000 0000_H) at each frame start. Thus unused nibbles are always read as zero.

The bit CRCI reflect, whether the conditions are met to issue an interrupt signal and set the referring interrupt status bit in INTSTATA for the latest frame. Thus it is independent from the old status of the referring sticky bit in INTSTATA before latest frame reception.

Peripheral Sensor Interface (PSI5)
RDRLx (x = 0-1)
Receive Data Register Low x (080H+x*36*4) Reset Value: 0000 0000_H


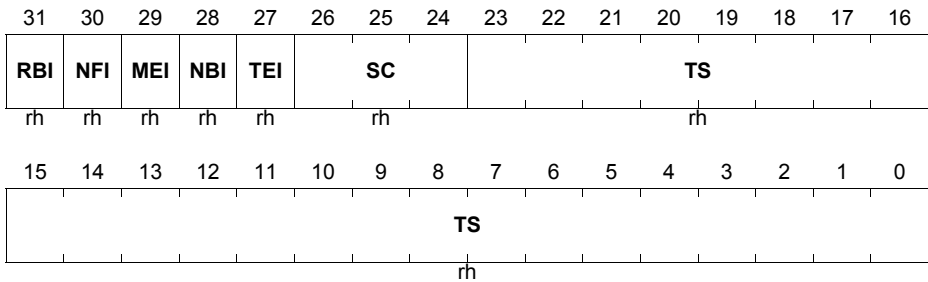
Field	Bits	Type	Description
CRCI	0	rh	CRC Error Flag This bit is set if the CRC or Parity check fails. Set as well if CRC can not be calculated: PDL = 0, CRC cut off by Manchester Error, too few bits, BOT, Sync Pulse) 0 _B correct CRC/Parity 1 _B wrong CRC/Parity
CRC	[3:1]	rh	CRC CRC/parity bit of last PSI5 frame. CRC0 or Parity bit is on bit position 1. In case of NFI, this field is cleared.
RD	[31:4]	rh	RD Receive data of last PSI5 frame. D0 is on bit position 4. In case of NFI, this field is cleared.

Receive Data Registers RDRHx

The Receive Data Register RDRHx for channel x shows the data content of a received data frame.

The internal receive buffer is always cleared (0x0000 0000_H) at each frame start. Thus unused bits are always read as zero.

The bits RBI, RMI, RSI, NBI and TEI reflect, whether the conditions are met to issue an interrupt signal and set the referring interrupt status bit in INTSTATA for the latest frame. Thus it is independent from the the old status of the referring sticky bit in INTSTATA before latest frame reception.

Peripheral Sensor Interface (PSI5)
RDRHx (x = 0-1)
Receive Data Register High x (084H+x*36*4) Reset Value: 0000 0000_H


Field	Bits	Type	Description
TS	[23:0]	rh	Time Stamp of the last received PSI5 frame. RCRC.TSR determines if SPTSCx or SFTSCx is stored in RDRHx. (It can be selected if the time stamp of the last sync pulse is stored or the time stamp for the start of frame).
SC	[26:24]	rh	Slot Counter In Synchronous Mode (RCRAx.ASYN=0): Number of the slot the SOF of the frame was received in. In Asynchronous Mode (RCRAx.ASYN=1): revolving frame number between 1 and 6. 000 _B not valid (SOF too early) 001 _B Slot 1 ... _B ... 110 _B Slot 6 111 _B not valid
TEI	27	rh	Time Slot Error Flag In Synchronous Mode (RCRAx.ASYN = 0), this bit is set if the SOF and the EOF of a frame are received in different time slots. The slots are constraint by the watch dog timer (see WDTxy). In Asynchronous Mode (RCRAx.ASYN = 1), this bit is set if the distance between two frames is longer than specified in WDL0. 0 _B no error 1 _B error

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
NBI	28	rh	<p>Number of bits Error Flag</p> <p>This bit is set if the last frame received less bits than expected but no manchester coding error occurred until S1, S2, M0 and M1 where received.</p> <p>Note that the frame after the error might be completely wrong.</p> <p>0_B all bits received 1_B not all bits received</p>
MEI	29	rh	<p>Error in Message Bits Flag</p> <p>This bit is set if a manchester error occurred in Bit M0 or M1. (Only if configured in RCRBx.MSG)</p> <p>Note that the frame after the error might be completely wrong.</p> <p>0_B frame is correct 1_B frame is not correct</p>
NFI	30	rh	<p>No Frame Received Flag</p> <p>This bit is set at the end of a slot if neither an SOF nor an EOF was received in this slot. If only the two start bits are received and no further data, this "SOF" is discarded completely and treated like no frame received as well.</p> <p>If RCRBx.VBS is set, an empty frame is stored anyhow and RDIX is issued as well. The content of RDRL/Hx / RDML/Hx is valid for SC, TS (captured at the end of the slot) and for this bit (NFI). Bit fields RD and CRC are cleared (all zero).</p> <p>0_B no error 1_B error</p>
RBI	31	rh	<p>Receive Buffer Overflow Flag</p> <p>This bit is set after a frame has been received while the old one was not read from RDRHx. I.e. the kernel wants to set any of the two interrupts RSI and RDI and finds any of these two interrupts already set. The old data is overwritten by the new data.</p> <p>0_B No overflow 1_B Overflow</p>

33.14.5 Receive Data Memory

The module design allows for usage in 2 ways without mode selection:

Ring Buffer Usage

The Receive Data Memory x is built as a ring buffer. I.e. the first frame is stored in RDML/Hx0, the next in RDML/Hx1 and so on. At buffer 31 a wrap around is performed automatically. I.e. the next frame is stored in buffer 0. RFCx.WRP is showing the last location written and automatically incremented after write.

Receive Buffer Overrun Interrupt Flag RBI is set, if a buffer is written before the SW cleared Receive Data Interrupt Overview Flag RDIOVx (unchecked or erroneous frame received or No Frame Received (NFI) message to be stored).

FIFO Usage

Writing to the Receive Data Memory by the module moves the write pointer after writing. Reading from RDFx 2 times with a 32 bit word access moves the read pointer to the next younger buffer automatically after the read access. It clears as well all interrupt overview bits for this buffer line in NBIOVx, CRCIOVx, TEIOVx, RMIOVx, RSIOVx; RDIOVx, NFIOVx, MEIOVx. This allows for fully automatic DMA transfers without CPU intervention. The DMA simply needs to be set up in a way that it reads exactly the number of buffers configured with the FIFO warning level FWL. The application SW configures "FIFO Warning Level Request Interrupt" FWI so that it is routed to the referring DMA channel as trigger source. Note that FWI does not need to be cleared to repetitively trigger the DMA.

If RFCx.WRAP is cleared and the read pointer (after the auto increment said above) equals the write pointer while reading from RDFx, a Receive Memory Underrun Interrupt RUIx is flagged and the read pointer is not moved on automatically even if RDFx is read.

IF RFCx.FRQ is set, the FIFO needs to be flushed by RFCx.FLU.

Peripheral Sensor Interface (PSI5)

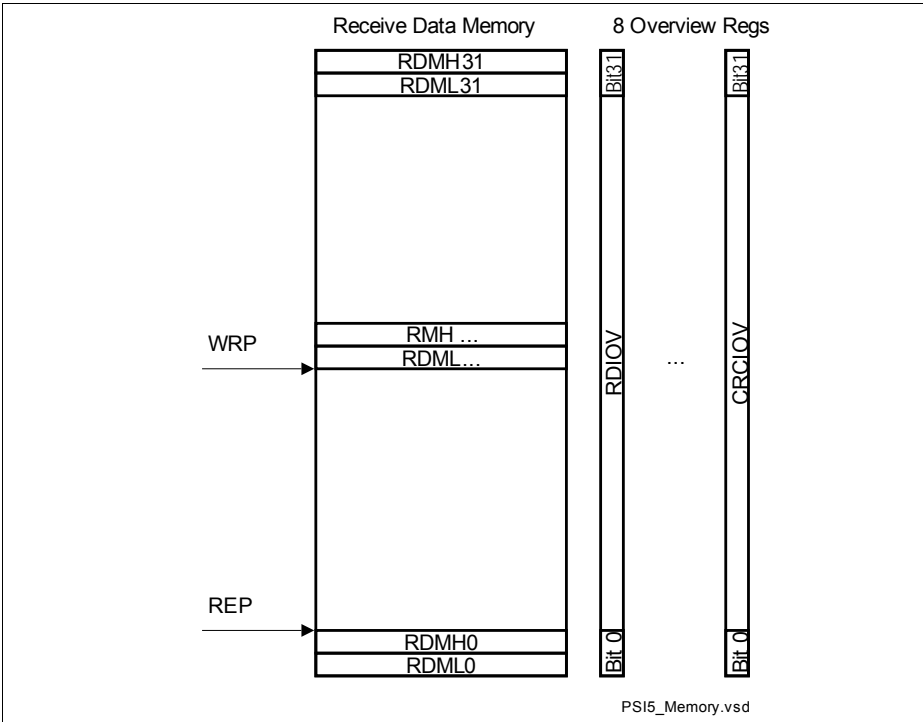
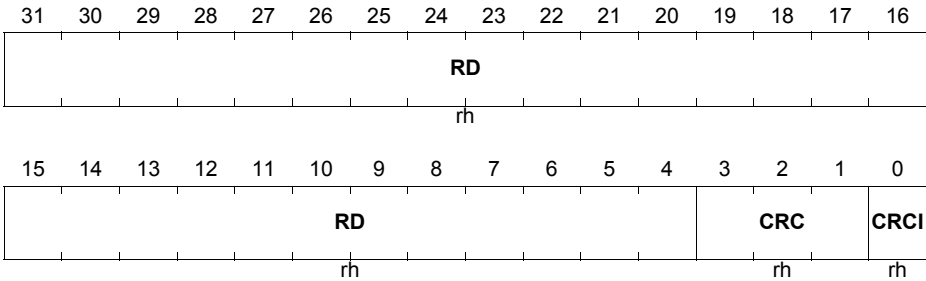


Figure 33-19 Receive Data Memory

Receive Data Memory RDMLx_y

The Receive Data Memory RDML_x for channel x shows the data content of up to 32 received data frames.

The internal receive buffer is always cleared (0x0000 0000_H) at each frame start. Thus unused nibbles are always read as zero.

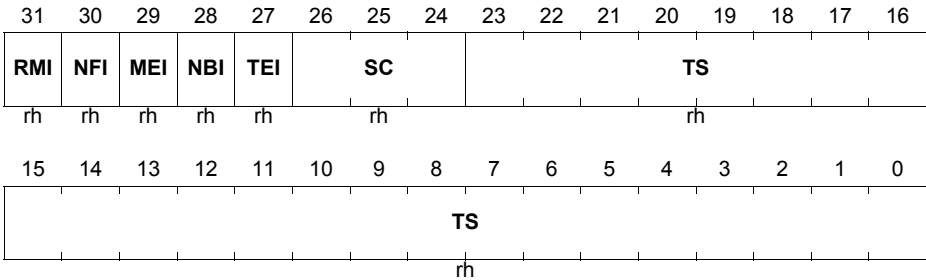
Peripheral Sensor Interface (PSI5)
RDML0y (y = 0-31)
Receive Data Memory Low 0y (600H+y*8) Reset Value: 0000 0000_H
RDML1y (y = 0-31)
Receive Data Memory Low 1y (700H+y*8) Reset Value: 0000 0000_H


Field	Bits	Type	Description
CRCI	0	rh	CRC Error Flag Copied from RDRLx at frame end.
CRC	[3:1]	rh	CRC Copied from RDRLx at frame end.
RD	[31:4]	rh	RD Copied from RDRLx at frame end.

Receive Data Memory RDMHxy

The Receive Data Memory RDMHx for channel x shows the data content of up to 32 received data frames.

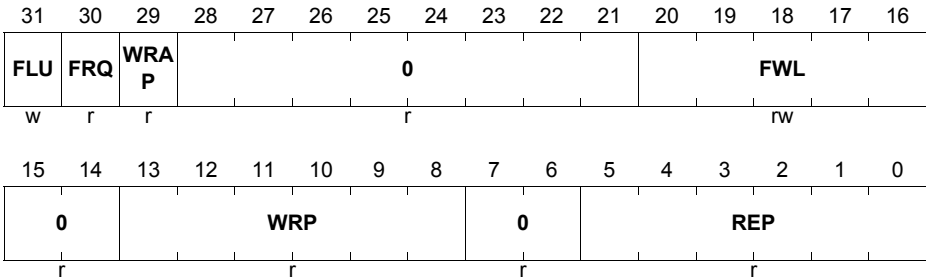
The internal receive buffer is always cleared (0x0000 0000_H) at each frame start. Thus unused nibbles are always read as zero.

Peripheral Sensor Interface (PSI5)
RDMH0y (y = 0-31)
Receive Data Memory High 0y (604H+y*8) Reset Value: 0000 0000_H
RDMH1y (y = 0-31)
Receive Data Memory High 1y (704H+y*8) Reset Value: 0000 0000_H


Field	Bits	Type	Description
TS	[23:0]	rh	Time Stamp Copied from RDRHx at frame end.
SC	[26:24]	rh	Slot Counter Copied from RDRHx at frame end.
TEI	27	rh	Time Slot Error Flag Copied from RDRHx at frame end.
NBI	28	rh	Number of bits Error Flag Copied from RDRHx at frame end.
MEI	29	rh	Error in Messaging Bits Flag Copied from RDRHx at frame end.
NFI	30	rh	No Frame Received Flag Copied from RDRHx at frame end.
RMI	31	rh	Receive Memory Overflow Flag Copied from INTSTATAx at frame end.

Receive FIFO Control Register RFCx

RFCx contains control bits for the FIFO and Ring Buffer operation.

Peripheral Sensor Interface (PSI5)
RFCx (x = 0-1)
Receive FIFO Control Register x (3E4_H+x*4)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
REP	[5:0]	r	FIFO Read Pointer points to the buffer to be read next. Incremented after read. The last bit indicates if RDMxH or RDMxL is read. This LSB is ignored for FWL comparison WRP - REP collision (RBI, RUI)
WRP	[13:8]	r	FIFO/Ring Buffer Write Pointer points to the buffer written last. Incremented before write. The last bit indicates if RDMxH or RDMxL is written. This LSB is ignored for FWL comparison WRP - REP collision (RBI, RUI).
FWL	[20:16]	rw	FIFO Warning Level Number of new entries at which the FIFO Warning Interrupt FWIx is set automatically. For calculation WRP and REP are used. I.e. only if the memory is read via RDFx, this works properly.
WRAP	29	r	Write Pointer WRAP Indicator If set, the Write Pointer is one wrap around ahead of the Read Pointer. It is cleared, if the the Read Pointer wraps around too or when FLU is set by SW.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
FRQ	30	r	Flush Request if set, FIFO read and write pointers are not incremented any more due to excessive RMI. (Too many overrun conditions). This bit is cleared when FLU is set.
FLU	31	w	Flush if set, FIFO read and write pointers are cleared. Bits and WRAP are cleared. SW needs to clear interrupts as needed. This bit can only be set and is cleared by HW. Always read as 0.
0	[28:21] , [15:14] , [7:6]	r	Reserved Read as 0; should be written with 0.

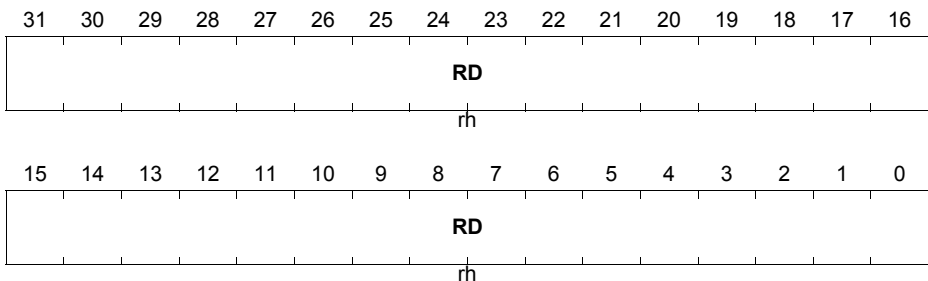
Receive Data FIFO RDFx

The Receive Data FIFO RDFx for channel x shows the oldest data content of up to 32 received data frames. I.e. RFCx.REP is pointing at the buffer presented.

Do not set a debugger watch point on this register as any read access will issue a hard ware request for new data (move read pointer REP and present referring memory content in RDFx). A debugger watch point can be set on the RDML/Hx and RFCx to monitor the FIFO behavior.

RDFx (x = 0-1)

Receive Data FIFO x (3F8H+x*4) **Reset Value: 0000 0000_H**



Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
RD	[31:0]	rh	RD Shows the content of RDML/Hxy; y = RFCx.REP[5:1]; L/H = RFCx.REP[0]. Reading this register triggers incrementation of REP and presentation of next FIFO value at next read access! Once a complete buffer is read (64 bit, 2 accesses to RDF) all overview flags for this very buffer are cleared automatically.

Peripheral Sensor Interface (PSI5)
RSI Overview Register

The RSI Overview Register RSIOVx contains flags to manage the receive memory of PSI5 channel x.

RSIOVx (x = 0-1)

RSI Overview Register **(40C_H+x*4)** **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSI3	RSI3	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI1	RSI1	RSI1	RSI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSI1	RSI1	RSI1	RSI1	RSI1	RSI1	RSI9	RSI8	RSI7	RSI6	RSI5	RSI4	RSI3	RSI2	RSI1	RSI0
5	4	3	2	1	0										
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RSly (y = 0-31)	y	rh	RSI Flag of Buffer y Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit RSICLRx.RSly. This bit can be set by bit RSISEx.RSly.

Peripheral Sensor Interface (PSI5)
RMI Overview Register

The RMI Overview Register RMIOVx contains flags to manage the receive memory of PSI5 channel x.

RMIOVx (x = 0-1)

RMI Overview Register (420_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMI3	RMI3	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI1	RMI1	RMI1	RMI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMI1	RMI1	RMI1	RMI1	RMI1	RMI1	RMI9	RMI8	RMI7	RMI6	RMI5	RMI4	RMI3	RMI2	RMI1	RMI0
5	4	3	2	1	0										
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RMly (y = 0-31)	y	rh	RMI Flag of Buffer y Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit RMICLRx.RMly. This bit can be set by bit RMISEx.RMly.

Peripheral Sensor Interface (PSI5)
NBI Overview Register

The NBI Overview Register NBIOVx contains flags to manage the receive memory of PSI5 channel x.

NBIOVx (x = 0-1)

NBI Overview Register (434_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NBI3	NBI3	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI1	NBI1	NBI1	NBI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NBI1	NBI1	NBI1	NBI1	NBI1	NBI1	NBI9	NBI8	NBI7	NBI6	NBI5	NBI4	NBI3	NBI2	NBI1	NBI0
5	4	3	2	1	0										
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NBly (y = 0-31)	y	rh	NBI Flag of Buffer y Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit NBICLRx.NBly. This bit can be set by bit NBISEx.NBly.

Peripheral Sensor Interface (PSI5)
TEI Overview Register

The TEI Overview Register TEIOVx contains flags to manage the receive memory of PSI5 channel x.

TEIOVx (x = 0-1)

TEI Overview Register (448_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEI3	TEI3	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI1	TEI1	TEI1	TEI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEI1	TEI1	TEI1	TEI1	TEI1	TEI1	TEI9	TEI8	TEI7	TEI6	TEI5	TEI4	TEI3	TEI2	TEI1	TEI0
5	4	3	2	1	0										
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
TEI_y (y = 0-31)	y	rh	TEI Flag of Buffer y Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit TEICLRx.TEIy. This bit can be set by bit TEISETx.TEIy.

Peripheral Sensor Interface (PSI5)
CRCI Overview Register

The CRCI Overview Register CRCIOVx contains flags to manage the receive memory of PSI5 channel x.

CRCIOVx (x = 0-1)

CRCI Overview Register $(45C_H+x*4)$ **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
CRCI_y (y = 0-31)	y	rh	CRCI Flag of Buffer y Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit CRCICLRx.CRCI _y . This bit can be set by bit CRCISETx.CRCI _y .

Peripheral Sensor Interface (PSI5)
RDI Overview Register

The RDI Overview Register RDIOVx contains flags to manage the receive memory of PSI5 channel x.

RDIOVx (x = 0-1)

RDI Overview Register (470_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDI3	RDI3	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI1	RDI1	RDI1	RDI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDI1	RDI1	RDI1	RDI1	RDI1	RDI1	RDI9	RDI8	RDI7	RDI6	RDI5	RDI4	RDI3	RDI2	RDI1	RDI0
5	4	3	2	1	0										
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RDly (y = 0-31)	y	rh	RDI Flag of Buffer y Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit RDICLRx.RDly. This bit can be set by bit RDSETx.RDly.

Peripheral Sensor Interface (PSI5)
NFI Overview Register

The NFI Overview Register NFIOVx contains flags to manage the receive memory of PSI5 channel x.

NFIOVx (x = 0-1)

NFI Overview Register (484_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFI3	NFI3	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI1	NFI1	NFI1	NFI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NFI1	NFI1	NFI1	NFI1	NFI1	NFI1	NFI9	NFI8	NFI7	NFI6	NFI5	NFI4	NFI3	NFI2	NFI1	NFI0
5	4	3	2	1	0										
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
NFly (y = 0-31)	y	rh	NFI Flag of Buffer y Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit NFICLRx.NFly. This bit can be set by bit NFISEx.NFly.

Peripheral Sensor Interface (PSI5)
MEI Overview Register

The MEI Overview Register MEIOVx contains flags to manage the receive memory of PSI5 channel x.

MEIOVx (x = 0-1)

MEI Overview Register (498_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEI3	MEI3	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI1	MEI1	MEI1	MEI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEI1	MEI1	MEI1	MEI1	MEI1	MEI1	MEI9	MEI8	MEI7	MEI6	MEI5	MEI4	MEI3	MEI2	MEI1	MEI0
5	4	3	2	1	0										
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
MEI_y (y = 0-31)	y	rh	MEI Flag of Buffer y Copied from INTSTATAx at frame end while RFCx.WRP pointed to buffer y. This bit can be cleared by bit MEICLRx.MEI _y . This bit can be set by bit MEISETx.MEI _y .

Peripheral Sensor Interface (PSI5)
RSI Overview Set Register

The RSI Overview Set Register RSIOVx contains bits to set the status flags of RSIOVx.

RSIOVx (x = 0-1)

RSIOVx Set Register $(4AC_H + x * 4)$ **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSI3	RSI3	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI1	RSI1	RSI1	RSI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSI1	RSI1	RSI1	RSI1	RSI1	RSI1	RSI9	RSI8	RSI7	RSI6	RSI5	RSI4	RSI3	RSI2	RSI1	RSI0
5	4	3	2	1	0										
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Field	Bits	Type	Description
RSIy (y = 0-31)	y	w	Set RSI Flag of Buffer y Setting this bit sets bit RSIOVx.RSIy Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
RMI Overview Set Register

The RMI Overview Set Register RMIOVx contains bits to set the status flags of RSIOVx.

RMIOVx (x = 0-1)
RMIOVx Set Register
(4C0_H+x*4)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMI3	RMI3	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI1	RMI1	RMI1	RMI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMI1	RMI1	RMI1	RMI1	RMI1	RMI1	RMI9	RMI8	RMI7	RMI6	RMI5	RMI4	RMI3	RMI2	RMI1	RMI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RMly (y = 0-31)	y	w	Set RMI Flag of Buffer y Setting this bit sets bit RMIOVx.RMly Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
NBI Overview Set Register

The NBI Overview Set Register NBIOVx contains bits to set the status flags of RSIOVx.

NBIOVx (x = 0-1)
NBIOVx Set Register
(4D4_H+x*4)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NBI3	NBI3	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI1	NBI1	NBI1	NBI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NBI1	NBI1	NBI1	NBI1	NBI1	NBI1	NBI9	NBI8	NBI7	NBI6	NBI5	NBI4	NBI3	NBI2	NBI1	NBI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
NBIy (y = 0-31)	y	w	Set NBI Flag of Buffer y Setting this bit sets bit NBIOVx.NBIy Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
TEI Overview Set Register

The TEI Overview Set Register TEISETx contains bits to set the status flags of TEIOVx.

TEISETx (x = 0-1)
TEIOVx Set Register **(4E8_H+x*4)** **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEI3	TEI3	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI1	TEI1	TEI1	TEI1	
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEI1	TEI1	TEI1	TEI1	TEI1	TEI1	TEI9	TEI8	TEI7	TEI6	TEI5	TEI4	TEI3	TEI2	TEI1	TEI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
TEIy (y = 0-31)	y	w	Set TEI Flag of Buffer y Setting this bit sets bit TEIOVx.TEIy Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
CRCI Overview Set Register

The CRCI Overview Set Register CRCISETx contains bits to set the status flags of CRCIOVx.

CRCISETx (x = 0-1)
CRCIOVx Set Register
 $(4FC_H + x * 4)$
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
CRCI_y (y = 0-31)	y	w	Set CRCI Flag of Buffer y Setting this bit sets bit CRCIOVx.CRCI _y Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
RDI Overview Set Register

The RDI Overview Set Register RDISETx contains bits to set the status flags of RDIOVx.

RDISETx (x = 0-1)

RDIOVx Set Register ($510_H + x * 4$) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDI3	RDI3	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI1	RDI1	RDI1	RDI1	
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDI1	RDI1	RDI1	RDI1	RDI1	RDI1	RDI9	RDI8	RDI7	RDI6	RDI5	RDI4	RDI3	RDI2	RDI1	RDI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RDly (y = 0-31)	y	w	Set RDI Flag of Buffer y Setting this bit sets bit RDIOVx.RDly Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
NFI Overview Set Register

The NFI Overview Set Register NFIOVx contains bits to set the status flags of NFIOVx.

NFIOVx (x = 0-1)

NFIOVx Set Register ($524_H + x * 4$) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFI3	NFI3	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI1	NFI1	NFI1	NFI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NFI1	NFI1	NFI1	NFI1	NFI1	NFI1	NFI9	NFI8	NFI7	NFI6	NFI5	NFI4	NFI3	NFI2	NFI1	NFI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
NFly (y = 0-31)	y	w	Set NFI Flag of Buffer y Setting this bit sets bit NFIOVx.NFIy Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
MEI Overview Set Register

The MEI Overview Set Register MEISETx contains bits to set the status flags of MEIOVx.

MEISETx (x = 0-1)

MEIOVx Set Register (538H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEI3	MEI3	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI1	MEI1	MEI1	MEI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEI1	MEI1	MEI1	MEI1	MEI1	MEI1	MEI9	MEI8	MEI7	MEI6	MEI5	MEI4	MEI3	MEI2	MEI1	MEI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
MEIy (y = 0-31)	y	w	Set MEI Flag of Buffer y Setting this bit sets bit MEIOVx.MEIy Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
RSI Overview Clear Register

The RSI Overview Clear Register RSICLRx contains bits to Clear the status flags of RSIOVx.

RSICLRx (x = 0-1)
RSIOVx Clear Register
(54C_H+x*4)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSI3	RSI3	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI2	RSI1	RSI1	RSI1	RSI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSI1	RSI1	RSI1	RSI1	RSI1	RSI1	RSI9	RSI8	RSI7	RSI6	RSI5	RSI4	RSI3	RSI2	RSI1	RSI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RSly (y = 0-31)	y	w	Clear RSI Flag of Buffer y Setting this bit clears bit RSIOVx.RSIy Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
RMI Overview Clear Register

The RMI Overview Clear Register RMICLRx contains bits to Clear the status flags of RMIOVx.

RMICLRx (x = 0-1)

RMIOVx Clear Register (560_H+x*4) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RMI3	RMI3	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI2	RMI1	RMI1	RMI1	RMI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RMI1	RMI1	RMI1	RMI1	RMI1	RMI1	RMI9	RMI8	RMI7	RMI6	RMI5	RMI4	RMI3	RMI2	RMI1	RMI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RMly (y = 0-31)	y	w	Clear RMI Flag of Buffer y Setting this bit clears bit RMIOVx.RMly Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
NBI Overview Clear Register

The NBI Overview Clear Register NBICLR_x contains bits to Clear the status flags of NBIOV_x.

NBICLR_x (x = 0-1)
NBIOV_x Clear Register
(574_H+x*4)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NBI3	NBI3	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI2	NBI1	NBI1	NBI1	NBI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NBI1	NBI1	NBI1	NBI1	NBI1	NBI1	NBI9	NBI8	NBI7	NBI6	NBI5	NBI4	NBI3	NBI2	NBI1	NBI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
NBI_y (y = 0-31)	y	w	Clear NBI Flag of Buffer y Setting this bit clears bit NBIOV _x .NBI _y Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
TEI Overview Clear Register

The TEI Overview Clear Register TEICLR_x contains bits to clear the status flags of TEIOV_x.

TEICLR_x (x = 0-1)

TEIOV_x Clear Register (588_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEI3	TEI3	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI2	TEI1	TEI1	TEI1	TEI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEI1	TEI1	TEI1	TEI1	TEI1	TEI1	TEI9	TEI8	TEI7	TEI6	TEI5	TEI4	TEI3	TEI2	TEI1	TEI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
TEI_y (y = 0-31)	y	w	Clear TEI Flag of Buffer y Setting this bit clears bit TEIOV _x .TEI _y Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
CRCI Overview Clear Register

The CRCI Overview Clear Register CRCICLR_x contains bits to Clear the status flags of CRCIOV_x.

CRCICLR_x (x = 0-1)
CRCIOV_x Clear Register
(59C_H+x*4)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI	CRCI
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
CRCI_y (y = 0-31)	y	w	Clear CRCI Flag of Buffer y Setting this bit clears bit CRCIOV _x .CRCI _y Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
RDI Overview Clear Register

The RDI Overview Clear Register RDICLRx contains bits to clear the status flags of RDIOVx.

RDICLRx (x = 0-1)

RDIOVx Clear Register (5B0_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDI3	RDI3	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI2	RDI1	RDI1	RDI1	RDI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDI1	RDI1	RDI1	RDI1	RDI1	RDI1	RDI9	RDI8	RDI7	RDI6	RDI5	RDI4	RDI3	RDI2	RDI1	RDI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RDly (y = 0-31)	y	w	Clear RDI Flag of Buffer y Setting this bit clears bit RDIOVx.RDly Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
NFI Overview Clear Register

The NFI Overview Clear Register NFICLR_x contains bits to clear the status flags of NFIOV_x.

NFICLR_x (x = 0-1)
NFIOV_x Clear Register
(5C4_H+x*4)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFI3	NFI3	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI2	NFI1	NFI1	NFI1	NFI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NFI1	NFI1	NFI1	NFI1	NFI1	NFI1	NFI9	NFI8	NFI7	NFI6	NFI5	NFI4	NFI3	NFI2	NFI1	NFI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
NFly (y = 0-31)	y	w	Clear NFI Flag of Buffer y Setting this bit clears bit NFIOV _x .NFly Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)
MEI Overview Clear Register

The MEI Overview Clear Register MEICLR_x contains bits to clear the status flags of MEIOV_x.

MEICLR_x (x = 0-1)

MEIOV_x Clear Register (5D8_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MEI3	MEI3	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI2	MEI1	MEI1	MEI1	MEI1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEI1	MEI1	MEI1	MEI1	MEI1	MEI1	MEI9	MEI8	MEI7	MEI6	MEI5	MEI4	MEI3	MEI2	MEI1	MEI0
5	4	3	2	1	0										
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
MEI_y (y = 0-31)	y	w	Clear MEI Flag of Buffer y Setting this bit clears bit MEIOV _x .MEI _y Clearing this bit has no effect. Reading this bit returns always zero.

33.14.6 Sync Pulse Control

The sync pulse generation provides the following functionality:

- periodic sync pulses, driven by a common module time base (PTE is set)
- angle synchronous sync pulses, driven by GTM (ETE is set)
- sync pulses for ECU to Sensor communication (PSI5 V1.3) (EPS is cleared)
- sync pulses for ECU to Sensor communication (PSI5 enhanced) (EPS is set)
- sync pulses provided by GTM (Bypass BYP set)
- Blank Out (switch off receiver during and after the Sync Pulse)

It is possible to switch on and off the periodic trigger or the angle synchronous trigger at any time. If data is available (TOF is set) the pulses will be modulated according to the selected standard. EPS selects between extended standard with pulse length modulation and legacy mode according to PSI5 standard V1.3 with pulse dropping. The minimum pulse length is defined by PGC.PLEN for both standards. For the extended standard, the pulses are prolonged by PGC.DEL if a one is coded. Exception is the bypass path (BYP is set). These pulses are transmitted without any modulation.

Any combination of PTE, ETE and BYP can produce too long pulses!

Peripheral Sensor Interface (PSI5)

An ECU to Sensor communication can be stopped at any point in time by setting the flush bit. This will clear the send register and flag TOF.

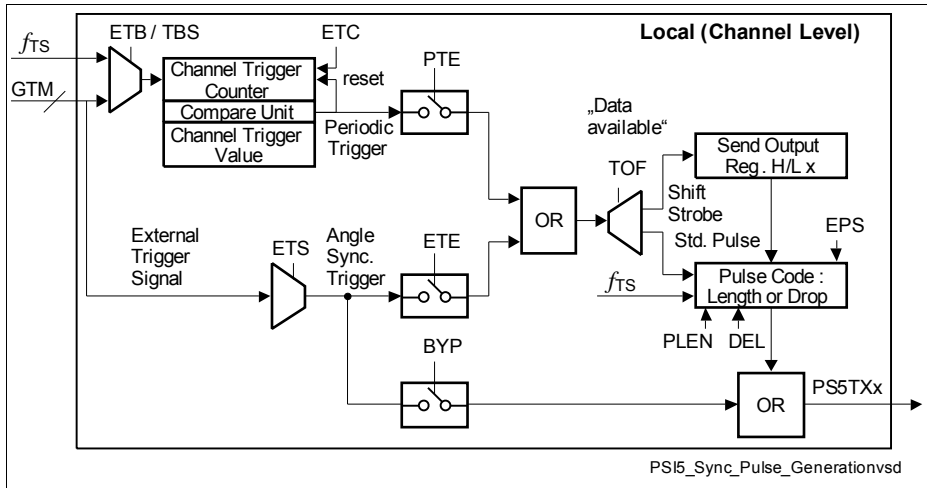


Figure 33-20 PSI5 Sync Pulse Generation

Blank Out Time

BOT defines the length of the blank out time in TTS times (defined by FDRT) starting from the end of a Sync Pulse. The receiver is always switched off starting from the beginning of a Sync Pulse. BOT keeps the receiver switched off additionally after the Sync Pulse. This suppresses potential noise on the receive line after the pulse and thus eases the design of the transceiver.

Peripheral Sensor Interface (PSI5)

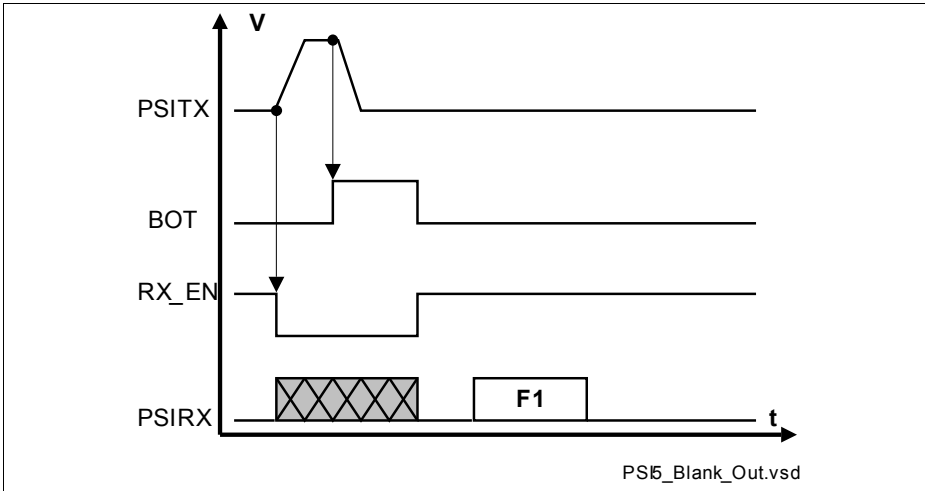


Figure 33-21 PSI5 Blank Out Time

Pulse Generation Control Registers PGCx

The Pulse Generation Control Register PGC contains control data for the sync pulse generation. It contains as well the trigger control bits required for sending data from the PSI5 module to the sensor / external PSI5 device.

PGCx (x = 0-1)

Pulse Generation Control Register x(088_H+x*36*4) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BOT				BYP		ETE	ETS			PTE	ETB				
rw				rw		rw	rw			rw	rw				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBS	0	DEL				0		PLEN							
rw	r	rw				r		rw							

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
PLEN	[5:0]	rw	Pulse Length Defines the length of the pulse in TTS times (defined by FDRT). This is the standard pulse width without data coding into the pulse width or for coding a '0'. 000000 _B Pulse length is 0 TTS 000001 _B Pulse length is 1 TTS 111111 _B Pulse length is 63 TTS
DEL	[13:8]	rw	Delay Length In case data is coded into the pulse length, this defines the ADDITIONAL length of the pulse in TTS times. The resulting sum length is the pulse width for coding a '1' into the pulse width. 000000 _B Pulse length is 0 TTS 000001 _B Pulse length is 1 TTS 111111 _B Pulse length is 63 TTS
TBS	15	rw	Time Base Select This bit selects the clock source for CTVx 0 _B Internal, CTV counts in clock cycles of f_{TS} 1 _B External, CTV counts in clock cycles of f_{GTM}
ETB	[18:16]	rw	External Time Base Select Selects the external clock line for counter CTVx. 000 _B TRIG0 001 _B TRIG1 ... _B ... 101 _B TRIG5 110 _B reserved, no trigger selected 111 _B reserved, no trigger selected
PTE	19	rw	Periodic Trigger Enable Periodic trigger is defined by CTVx. Should be 0 if ETE or BYP is set. 0 _B disabled 1 _B enabled

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
ETS	[22:20]	rw	External Trigger Select Selects the external trigger line for pulse generation (e.g. angle synchronous). 000 _B TRIG0 001 _B TRIG1 ... _B ... 101 _B TRIG5 110 _B reserved, no trigger selected 111 _B reserved, no trigger selected
ETE	23	rw	External Trigger Enable “Angle sync. trigger”, external line is selected by ETS. Should be 0 if PTE or BYP is set. 0 _B disabled 1 _B enabled
BYP	24	rw	Bypass Enable An external signal, selected by ETS directly drives the output of channel x. Should be 0 if PTE or ETE is set. 0 _B disabled 1 _B enabled
BOT	[31:25]	rw	Blank Out Time BOT defines the length of the blank out time in TTS times (defined by FDRT) starting from the end of a Sync Pulse. The receiver is always switched off starting from the beginning of a Sync Pulse. BOT keeps the receiver switched off additionally after the Sync Pulse. This suppresses potential noise on the receive line after the pulse and thus eases the design of the transceiver. 0000000 _B Blank Out Time is 0 TTS 0000001 _B Blank Out Time is 1 TTS 1111111 _B Blank Out Time is 127 TTS
0	14, [7:6]	r	Reserved Read as 0; should be written with 0.

Channel Trigger Value Register x CTVx

The Channel Trigger Value Register x contains a two cell reset counter. It allows for setting up periodic sync pulses for channel x.

The Channel Trigger Value CTV is the compare value which clears the counter CTC on

Peripheral Sensor Interface (PSI5)

equality and only on equality. At exactly this time, a sync pulse is triggered. The compare on equality allows to preset CTC and thus configure an offset between the CTVx.CTCs. This allows for staggering periodic sync pulses that must have frequencies with common multiples (e.g. same frequency, double etc.).

In this document OFFSET = 65535 - "CTC preset value"

I.e. CTC preset is > CTV so that the the time to to roll over is the offset. Of course CTC preset can be < CTV so that this CTC has some advance. Note that CTC can be written only if it is disabled by clearing GCR.ETCx.

Staggering the pulses might be required if the current source for the external phys can provide only limited current so that not all channels can be provided with the additional current that is required to apply the voltage increase representing a proper sync pulse. For a synchronization of all counters CTVx.CTC, bits GCR.ETC0 .. 1 can be used.

Peripheral Sensor Interface (PSI5)

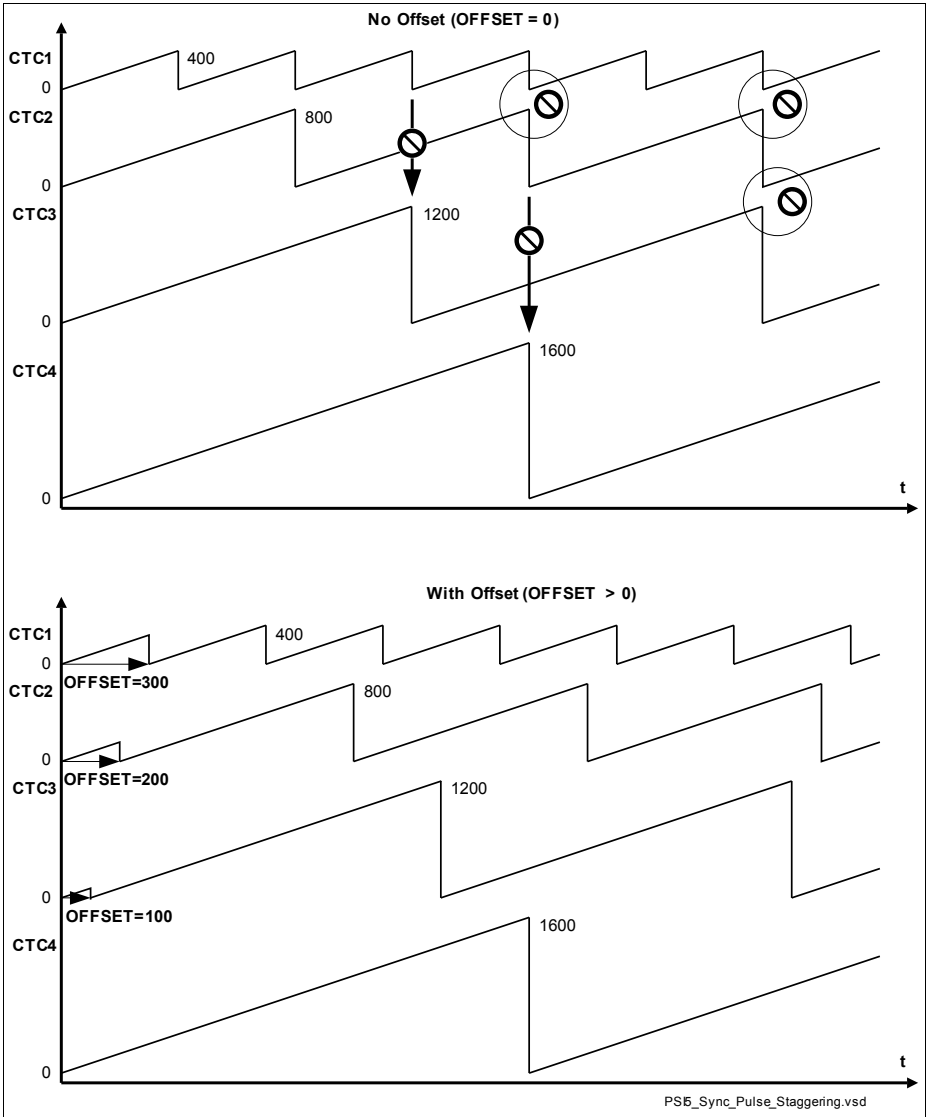
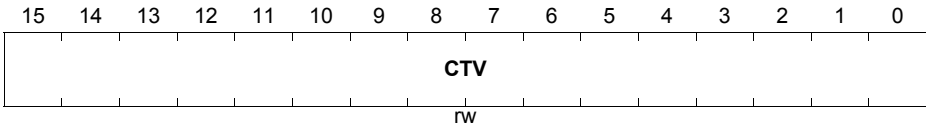
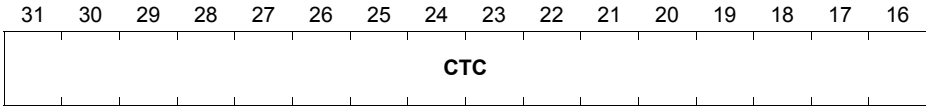


Figure 33-22 PSI5 Pulse Staggering

Peripheral Sensor Interface (PSI5)
CTV_x (x = 0-1)
Channel Trigger Value Register x (08C_H+x*36*4)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CTV	[15:0]	rw	Channel Trigger Value CTV Contains the compare value (exact match) of Channel Trigger CTC at which a sync pulse is triggered for channel x and the counter CTC is cleared. If cleared, CTC is stopped and no pulse is generated.
CTC	[31:16]	rw	Channel Trigger Counter This bit field allows to read the current counter value of the reset timer cell CTVx. If GCR.ETCx is cleared, CTC can be written.

Peripheral Sensor Interface (PSI5)

Send Data Preparation

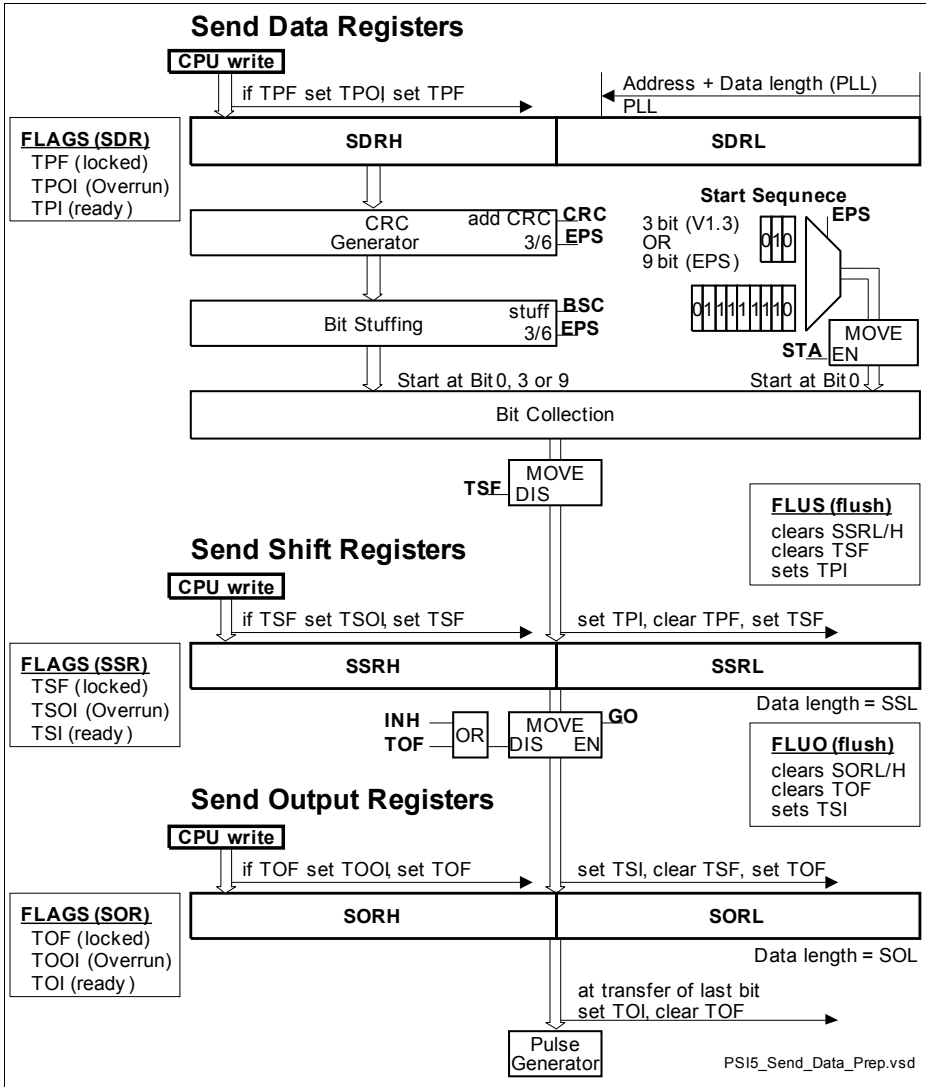


Figure 33-23 PSI5 Send Data Preparation

Peripheral Sensor Interface (PSI5)
Send Control Registers SCR_x

The Send Control Register SCR contains control data bits required for sending data from the PSI5 module to the sensor / external PSI5 device.

Data is collected in SDRL/H_x and the control bits are collected in SCR_x.

The send control unit is build to transmit complete ECU to sensor frames.

SCR_x (x = 0-1)

Send Control Register x (090_H+x*36*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRQ	0	TOF	TSF	TPF	GO	INH	STA	CRC	SOL						
r	r	r	r	r	w	rw	rw	rw	rw						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLU O	FLU S	SSL					BSC	EPS	PLL						
w	w	rw					rw	rw	rw						

Field	Bits	Type	Description
PLL	[5:0]	rw	Pay Load Length of Registers SDRL/H Defines the length that is taken into account: 000000 _B length is 1 000001 _B length is 2 ... 111111 _B length is 64 If PLL > 31, SDRH needs to be written to trigger the HW for automatic STA, BSC or CRC generation or just moving data to SSRL/H. Else writing to SDRL is sufficient. PLL needs to be written before SDRL/H is used for proper operation. If insertion of STA, BSC and CRC results in more than 64 bits, the MSBs are truncated in SSR.
EPS	6	rw	Enhanced Protocol Selection 0 _B PSI5 V1.3. No Pulse length extension 1 _B Enhanced Protocol with Pulse length extension.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
BSC	7	rw	Bit Stuffing Control 0 _B No automatic bit stuffing 1 _B Automatic bit stuffing is enabled. Depending from bit EPS after 3 bits a '1' is inserted (V1.3) or after 6 bits a '0' is inserted (enhanced Power Train Mode) Note that this makes the frame longer.
SSL	[13:8]	rw	Pay Load Length of Registers SSRL/H Defines the length that is taken into account. Start Sequence, BSC and CRC need to be added to PLL by SW if SSL is calculated based on PLL. 000000 _B length is 1 000001 _B length is 2 111111 _B length is 64 If SSL > 31, SSRH needs to be written to trigger the HW for automatic moving data to SORL/H. Else writing to SSRL is sufficient. SSL needs to be written before SSRL/H is used for proper operation.
FLUS	14	w	Flush SSRH/Lx Setting this bit stops the shifting process and flushes (clears) SSRH/Lx and TSF. TPIx is issued at the end of successful flushing. Reads always as zero.
FLUO	15	w	Flush SORH/Lx Setting this bit stops the sending process and flushes (clears) SORH/Lx and TOF. TSIx is issued at the end of successful flushing. Reads always as zero.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
SOL	[21:16]	rw	<p>Pay Load Length of Registers SORL/H</p> <p>Defines the length that is taken into account. Start Sequence, CRC and Stuffing needs to be added by SW to PLL if SOL is calculated based on PLL.</p> <p>000000_B length is 1 000001_B length is 2 ... 111111_B length is 64</p> <p>If SOL > 31, SORH needs to be written to trigger the HW for automatic moving data to the pulse generator. Else writing to SORL is sufficient. SOL needs to be written before SORL/H is used for proper operation.</p>
CRC	22	rw	<p>CRC Generation Control</p> <p>0_B CRC is not generated automatically. 1_B CRC is automatically generated and added to the frame by HW (according to EPS). Note that this makes the frame longer. CRC calculation is not performed, if at least 1 CRC bit does not fit into 64 bit, e.g. due to STA or BSC. In this case, one or two zero bits are inserted.</p>
STA	23	rw	<p>Start Sequence Generation Control</p> <p>0_B no start sequence generated, payload starts at bit 0! 1_B automatically generated and added to the frame by HW (according to EPS) payload starts at bit 3/9 (EPS = 0/1). Note that this makes the frame longer.</p>

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
INH	24	rw	<p>Inhibit Transfer</p> <p>This inhibits the automatic transfer from the shift registers SSRL/H to SORL/H after the preparation of all automatic bit fields (STA, BSC, CRC). After a write access to SDRL/Hx, the automatic preparation is done by HW if selected. If automatic preparation of the start sequence is not selected, data is moved by HW without change to SSRL/H.</p> <p>When the HW is done with the transfer to the internal shift registers, interrupt TPI is activated, flag TSF is set and flag TPF is cleared automatically. Depending on INH, GO must be set after TPI before transfer to SORL/H takes place: After the transfer to SORL/H, TSI is activated, TOF is set and and TSF is cleared automatically. .</p> <p>INH defines the start mode:</p> <p>0_B automatic transfer 1_B GO bit must be set before transfer</p>
GO	25	w	<p>Release prepared Send data</p> <p>This is only relevant if INH is set It allows for manual control of the transfer from SSRH/L to SORL/H. It is always read as zero and is automatically cleared.</p> <p>0_B no transfer 1_B start transfer</p>
TPF	26	r	<p>Transmit Preparation Flag</p> <p>If set, preparation is in progress: start sequence and/ or CRC and/or stuffing and/or at least copying data from SDRL/H. If set, write access to SDRL/H will not change any data and issue TPOI. It is cleared automatically after preparation is finished.</p>

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
TSF	27	r	Transmit Shift Flag If set, data in SSRL/H is waiting to be shifted to SORL/H. This can be caused by an automatic transfer from SDRL/H or from a CPU write access to SSRL/H. If $SSL < 32$ it is sufficient to write SSRL. Else, only writing SSRH will set TSF. If set, write access to SSRL/H will not change any data and issue TSOI. It is cleared automatically after data was shifted to SORL/H or after flushing the register by setting bit FLUS.
TOF	28	r	Transmit Output Flag If set, data in SORL/H is waiting to be transmitted to the sensor. This can be caused by an automatic transfer from SSRL/H or from a CPU write access to SORL/H. If $SOL < 32$ it is sufficient to write SORL. Else, only writing SDRH will set TSF. If set, write access to SORL/H will not change any data and issue TOOI. It is cleared automatically after data was transmitted to the sensor or after flushing the register by setting bit FLUO.
TRQ	31	r	Transfer Request in Progress While a transfer is being sent this bit is set. Write access is ignored.
0	[30:29]	r	Reserved Read as 0; should be written with 0.

Send Data Registers SDRLx

The Send Data Register SDRLx for channel x shows the data content of a data frame to be sent.

Data to be sent must be written while the channel is enabled (GCR.CEN is set). While CEN is cleared the HW can not detect the write access as all state machines are stopped.

Peripheral Sensor Interface (PSI5)
SDRLx (x = 0-1)
Send Data Register Low x **(094_H+x*36*4)** **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD3	SD3	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD1	SD1	SD1	SD1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD1	SD1	SD1	SD1	SD1	SD1	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
5	4	3	2	1	0										
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SDy (y = 0-31)	y	rw	SDy Send data of next ECU to Sensor frame. The unused MSBs (bit position SCRx.PLL and higher) must be written with '0'. If PLL is < 32, SDRHx must not be written.

Send Data Registers SDRHx

The Send Data Register SDRHx for channel x shows the data content of a data frame to be sent.

SDRHx (x = 0-1)
Send Data Register High x **(098_H+x*36*4)** **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD6	SD6	SD6	SD6	SD5	SD5	SD5	SD5	SD5	SD5	SD5	SD5	SD5	SD5	SD4	SD4
3	2	1	0	9	8	7	6	5	4	3	2	1	0	9	8
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD4	SD4	SD4	SD4	SD4	SD4	SD4	SD4	SD3	SD3	SD3	SD3	SD3	SD3	SD3	SD3
7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
SDy (y = 63-32)	y-32	rw	SDy Send data of next ECU to Sensor frame. The unused MSBs (bit position SCRx.PLL and higher) must be written with '0'. If PLL is < 32, SDRHx must not be written.

Send Shift Registers SSRLx

The Send Shift Register SSRLx for channel x shows the data that will be shifted to the Send Output Register SORL. It contains as well the stuff bits as specified in PSI5 V1.3 or in the enhancement proposal of PSI5 standard as of July 2010. During transmission, all bits are shifted to LSB with each sync pulse. MSB is filled up with zero.

SSRLx (x = 0-1)

Send Shift Register Low x (09C_H+x*36*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD3	SD3	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD1	SD1	SD1	SD1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD1	SD1	SD1	SD1	SD1	SD1	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
5	4	3	2	1	0										
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SDy (y = 0-31)	y	rw	SDy Send data of next ECU to Sensor frame. The sequence is: STS, AD, SD, CRC. Depending on individual field length, the bit stream is continued in SSRHx.

Send Shift Registers SSRHx

Continues the bit stream as described in SSRLx.

Peripheral Sensor Interface (PSI5)
SSRHx (x = 0-1)
Send Shift Register High x (0A0_H+x*36*4) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD6	SD6	SD6	SD6	SD5	SD5	SD5	SD5	SD5	SD5	SD5	SD5	SD5	SD5	SD4	SD4
3	2	1	0	9	8	7	6	5	4	3	2	1	0	9	8
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD4	SD4	SD4	SD4	SD4	SD4	SD4	SD4	SD3	SD3	SD3	SD3	SD3	SD3	SD3	SD3
7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SDy (y = 63-32)	y-32	rw	SDy Continues the bit stream as described in SSR _L x.

Send Output Registers SORLx

The Send Output Register SORLx for channel x shows the data that will be shifted to the pulse generator. During transmission, all bits are shifted to LSB with each sync pulse. MSB is filled up from SORH bit 0, while the MSB of SORH is filled up with zeros.

SORLx (x = 0-1)
Send Output Register Low x (0A4_H+x*36*4) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD3	SD3	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD2	SD1	SD1	SD1	SD1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD1	SD1	SD1	SD1	SD1	SD1	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
5	4	3	2	1	0										
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
SDy (y = 0-31)	y	rw	SDy Send data of next ECU to Sensor frame. The sequence is: STS, AD, SD, CRC. Depending on individual field length, the bit stream is continued in SORHx.

Send Output Registers SORHx

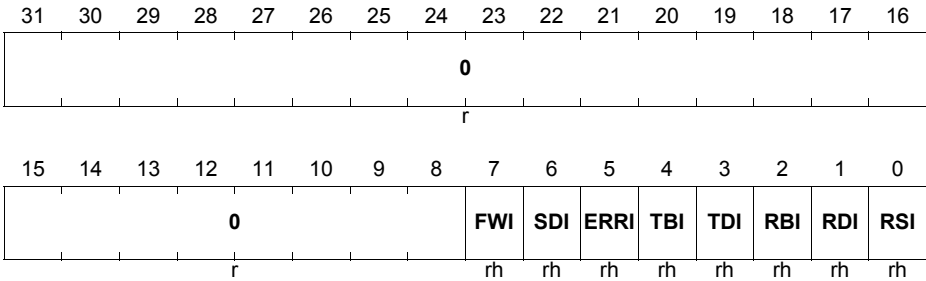
Continues the bit stream as described in SORLx.

SORHx (x = 0-1)

Send Output Register High x **(0A8_H+x*36*4)** **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SD6 3	SD6 2	SD6 1	SD6 0	SD5 9	SD5 8	SD5 7	SD5 6	SD5 5	SD5 4	SD5 3	SD5 2	SD5 1	SD5 0	SD4 9	SD4 8
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD4 7	SD4 6	SD4 5	SD4 4	SD4 3	SD4 2	SD4 1	SD4 0	SD3 9	SD3 8	SD3 7	SD3 6	SD3 5	SD3 4	SD3 3	SD3 2
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SDy (y = 63-32)	y-32	rw	SDy Continues the bit stream as described in SORLx.

33.14.7 Interrupt Control Registers
INTOV
Interrupt Overview Register
(2F8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RSI	0	rh	Interrupt Pending on any Node Pointer RSI If any interrupt requested flag is set for any Node Pointer RSI in register INTSTATx AND the referring interrupt is enabled in INTENx then INTOV.RSI is set. It is automatically reset if all flags in INTSTATx are cleared for which the referring interrupt is enabled in INTENx.
RDI	1	rh	Interrupt Pending on any Node Pointer RDI If any interrupt requested flag is set for any Node Pointer RDI in register INTSTATx AND the referring interrupt is enabled in INTENx then INTOV.RDI is set. It is automatically reset if all flags in INTSTATx are cleared for which the referring interrupt is enabled in INTENx.
RBI	2	rh	Interrupt Pending on any Node Pointer RBI If any interrupt requested flag is set for any Node Pointer RBI in register INTSTATx AND the referring interrupt is enabled in INTENx then INTOV.RBI is set. It is automatically reset if all flags in INTSTATx are cleared for which the referring interrupt is enabled in INTENx.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
TDI	3	rh	Interrupt Pending on any Node Pointer TDI If any interrupt requested flag is set for any Node Pointer TDI in register INTSTATx AND the referring interrupt is enabled in INTENx then INTOV.TDI is set. It is automatically reset if all flags in INTSTATx are cleared for which the referring interrupt is enabled in INTENx.
TBI	4	rh	Interrupt Pending on any Node Pointer TBI If any interrupt requested flag is set for any Node Pointer TBI in register INTSTATx AND the referring interrupt is enabled in INTENx then INTOV.TBI is set. It is automatically reset if all flags in INTSTATx are cleared for which the referring interrupt is enabled in INTENx.
ERRI	5	rh	Interrupt Pending on any Node Pointer ERRI If any interrupt requested flag is set for any Node Pointer ERRI in register INTSTATx AND the referring interrupt is enabled in INTENx then INTOV.ERRI is set. It is automatically reset if all flags in INTSTATx are cleared for which the referring interrupt is enabled in INTENx.
SDI	6	rh	Interrupt Pending on any Node Pointer SDI If any interrupt requested flag is set for any Node Pointer SDI in register INTSTATx AND the referring interrupt is enabled in INTENx then INTOV.SDI is set. It is automatically reset if all flags in INTSTATx are cleared for which the referring interrupt is enabled in INTENx.
FWI	7	rh	Interrupt Pending on any Node Pointer FWI If any interrupt requested flag is set for any Node Pointer FWI in register INTSTATx AND the referring interrupt is enabled in INTENx then INTOV.FWI is set. It is automatically reset if all flags in INTSTATx are cleared for which the referring interrupt is enabled in INTENx.
0	[31:8]	r	Reserved Read as 0.

Peripheral Sensor Interface (PSI5)
Interrupt Status Registers

The Interrupt Status Registers INTSTATA/Bx contains status bits that show the status of any interrupt of PSI5 channel x.

The bits are set independently from the referring Interrupt Enable in Register INTENx. Thus they can be used as status bits as well e.g. by a SW based on polling.

INTSTATAx (x = 0-1)

Interrupt Status Register A x (310_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															NFI
															rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOOI	TOI	TSOI	TSI	TPOI	TPI	RMI	RUI	FWI	CRCI	MEI	NBI	TEI	RBI	TDI	RSI
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RSI	0	rh	<p>Receive Success Interrupt Request Flag</p> <p>This bit is set at the successfully received end of a frame. It indicates that this frame is free of the errors NFI, TEI, NBI, MEI, CRCI if selected to be taken into account in register GCR.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.RSI.</p> <p>This bit can be set by bit INTSETAX.RSI.</p> <p>This bit is set independently from INTENAX.</p>

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
RDI	1	rh	<p>Receive Data Interrupt Request Flag</p> <p>RDI is activated when a received frame is moved to a Receive Data Register RDRL/Hx. Both RDI and RSI will be issued together at correct reception.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.RDI. This bit can be set by bit INTSETAX.RDI. This bit is set independently from INTENAX.</p>
RBI	2	rh	<p>Receive Buffer Overflow Interrupt Request Flag</p> <p>This bit is set after a frame has been received while the old one was not read from RDRHx. I.e. the kernel wants to set any of the two interrupts RSI and RDI and finds any of these two interrupts already set. The old data is overwritten by the new data. Thus, RBI can be ignored if the receive memory is used.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit is NOT cleared by reading RDRx. This bit can be cleared by bit INTCLRAX.RBI. This bit can be set by bit INTSETAX.RBI. This bit is set independently from INTENAX.</p>

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
TEI	3	rh	<p>Time Slot Error Interrupt Request Flag</p> <p>In Synchronous Mode (RCRAx.ASYN = 0), this bit is set if the SOF and the EOF of a frame are received in different time slots. The slots are constraint by the watch dog timer (see WDTxy).</p> <p>In Asynchronous Mode (RCRAx.ASYN = 1), this bit is set if the distance between two frames is longer than specified in WDLO.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.TEI.</p> <p>This bit can be set by bit INTSETAX.TEI.</p> <p>This bit is set independently from INTENAX.</p>
NBI	4	rh	<p>Number of Bits Wrong Request Flag</p> <p>This bit is set if the last frame received less bits than expected but no Manchester coding error occurred until S1, S2, M0 and M1 where received.</p> <p>Note that the frame after the error might be completely wrong.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.NBI.</p> <p>This bit can be set by bit INTSETAX.NBI.</p> <p>This bit is set independently from INTENAX.</p>
MEI	5	rh	<p>Error in Message Bits Flag</p> <p>This bit is set if a manchester error occurred in Bit M0 or M1. (Only if configured in RCRBx.MSG)</p> <p>Note that the frame after the error might be completely wrong.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.MEI.</p> <p>This bit can be set by bit INTSETAX.MEI.</p> <p>This bit is set independently from INTENAX.</p>

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
CRCI	6	rh	<p>CRC Error Request Flag</p> <p>This bit is set if the CRC fails. Set as well if CRC can not be calculated: PDL = 0, CRC cut off by Manchester Error, too few bits, BOT, Sync Pulse)</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.CRCI. This bit can be set by bit INTSETAX.CRCI. This bit is set independently from INTENAX.</p>
FWI	7	rh	<p>FIFO Warning Level Request Flag</p> <p>This bit is set after if the configured warning level of the FIFOx was reached. (See RFCx)</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.FWI. This bit can be set by bit INTSETAX.FWI. This bit is set independently from INTENAX.</p>
RUI	8	rh	<p>Receive Memory Underrun Interrupt Request Flag</p> <p>This bit is set after a SPB master reads from FIFO while no new data was available. (See RFCx)</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.RUI. This bit can be set by bit INTSETAX.RUI. This bit is set independently from INTENAX.</p>

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
RMI	9	rh	<p>Receive Memory Overflow Flag</p> <p>This bit is set after a frame has been received while RDIOVx.RDI[RFCx.WRP] is set. I.e. the kernel wants to set flag RDIxWRP and finds this flag already set. The old data is overwritten by the new data in the buffer memory.</p> <p>0_B No overflow 1_B Overflow</p> <p>This bit can be cleared by bit INTCLRAX.RMIx. This bit can be set by bit INTSETAX.RMIx. This bit is set independently from INTENAX.</p>
TPI	10	rh	<p>Transfer Preparation Interrupt Request Flag</p> <p>This bit is set after data to be transferred has been moved from SDRL/H to SSRL/H. Thus a new value can be written to SDRL/Hx and the prepared data can be checked and modified by SW.</p> <p>0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit is automatically cleared by writing SDRx. This is an exception! It allows for DMA transfers without CPU activity. This bit can be cleared by bit INTCLRAX.TDI. This bit can be set by bit INTSETAX.TDI. This bit is set independently from INTENAX.</p>
TPOI	11	rh	<p>Transmit Preparation Overflow Interrupt Request Flag</p> <p>This bit is set after if SDRL/H is written while TPF is set. The old data is NOT overwritten.</p> <p>0_B No interrupt was requested since this bit was cleared the last time 1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.TPOI. This bit can be set by bit INTSETAX.TPOI. This bit is set independently from INTENAX.</p>

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
TSI	12	rh	<p>Transfer Shift Interrupt Request Flag</p> <p>This bit is set after data to be transferred has been moved from SSRL/H to SORL/H. Thus a new value can be written to SSRL/Hx.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit is NOT cleared by writing SSRL/Hx. This bit can be cleared by bit INTCLRAX.TSI. This bit can be set by bit INTSETAX.TSI. This bit is set independently from INTENAX.</p>
TSOI	13	rh	<p>Transmit Shift Overflow Interrupt Request Flag</p> <p>This bit is set after if SSRL/H is written while TSF is set.</p> <p>The old data is NOT overwritten.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.TSOI. This bit can be set by bit INTSETAX.TSOI. This bit is set independently from INTENAX.</p>
TOI	14	rh	<p>Transfer Output Interrupt Request Flag</p> <p>This bit is set after last bit to be transferred has been moved from SORL/H to the pulse generator. Thus a new value can be written to SORL/Hx.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit is NOT cleared by writing SORL/Hx. This bit can be cleared by bit INTCLRAX.TOI. This bit can be set by bit INTSETAX.TOI. This bit is set independently from INTENAX.</p>

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
TOOI	15	rh	<p>Transmit Shift Overflow Interrupt Request Flag</p> <p>This bit is set after if SORL/H is written while TOF is set. The old data is NOT overwritten.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.TOOI. This bit can be set by bit INTSETAX.TOOI. This bit is set independently from INTENAX.</p>
NFI	16	rh	<p>No Frame Received Interrupt Flag</p> <p>This bit is set at the end of a slot if neither an SOF nor an EOF was received in this slot. If only the two start bits are received and no further data, this “SOF” is discarded completely and treated like no frame received as well. If the “SOF only frame” crosses the WDL, NFI occurs after WDL and EOF detection. If RCRBx.VBS is set, an empty frame is stored anyhow. In this case RDIX is issued as well. The only valid information is SC, TS (captured at the end of the slot) and this bit (NFI).</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRAX.NFI. This bit can be set by bit INTSETAX.NFI. This bit is set independently from INTENAX.</p>
0	[31:17]	r	<p>Reserved</p> <p>Read as 0.</p>

Peripheral Sensor Interface (PSI5)
INTSTATBx (x = 0-1)
Interrupt Status Register B x
(324_H+x*4)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
0												SCRI	SCRI	SCRI	SCRI	SCRI	SCRI	SOI5	SOI4
												5	4	3	2	1	0	rh	rh
												rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
SOI3	SOI2	SOI1	SOI0	SDI5	SDI4	SDI3	SDI2	SDI1	SDI0	WSI5	WSI4	WSI3	WSI2	WSI1	WSI0				
rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh				

Field	Bits	Type	Description
WSly (y = 0-5)	y	rh	<p>Wrong Serial Protocol Error Request Flag</p> <p>This bit is set if the Messaging bits are configured and do not show a start sequence for more than 18 frames of slot y.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRBx.WSly. This bit can be set by bit INTSETBx.WSly. This bit is set independently from INTENBx.</p>
SDly (y = 0-5)	y+6	rh	<p>Serial Data Receive Interrupt Request Flag</p> <p>This bit is set after all serial data bits have been received via the Messaging bit field of slot y. This does NOT indicate a successful check of the CRC.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRBx.SDly. This bit can be set by bit INTSETBx.SDly. This bit is set independently from INTENBx.</p>

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
SOly (y = 0-5)	y+12	rh	<p>Serial Data Buffer Overrun Interrupt Request Flag</p> <p>This bit is set if the HW wants to set SDly while SDly is still set.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRBx.SOly. This bit can be set by bit INTSETBx.SOly. This bit is set independently from INTENBx.</p>
SCRly (y = 0-5)	y+18	rh	<p>Serial Data CRC Error Request Flag</p> <p>This bit is set if the CRC of the serial message fails. This includes a check of the Messaging bit field for correct 0 values of bit 1 in frames 7, 13 and 18.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRBx.SCRly. This bit can be set by bit INTSETx.SCRly. This bit is set independently from INTENBx.</p>
0	[31:24]	r	<p>Reserved</p> <p>Read as 0.</p>

Peripheral Sensor Interface (PSI5)
Interrupt Set Registers

The Interrupt Set Register INTSETA/Bx contain control bits that trigger an interrupt pulse for any interrupt of PSI5 channel x.

INTSETAx (x = 0-1)

Interrupt Set Register A x (338_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															NFI
r															w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOOI	TOI	TSOI	TSI	TPOI	TPI	RMI	RUI	FWI	CRCI	MEI	NBI	TEI	RBI	RDI	RSI
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RSI	0	w	Set Interrupt Request Flag RSI Setting this bit set bit INTSTATx.RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	Set Interrupt Request Flag RDI Setting this bit set bit INTSTATx.RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	Set Interrupt Request Flag RBI Setting this bit set bit INTSTATx.RBI. Clearing this bit has no effect. Reading this bit returns always zero.
TEI	3	w	Set Interrupt Request Flag TEI Setting this bit set bit INTSTATx.TEI. Clearing this bit has no effect. Reading this bit returns always zero.
NBI	4	w	Set Interrupt Request Flag NBI Setting this bit set bit INTSTATx.NBI. Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
MEI	5	w	Set Interrupt Request Flag MEI Setting this bit set bit INTSTATx.MEI. Clearing this bit has no effect. Reading this bit returns always zero.
CRCI	6	w	Set Interrupt Request Flag CRCI Setting this bit set bit INTSTATx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.
FWI	7	w	Set Interrupt Request Flag FWI Setting this bit set bit INTSTATx.FWI. Clearing this bit has no effect. Reading this bit returns always zero.
RUI	8	w	Set Interrupt Request Flag RUI Setting this bit set bit INTSTATx.RUI. Clearing this bit has no effect. Reading this bit returns always zero.
RMI	9	w	Set Interrupt Request Flag RMI Setting this bit set bit INTSTATx.RMI. Clearing this bit has no effect. Reading this bit returns always zero.
TPI	10	w	Set Interrupt Request Flag TPI Setting this bit set bit INTSTATx.TPI. Clearing this bit has no effect. Reading this bit returns always zero.
TPOI	11	w	Set Interrupt Request Flag TPOI Setting this bit set bit INTSTATx.TPOI. Clearing this bit has no effect. Reading this bit returns always zero.
TSI	12	w	Set Interrupt Request Flag TSI Setting this bit set bit INTSTATx.TSI. Clearing this bit has no effect. Reading this bit returns always zero.
TSOI	13	w	Set Interrupt Request Flag TSOI Setting this bit set bit INTSTATx.TSOI. Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
TOI	14	w	Set Interrupt Request Flag TOI Setting this bit set bit INTSTATx.TOI. Clearing this bit has no effect. Reading this bit returns always zero.
TOOI	15	w	Set Interrupt Request Flag TOOI Setting this bit set bit INTSTATx.TOOI. Clearing this bit has no effect. Reading this bit returns always zero.
NFI	16	w	Set Interrupt Request Flag NFI Setting this bit set bit INTSTATx.NFI. Clearing this bit has no effect. Reading this bit returns always zero.
0	[31:17]	r	Reserved Read as 0; should be written with 0.

INTSETBx (x = 0-1)
Interrupt Set Register B x
(34C_H+x*4)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			0					SCRI	SCRI	SCRI	SCRI	SCRI	SCRI	SOI5	SOI4
								5	4	3	2	1	0		
			r					w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOI3	SOI2	SOI1	SOI0	SDI5	SDI4	SDI3	SDI2	SDI1	SDI0	WSI5	WSI4	WSI3	WSI2	WSI1	WSI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
WSly (y = 0-5)	y	w	Set Interrupt Request Flag WSly Setting this bit set bit INTSTATx.WSly. Clearing this bit has no effect. Reading this bit returns always zero.
SDly (y = 0-5)	y+6	w	Set Interrupt Request Flag SDly Setting this bit set bit INTSTATx.SDly. Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
SOly (y = 0-5)	y+12	w	Set Interrupt Request Flag SOly Setting this bit set bit INTSTATx.SOly. Clearing this bit has no effect. Reading this bit returns always zero.
SCRly (y = 0-5)	y+18	w	Set Interrupt Request Flag SCRly Setting this bit set bit INTSTATx.SCRly. Clearing this bit has no effect. Reading this bit returns always zero.
0	[31:24]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface (PSI5)
Interrupt Clear Register

The Interrupt Clear Register INTCLRA/Bx contain control bits that clear the status of any interrupt of PSI5 channel x.

INTCLRAx (x = 0-1)

Interrupt Clear Register A x (360_H+x*4) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															NFI
															w
r															w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOOI	TOI	TSOI	TSI	TPOI	TPI	RMI	RUI	FWI	CRCI	MEI	NBI	TEI	RBI	RDI	RSI
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
RSI	0	w	Clear Interrupt Request Flag RSI Setting this bit clears bit INTSTATx.RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	Clear Interrupt Request Flag RDI Setting this bit clears bit INTSTATx.RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	Clear Interrupt Request Flag RBI Setting this bit clears bit INTSTATx.RBI. Clearing this bit has no effect. Reading this bit returns always zero.
TEI	3	w	Clear Interrupt Request Flag TEI Setting this bit clears bit INTSTATx.TEI. Clearing this bit has no effect. Reading this bit returns always zero.
NBI	4	w	Clear Interrupt Request Flag NBI Setting this bit clears bit INTSTATx.NBI. Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
MEI	5	w	Clear Interrupt Request Flag MEI Setting this bit clears bit INTSTATx.MEI. Clearing this bit has no effect. Reading this bit returns always zero.
CRCI	6	w	Clear Interrupt Request Flag CRCI Setting this bit clears bit INTSTATx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.
FWI	7	w	Clear Interrupt Request Flag FWI Setting this bit clears bit INTSTATx.FWI. Clearing this bit has no effect. Reading this bit returns always zero.
RUI	8	w	Clear Interrupt Request Flag RUI Setting this bit clears bit INTSTATx.RUI. Clearing this bit has no effect. Reading this bit returns always zero.
RMI	9	w	Clear Interrupt Request Flag RMI Setting this bit clears bit INTSTATx.RMI. Clearing this bit has no effect. Reading this bit returns always zero.
TPI	10	w	Clear Interrupt Request Flag TPI Setting this bit clears bit INTSTATx.TPI. Clearing this bit has no effect. Reading this bit returns always zero.
TPOI	11	w	Clear Interrupt Request Flag TPOI Setting this bit clears bit INTSTATx.TPOI. Clearing this bit has no effect. Reading this bit returns always zero.
TSI	12	w	Clear Interrupt Request Flag TSI Setting this bit clears bit INTSTATx.TSI. Clearing this bit has no effect. Reading this bit returns always zero.
TSOI	13	w	Clear Interrupt Request Flag TSOI Setting this bit clears bit INTSTATx.TSOI. Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
TOI	14	w	Clear Interrupt Request Flag TOI Setting this bit clears bit INTSTATx.TOI. Clearing this bit has no effect. Reading this bit returns always zero.
TOOI	15	w	Clear Interrupt Request Flag TOOI Setting this bit clears bit INTSTATx.TOOI. Clearing this bit has no effect. Reading this bit returns always zero.
NFI	16	w	Clear Interrupt Request Flag NFI Setting this bit clears bit INTSTATx.NFI. Clearing this bit has no effect. Reading this bit returns always zero.
0	[31:17]	r	Reserved Read as 0; should be written with 0.

INTCLRBx (x = 0-1)
Interrupt Clear Register A x (374_H+x*4) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
			0					SCRI	SCRI	SCRI	SCRI	SCRI	SCRI	SOI5	SOI4
								5	4	3	2	1	0		
			r					w	w	w	w	w	w	w	w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOI3	SOI2	SOI1	SOI0	SDI5	SDI4	SDI3	SDI2	SDI1	SDI0	WSI5	WSI4	WSI3	WSI2	WSI1	WSI0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Field	Bits	Type	Description
WSly (y = 0-5)	y	w	Clear Interrupt Request Flag WSly Setting this bit clears bit INTSTATx.WSly. Clearing this bit has no effect. Reading this bit returns always zero.
SDly (y = 0-5)	y+6	w	Clear Interrupt Request Flag SDly Setting this bit clears bit INTSTATx.SDly. Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
SOI_y (y = 0-5)	y+12	w	Clear Interrupt Request Flag SOI_y Setting this bit clears bit INTSTATx.SOI _y . Clearing this bit has no effect. Reading this bit returns always zero.
SCRI_y (y = 0-5)	y+18	w	Clear Interrupt Request Flag SCRI_y Setting this bit clears bit INTSTATx.SCRI _y . Clearing this bit has no effect. Reading this bit returns always zero.
0	[31:24]	r	Reserved Read as 0; should be written with 0.

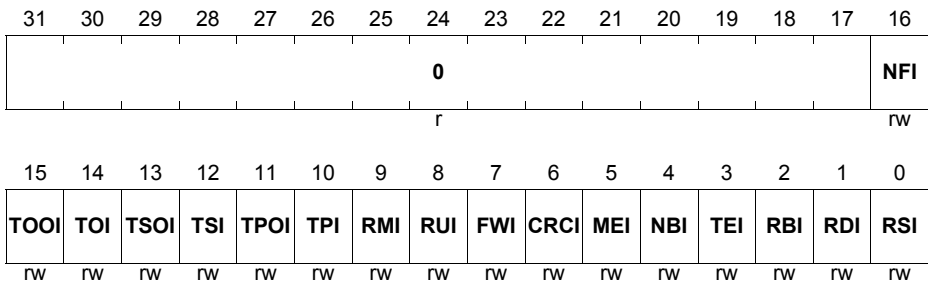
Peripheral Sensor Interface (PSI5)
Interrupt Enable Registers

The Interrupt Enable Register INTENA/B x contain control bits that enable the interrupt source of any interrupt of PSI5 channel x.

The Interrupt Status bits in register INTSTATA/Bx are set independently from the Interrupt Enable in Register INTENA/Bx.

INTENAx (x = 0-1)

Interrupt Enable Register A x (388_H+x*4) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RSI	0	rw	Enable Interrupt Request RSI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
RDI	1	rw	Enable Interrupt Request RDI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
RBI	2	rw	Enable Interrupt Request RBI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
TEI	3	rw	Enable Interrupt Request TEI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
NBI	4	rw	Enable Interrupt Request NBI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
MEI	5	rw	Enable Interrupt Request MEI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
CRCI	6	rw	Enable Interrupt Request CRCI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
FWI	7	rw	Enable Interrupt Request FWI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
RUI	8	rw	Enable Interrupt Request RUI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
RMI	9	rw	Enable Interrupt Request RMII 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
TPI	10	rw	Enable Interrupt Request TPI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
TPOI	11	rw	Enable Interrupt Request TPOI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
TSI	12	rw	Enable Interrupt Request TSI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
TSOI	13	rw	Enable Interrupt Request TSOI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
TOI	14	rw	Enable Interrupt Request TOI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
TOOI	15	rw	Enable Interrupt Request TOOI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
NFI	16	rw	Enable Interrupt Request NFI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
0	[31:17]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface (PSI5)

INTENBx (x = 0-1)
Interrupt Enable Register Bx
(39CH+x*4)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SCRI	SCRI	SCRI	SCRI	SCRI	SCRI	SOI5	SOI4
								5	4	3	2	1	0		
								r	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOI3	SOI2	SOI1	SOI0	SDI5	SDI4	SDI3	SDI2	SDI1	SDI0	WSI5	WSI4	WSI3	WSI2	WSI1	WSI0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
WSIy (y = 0-5)	y	rw	Enable Interrupt Request WSIy 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
SDIy (y = 0-5)	y+6	rw	Enable Interrupt Request SDIy 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
SOIy (y = 0-5)	y+12	rw	Enable Interrupt Request SOIy 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
SCRIy (y = 0-5)	y+18	rw	Enable Interrupt Request SCRIy 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
0	[31:24]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface (PSI5)

Interrupt Node Pointer Register

The Interrupt Node Pointer Register INPx contains the node pointers of PSI5 channel x.

Node Pointer ERRI is one single node pointer for the following interrupts:

- TEI
- NFI
- NBI
- MEI
- CRCI
- RUI
- RMI
- WSly
- SOly
- SCRly

Node Pointer TBI is one single node pointer for the following interrupts:

- TPOI
- TSOI
- TOOI

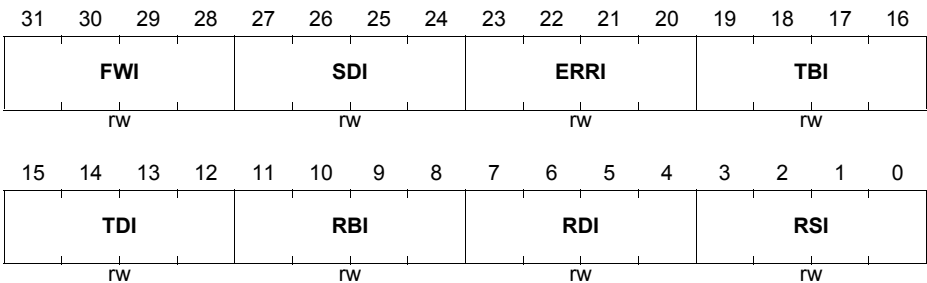
Node Pointer TDI is one single node pointer for the following interrupts:

- TPI
- TSI
- TOI

INPx (x = 0-1)

Interrupt Node Pointer Register x (2FC_H+x*4)

Reset Value: 0000 0000_H



Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
RSI	[3:0]	rw	<p>Interrupt Node Pointer for Interrupt RSI</p> <p>This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATAx.RSI (if enabled by bit INTENAx.RSI).</p> <p>0000_B Trigger Output TRIGO 0 is selected 0001_B Trigger Output TRIGO 1 is selected 0111_B Trigger Output TRIGO 7 is selected 1000_B reserved 1111_B reserved</p>
RDI	[7:4]	rw	<p>Interrupt Node Pointer for Interrupt RDI</p> <p>This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATAx.RDI (if enabled by bit INTENAx.RDI). For bit field definition, see RSI.</p>
RBI	[11:8]	rw	<p>Interrupt Node Pointer for Interrupt RBI</p> <p>This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATAx.RBI (if enabled by bit INTENAx.RBI). For bit field definition, see RSI.</p>
TDI	[15:12]	rw	<p>Interrupt Node Pointer for Interrupt TDI</p> <p>This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATAx.TPI, TSI, TOI (if enabled by bit INTENAx.TPI, TSI, TOI). For bit field definition, see RSI.</p>
TBI	[19:16]	rw	<p>Interrupt Node Pointer for Interrupt TBI</p> <p>This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATAx.TPOI, TSOI, TOOI (if enabled by bit INTENAx.TPOI, TSOI, TOOI). For bit field definition, see RSI.</p>

Peripheral Sensor Interface (PSI5)

Field	Bits	Type	Description
ERRI	[23:20]	rw	Interrupt Node Pointer for Interrupt ERRI This bit field defines the interrupt node, that is requested due to the set condition for bit TEI, NFI, NBI, MEI, CRCl, RUI, RMI, CRCl _y , WSly, SOly, SCRly. For bit field definition, see RSI.
SDI	[27:24]	rw	Interrupt Node Pointer for Interrupt SDI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.SDI (if enabled by bit INTENx.SDI). For bit field definition, see RSI.
FWI	[31:28]	rw	Interrupt Node Pointer for FWI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.FWI. For bit field definition, see RSI.

33.15 PSI5 Module Implementation

This section describes the PSI5 module interface as it is implemented in the TC27x. It especially covers clock control, port and on-chip connections, interrupt control, and address decoding.

33.15.1 Interface Connections of the PSI5 Module

Figure 33-24 shows the TC26x B-Step-specific implementation details and interconnections of the PSI5 module.

The PSI5 module is supplied with a separate clock control, address decoding, and interrupt control logic. The 8 modules' service request outputs are connected with the interrupt router. Outputs of the GTM module are connected to the timer inputs: 6 configured.

The serial data inputs PSIRXn of the receive channels of the PSI5 module as well as the sender outputs PSITXn are connected to GPIO lines. If the outputs are used, they are mapped to a different port pin than the referring PSI5 data input line as most phys will use separate TX and RX lines.

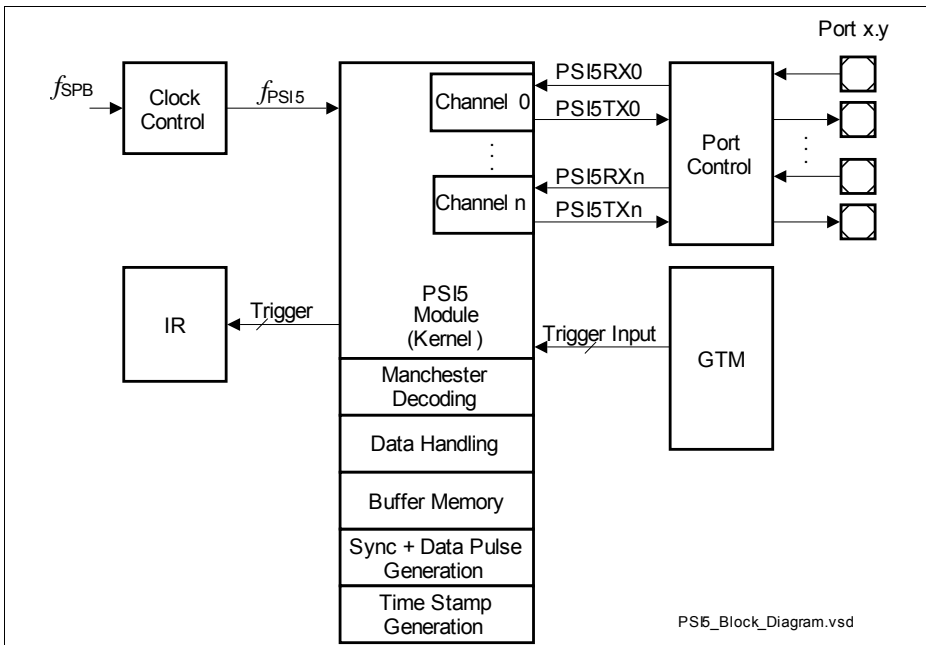


Figure 33-24 PSI5 Module Implementation and Interconnections

33.15.1.1 On-Chip Connections

This section describes the on-chip connections of the PSI5 module.

Interrupt Router Service Requests

The module has 8 Request Trigger outputs connecting it to the interrupt router. The request lines are connected to the IR controller as shown in [Table 33-4](#).

Table 33-4 Service Request Lines of PSI5

INP value	Request Line	Connected to	Description
0000b	TRIGO0	PSI5_SRC0	PSI5 Service Request Node 0
0001b	TRIGO1	PSI5_SRC1	PSI5 Service Request Node 1
0010b	TRIGO2	PSI5_SRC2	PSI5 Service Request Node 2
0011b	TRIGO3	PSI5_SRC3	PSI5 Service Request Node 3
0100b	TRIGO4	PSI5_SRC4	PSI5 Service Request Node 4
0101b	TRIGO5	PSI5_SRC5	PSI5 Service Request Node 5
0110b	TRIGO6	PSI5_SRC6	PSI5 Service Request Node 6
0111b	TRIGO7	PSI5_SRC7	PSI5 Service Request Node 7

Trigger Inputs

The module has 6 trigger inputs which can be randomly chosen by programming IOCRx.ETS, ECS or ETB. The trigger inputs (TRIG[5:0]) of the PSI5 module are connected to the GTM as shown in [Table 33-5](#).

Table 33-5 Trigger Input Lines of PSI5

Request Line	Connected to	Description
TRIG0	gtm.psi5_0_trig_0_o	GTM Trigger Line TRIG0
TRIG1	gtm.psi5_1_trig_0_o	GTM Trigger Line TRIG1
TRIG2	gtm.psi5_2_trig_0_o	GTM Trigger Line TRIG2
TRIG3	gtm.psi5_3_trig_0_o	GTM Trigger Line TRIG3
TRIG4	gtm.psi5_4_trig_0_o	GTM Trigger Line TRIG4
TRIG5	gtm.psi5_5_trig_0_o	GTM Trigger Line TRIG5

33.15.2 PSI5 Module-Related External Registers

The registers listed in [Figure 33-25](#) are not included in the PSI5 module kernel but must be programmed for proper operation of the PSI5 module.

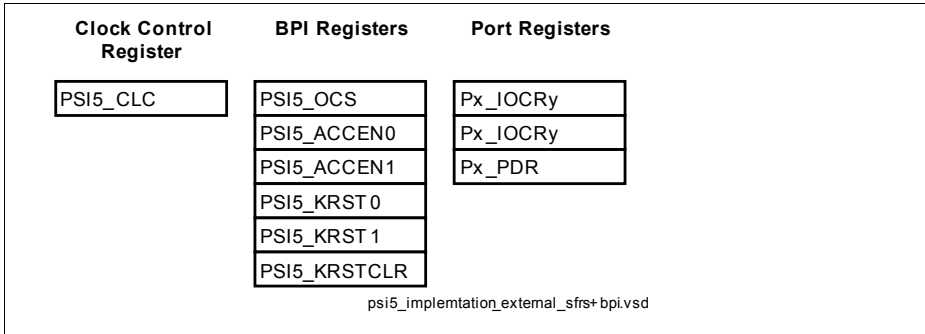


Figure 33-25 PSI5 Implementation-specific Special Function Registers

33.15.2.1 Port Control

The Port Control Registers contain:

- Input/output function selection (Port IOCR registers)
- Pad driver characteristics selection for the outputs (Port PDR registers)

Input/Output Function Selection

[Table 33-6](#) shows an overview how bits and bit fields must be programmed for the required I/O functionality of the PSI5 I/O lines.

The PSI5 interface input shares its input pins with digital ports.

For one channel the PSI5 output pins and PSI5 input pins are next to each other.

Table 33-6 PSI5 I/O Control Selection and Setup

PSI5 Channel	Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
0	P00.1/ PSIRX0A	PSI5_IOCRO.ALT1 = 00 _B	P00_IOCRO.PC1 = 0XXXX _B	I
	P00.2/ PSITX0	not applicable	P00_IOCRO.PC2 = 1X100 _B	O
	P02.3/ PSIRX0B	PSI5_IOCRO.ALT1 = 01 _B	P02_IOCRO.PC3 = 0XXXX _B	I
	P02.2/ PSITX0	not applicable	P02_IOCRO.PC2 = 1X100 _B	O
	P33.1/ PSIRX0C	PSI5_IOCRO.ALT1 = 10 _B	P33_IOCRO.PC1 = 0XXXX _B	I
	P33.2/ PSITX0	not applicable	P33_IOCRO.PC2 = 1X100 _B	O
1	P00.3/ PSIRX1A	PSI5_IOCRO1.ALT1 = 00 _B	P00_IOCRO.PC3 = 0XXXX _B	I
	P00.4/ PSITX1	not applicable	P00_IOCRO4.PC4 = 1X100 _B	O
	P02.5/ PSIRX1B	PSI5_IOCRO1.ALT1 = 01 _B	P02_IOCRO4.PC5 = 0XXXX _B	I
	P02.6/ PSITX1	not applicable	P02_IOCRO4.PC6 = 1X100 _B	O
	P33.3/ PSIRX1C	PSI5_IOCRO1.ALT1 = 10 _B	P33_IOCRO.PC3 = 0XXXX _B	I
	P33.4/ PSITX1	not applicable	P33_IOCRO4.PC4 = 1X100 _B	O

33.15.2.2 Timing Constraints

Delays and jitter inside the module:

- The time from the detection of the last bit of a frame at the input pin of the controller, to the presentation of the data to the software (including CRC calculation) is $\leq 3 \mu\text{s}$.
- The time from the sync pulse trigger output of the Sync Pulse Generator, to the output of the sync pulse trigger at the data output pin of the controller is $\leq 2 \mu\text{s}$.
- Latency between first high slope of first start bit / sync pulse and time stamp generation will be defined after design started, target is a few module clock cycles

Peripheral Sensor Interface (PSI5)

- Jitter between first high slope of first start bit / sync pulse and time stamp generation is calculated in the following way:
 - each timer in the chain adds its jitter given by the time resolution:
 - the TS counter itself has a resolution of nominally 1 μ s
 - the FDRT driving it has a jitter of one input clock phase (this can be avoided by using normal divider mode which should be default)
 - The PLL driving FDRT has a jitter of one input clock phase due to FM PLL clock clipping, this happens at nominally 600 MHz
 - The PLL driving FDRT has an accumulated jitter of +/- 5 ns (as of today, subject to change after characterization)
 - Thus, the jitter is at least $1\mu + 5$ ns

33.16 Revision History

This Internal Target Specification is based on: "PSI5 Specification **V1.3**".

Table 33-7 Revision History

Version Number	Changes to Previous Version
Rev_0.1D	First Draft (Bullet Points ++)
Rev_0.5D	First Draft with content, Pulse Generation tbd, Implementation details tbd Interrupt Flag Overview Registers for Memory Buffers tbd
Rev_0.9	First version for review Addresses, some implementation details and pictures tbd.
Rev_0.10	Updated Sync Pulse Generator block, added figure Updated ECU to Sensor Communication block, now containing 3 stages, added detailed figure Updated Serial Data (slow data) , added receive registers and memory for raw data Updated Manchester decoder and added figure Updated Watch Dog timer block, added 7th timer, added figure Added Time Stamp Generation and selection logic, added figure Cleaned up Interrupts and Triggers Added Port Mapping (full filling the requirements) Addresses, some implementation details and pictures tbd.
Rev_0.11	Added GTM input to Sync Pulse Generator block, updated figure Added Time Stamp configuration logic to support 3 Timers Updated Watch Dog timer block, added capability to check time windows instead of time outs only, updated figure Updated Time Stamp Generation and selection logic, to contain 3 counters (was 2) with individual selection of input clock (GTM or F_ts), was 2 counters with fixed inputs, updated figure Added interrupt "No Frame Received" and the possibility to store this event in the Receive Memory Added RDIOV, NFIOV and MEIOV and the referring set and clear regs. Added Picture of Receive Data Memory INTOV now shows pending interrupts per Node Pointer Addresses, some implementation details and pictures tbd.

Peripheral Sensor Interface (PSI5)
Table 33-7 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.00	<p>Added GCR.ETC0..5 to start/stop sync pulse counters synchronously.</p> <p>Added capability to set an OFFSET between sync pulse counters (CTCs are writable) including figure</p> <p>Removed PGCx.CLRA and CLR for clearing CTCs. Syncing of Trigger Counters CTCs is now done in GCR.ETC0 .. 5.</p> <p>Added Blank Out Time BOT</p> <p>Defined data value for not received frames (NFI)</p> <p>Added note not to set debugger Watch Point to FIFO read register</p> <p>Added capability to select sources to be regarded for RSI (see GCR)</p> <p>Enlarged Watch Dog Counters from 11 to 16 bits</p> <p>Added Asynchronous mode support RCRAx.ASYN</p> <p>Corrected Typos</p> <p>Implemented Address Map</p>
Rev_1.1	<p>Added hint on TPI</p> <p>Added description to RCRA.ASYN: in Asynchronous Mode no Sync Pulses are generated</p> <p>Added description of treatment of frames coming too early (see WDT)</p> <p>Added note that the unused bits in SDRL/H must be written with '0'</p>
Rev_1.2	<p>corrected formulas for f_{125} and f_{189}</p> <p>Added description to FDRT.ECEA/BC: rising edge clears</p> <p>Added description to FDRT.ECS, TSRA/B/C.ETB: "Channel must be disabled if changed (GCR.CEN = 0)."</p> <p>Added clarification to RCRBx.MSGy</p> <p>Kernel Reset Registers are bus master tag protected</p> <p>Note inserted on pulse length (default PLEN for both standards)</p> <p>Note inserted not to use PTE, ETE and BYP together.</p> <p>Changed wording of OIE and IIE</p> <p>Added note to SDR.PLL: If insertion of STA, BSC and CRC results in more than 64 bits, the MSBs are truncated in SSR</p> <p>Changed Bit Filed Name of SDSxz.SCRI</p> <p>Register Overview Table: the Kernel Registers are Master Tag protected ("P").</p> <p>Changed bit order in INTSTATATA to match INTSET/CLRA</p>
Rev_1.3	<p>Detailed bit description of CTS.ACLR and CLR</p> <p>Added CLC.RMC</p> <p>Detailed description of NFI</p>
Rev_1.4	<p>Added 3rd alternate port mapping</p>

Peripheral Sensor Interface (PSI5)
Table 33-7 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.5	Changed to 5 channels for TC29x
Rev_1.6	Added remarks (to CRC, SCRI, MEI, NBI) for clarity, no functional changes. Changed names of bits in INTOV according to their interrupt class representation (for more clarity) into: RSI, RDI, RBI, TDI, TBI, ERRI, SDI, FWI instead of INP[7:0]. Changed access restrictions of IOCR.CFEG/CREG from rw to w.
Rev_1.7	Removed CLC.RMC
Rev_1.8	Updated BE behavior
Rev_1.9	Added RFCx.FRQ and WRAP Note on hard suspend
Rev_1.10	Updated description of OCS
Rev_1.11	Updated description (typo) of SFTSCx Updated Port Chapter
Rev_1.12	Idle time mandatory before looking for start bits, also after BOT and Sync Pulse EOF of a dropped frame will not prevent NFI any more Added Manchester Decoder description from Concept Engineering Slide Set. NBI will be issued not only on too short frames but on too long frames as well for safer detection of wrong sensor assembly.
Rev_1.13	Changed Manchester Decoder sample counter max value from 39 to 41 Removed CR: "NBI will be issued not only on too short frames but on too long frames as well for safer detection of wrong sensor assembly."
Rev_1.14	Support for 2 channels implemented
Rev_1.15	Added pin out description to channel 3 and 4

34 Peripheral Sensor Interface with Serial PHY Connection (PSI5-S)

This document contains the following sections:

- Functional description of the PSI5-S kernel (see [Page 34-2](#))
- PSI5-S kernel register descriptions (see [Page 34-58](#))
- TC27x implementation details (see [Page 34-139](#))

34.1 PSI5-S Description

[Figure 34-1](#) shows a global view of the PSI5-S interface.

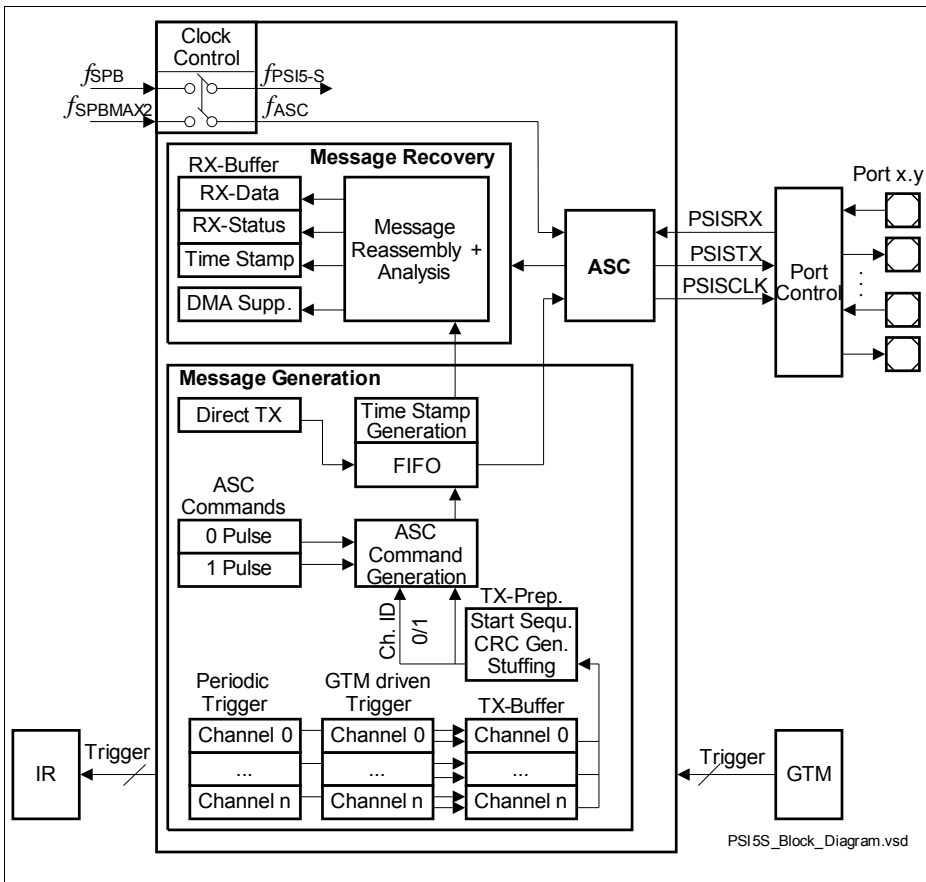


Figure 34-1 General Block Diagram of the PSI5-S Interface

Peripheral Sensor Interface with Serial PHY Connection

The PSI5-S module communicates with the external world via one ASC interface for all channels. This ASC is built into the PSI5-S module. If the optional bidirectional mode is used, the commands to the external Physical Layer Interface (PHY) are transmitted via this ASC.

34.1.1 Overview

The Peripheral Sensor Interface is an interface for automotive sensor applications. PSI5 is an open standard based on existing sensor interfaces for peripheral airbag sensors, already proven in millions of airbag systems.

The PSI5-S IP-module performs communication according to the PSI5 specification V2.0 2011-6.

PSI5 defines a current loop based serial communication link typically used to connect airbag sensors or other peripheral devices.

While the physical layer is done externally, this module manages communication with this PHY and the presentation of the data to the application. Note that there is no on chip PHY, i.e. the current to voltage and voltage to sync pulse translation is done externally.

Receive data on a PSI5-S channel can be set up according to the underlying application. In particular the number of bits forming one value is configured.

The message storage consists of a set of 3 32-bit registers for all channels: one PSI5-S Message Receive Data Register containing the received sensor data, one time stamp register containing the 24-bit value and one additional register containing status bits.

For synchronous communication mode, the module can consume sync pulses from the external GTM or generate periodic sync pulses by itself. It supports as well ECU to sensor communication by offering a 32-bit register, where messages can be set up by the CPU. All kinds of sync pulses are translated into a referring, programmable Byte to be sent via ASC.

The register set of the PSI5-S module can be accessed directly by the CPU for configuration, data read out and status query.

The PSI5-S module supports the following features:

General Features

- ASC based communication with compatible PSI5-S PHY
- Baud rate Generator (up to 12,5) MHz (@ 200 MHz ASC Sub Module Clock)
- Clock out generator to supply external PHY
- Conformance with PSI5 protocol specification V2.0 2011-6
- Data rates of 125 kbit/s and 189 kbit/s supported (by external PHY already)
- 8 PSI5-S channels sharing one common ASC (incl. channel 0 frame 1 special use)
- Supports 6 sensor slots per channel
- Asynchronous and synchronous PSI5 data transmission modes

Peripheral Sensor Interface with Serial PHY Connection

- 8 Interrupt or DMA triggers

Features of Message Recovery Block

- ASC format 10 Bit: 1 Start Bit, 8 Data bits, 1 Stop Bit (Up Stream)
- Each PSI5 Frame is transported in a Packet Frame consisting of 3 to 6 UART Frames transmitted back to back, i.e. with exactly one stop bit - no additional delay.
- Packet Frames are separated by a programmable idle time (1 .. 16 idle bits)
- Assembly and decoding of Packet frames (envelope)
- Routing of PSI5 Frames to their referring channel receive buffer in system memory by hardware DMA support
- Checking CRC of Packet Frames (XCRC)
- Frames with XCRC NOK will be routed to channel 0 frame 1
- The external PHY sends all Frames of an Asynchronous Channel with FID = 1, all frames of this channel have the same length and structure as defined for FID 1. Thus it is sufficient to configure frame 1 for asynchronous channels.
- Configurable data length 8 .. 28 bit + 3 bit PSI5 CRC or 1 bit PSI5 Parity
- CRC check of received PSI5 sensor data, CRC code still transparent
- NO HW Support of extended serial data messaging according to SENT SAE J2716 JAN 2010. Messaging bits are transported fully transparent for SW decoding
- 24-Bit time stamp on sync pulse request via ASC (resolution: 1µs)
- Two independent time bases for Time Stamp: clocked by GTM (1 out of n GTM signals is selectable) or internal periodic trigger generator
- PHY answers to CPU commands with standard frames of fixed predefined length in channel 0 frame 0.
- Error bits in Packet Frame are presented fully transparent and an interrupt is issued.
- One Interrupt for error free Packet Frame, relevant errors are selectable
- One Interrupt for Packet Frame reception disregarding any error.

Features of Message Generation Block

- Support of ECU to Sensor communication
- One 32-bit send data register per channel for downstream data input
- Data is shifted out LSB first, MSB is filled with '0' or '1' with each shift (depending on bit coding method: Tooth Gap or Pulse Width)
- ASC format 11 Bit: 1 Start Bit, 8 Data bits, 1 Parity, 1 Stop Bit (Down Stream)
- Downstream commands of fixed format (3 MSB = Channel ID, 5 LSB = Command)
- All commands are randomly programmable and thus not interpreted by PSI5-S
- Downstream data transmission by 2 different ASC commands (short/no and long pulse)
- 8 common lines of FIFO for all channels, writable by
- PSI5 Message Generation
- Sync Pulse Generation (pre programmed Bytes for '0' and '1')
- CPU Commands

Peripheral Sensor Interface with Serial PHY Connection

- Direct CPU Write Registers allow for insertion of commands into the FIFO for configuration and debugging
- 6 GTM inputs for Sync Pulse Triggering
- SW configurable staggering of Sync Pulse generation
- Generation of 3 or 6 bit CRC for downstream data
- Start sequence generator for downstream data (can be switched off)
- Bit stuffing generator for downstream data (can be switched off)
- CRC generator for downstream data (can be switched off)
- Internal Loop Back mode for check of CRC generator and SW development

Configurability

- No configuration options planned.

34.2 Definitions

SENT: Single Edge Nibble Transmission

CAN: Controller Area Network

LIN: Local Interconnect Network

ISO: International Organization for Standardization

PHY: Physical Layer Interface

ECU: Electronic Control Unit

FSM: Finite State Machine

ASIC: Application Specific Integrated Circuit

Nibble: Four Bit value between 0 and 15 = half a Byte = one character in hex (0 to F)

SOF: Start of Frame

EOF: End of Frame

STS: Start Sequence

AD: Address Data (e.g. of ECU to Sensor Frame)

SD: Send Data (Payload of ECU to Sensor Frame)

CRC: Cyclic Redundancy Check

UART: Universal Asynchronous Receiver Transmitter

34.3 General Operation

PSI5-S is intended for use in applications where high resolution sensor data needs to be communicated from a sensor to an Engine Control Unit (ECU). It is intended as a replacement for the lower resolution methods of 10 bit A/D converters and PWM and as a simpler low cost alternative to CAN or LIN. The implementation assumes that the

Peripheral Sensor Interface with Serial PHY Connection

sensor is a smart sensor containing a microprocessor or dedicated logic device (ASIC) to create the signal.

Figure 34-2 shows a typical TC27x application in which a PSI5-S interface reads a sensor device.

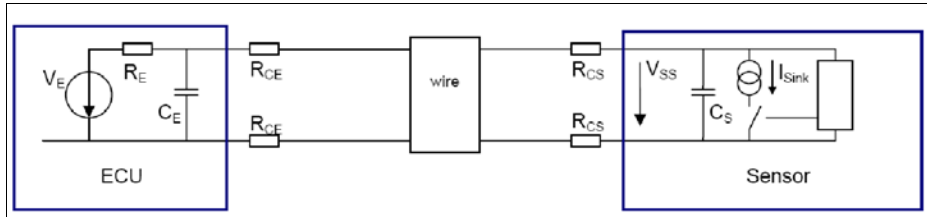


Figure 34-2 PSI5 to External Device Connection

PSI5-S communication is

- asynchronous, unidirectional from sensor to controller without any synchronization
- synchronous, bidirectional with a sync pulse from the ECU triggering messages
- synchronous, bidirectional, sync pulses additionally coding data from ECU to sensor

The sensor signal is transmitted as a series of current pulses in Manchester coding.

The sync pulses are transmitted by increasing the voltage of the current loop.

Figure 34-3 shows a typical TC27x application in which a PSI5 ECU reads multiple PSI sensor devices on a bus.

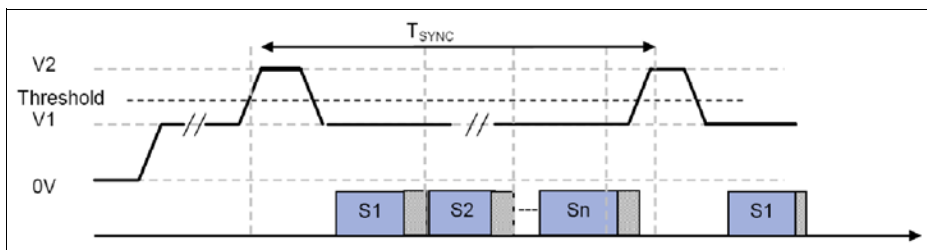


Figure 34-3 PSI5 Frames with Sync pulses

34.4 PSI5 ECU to Sensor Operation

ECU to Sensor communication is defined in PSI5 V2.0 2011-06. Data transmission and configuration of the sensor can be done by modulation of the Sync Pulses.

The Module supports two modes:

- Tooth Gap Method
- Pulse Width Method

Peripheral Sensor Interface with Serial PHY Connection

With the Tooth Gap Method, a specific start condition and a start sequence switch the sensor in ECU to Sensor communication mode. In this mode, the sync pulses are no longer used as triggers for PSI5 Frames but are interpreted as data for the sensor. A pulse represents a 1 and a missing pulse represents a 0. This requires isochronous pulses.

With the Pulse Width Method a standard length pulse represents a 0, a longer pulse a 1. This way continuous data transmission can be achieved. The pulses can be non isochronous, e.g. dependent from the angle position of a rotating axis.

34.5 Frame Formats and Definitions

This section describes the frame formats and definitions of the PSI5-S protocol.

34.5.1 Communication between PSI5-S and PHY via UART

This chapter treats the communication on the UART connection.

34.5.1.1 “Packet Frames” received from PHY

Figure 34-4 and **Figure 34-5** show the layout and definitions of the “Packet Frames”. Packet Frames are 3 to 6 “UART Frames” carrying one “PSI5 Frame”. The format accepted by PSI5-S is explained here. A UART frame is one transmission unit on the ASC. On the receive path it consists of one Start Bit, 8 Data Bits and one Stop bit. Parity can be chosen optionally, e.g. for loop back tests. The figures show the minimum and maximum Packet Frame population with data. Where the figures show ‘Res0’ bits, these bits can be crowded by more data bits if configured so.

Detection of Frame Boundaries and Packet Frame Processing

- SOF (Start of Frame) is defined by a start bit detected after idle on ASC.
- Idle equals at least 2 Stop Bits.
 - The required idle time can be configured between 1 and 16 additional Stop Bits.
 - An idle state signal is provided internally by the ASC to the reassembly unit.
 - The idle signal starts on detection of the first stop bit and ends when the first slope of the next start bit has passed the 2 out of 3 majority vote input filter.
- If idle is detected before the minimum of 3 UART Frames was received, these UART Frames are copied to channel 0 frame 1 (non recoverable frames).
- Based on the ChID and FID received in the first UART frame, the module checks
 - If the referring channel is enabled and
 - If the FID is valid and configured (PDL > 0) and
 - What the number is of expected UART frames configured in UFC
 - If at least this number of UART frames is received without idle and
 - If the XCRC is correct
- If these checks pass
 - EOF is detected after the correct number of UART frames

Peripheral Sensor Interface with Serial PHY Connection

- Data is copied to RDR, interrupts, and the status are updated in RDS. In particular CRC/P check is executed. Time Stamp (TSM) and Target address (TAR) are updated. Depending from bit RCRAx.FIDS the FID might be manipulated.
- The message recovery unit does not wait for idle to execute the steps above.
- If these checks fail (“non recoverable frames”)
 - EOF is detected after idle or a maximum of 6 UART Frames
 - The 1 to 6 UART Frames are stored in channel 0 frame 1.
- If more UART frames follow a passing or failing Packet Frame without idle,
 - EOF is detected after idle or a maximum of 6 UART Frames
 - The 1 to 6 UART Frames are stored in channel 0 frame 1
 - Reception is stopped
 - The next UART Frames following without idle time are ignored
 - Reception starts only after the idle time programmed in GCR.IDT is detected (“bubbling idiot” protection)

Non Recoverable Frame Presentation in RDR and RDS

- RDS.FID and ChID are derived from the first UART Frame received.
- RDR.RD[27:0] contains all bits of each UART Frame received after the header. I.e.
 - 0 bits for 1 UART Frame,
 - 8 bits for 2 UART Frames,
 - 16 bits for 3 UART Frames,
 - 24 bits for 4 UART Frames,
 - 28 bits for 5 and 6 UART Frames.
- RDS.CRC is determined as follows:
 - If 6 UART Frames were received: bits [6:4] of the 5th UART Frame.
 - If 1 .. 5 UART Frames where received: determined by PDL
 - PDL may not be reliable, but this way all data is covered either in the data field or in the XCRC field. In case PDL was correct, CRC is shown correctly.
- RDS.XCRC contains always the bits [7:2] of the last UART Frame received. Also if only 1 UART Frame was received.
- Note that XCRCI is always set as either XCRC was wrong or the number of UART frames received was wrong or it could not be determined.

Loop Back support

- Parity and number of stop bits need to be configured the same for sender and receiver in loop back mode. Parity control and polarity of parity can be configured separately each for TX and RX. This can be done before switching on loop back mode. Alternatively the parity control can be adjusted as well in one single register access together with switching the ASC part to loop back mode.
- Both ways are supported: add Parity bit to the receiver or use the transmitter without parity. (Standard use case is to use transmitter with parity)
- Data to be transmitted can be written directly to register CDW
- This way a synthetic Packet Frame can be sent to the reassembly unit for test purposes.

Peripheral Sensor Interface with Serial PHY Connection

- The received loop back data is treated in the same way as the normal receive data. Good frames are received in their referring channel and slot. Frames with wrong XCRC, idle time or UFC are stored in channel 0 frame 1.

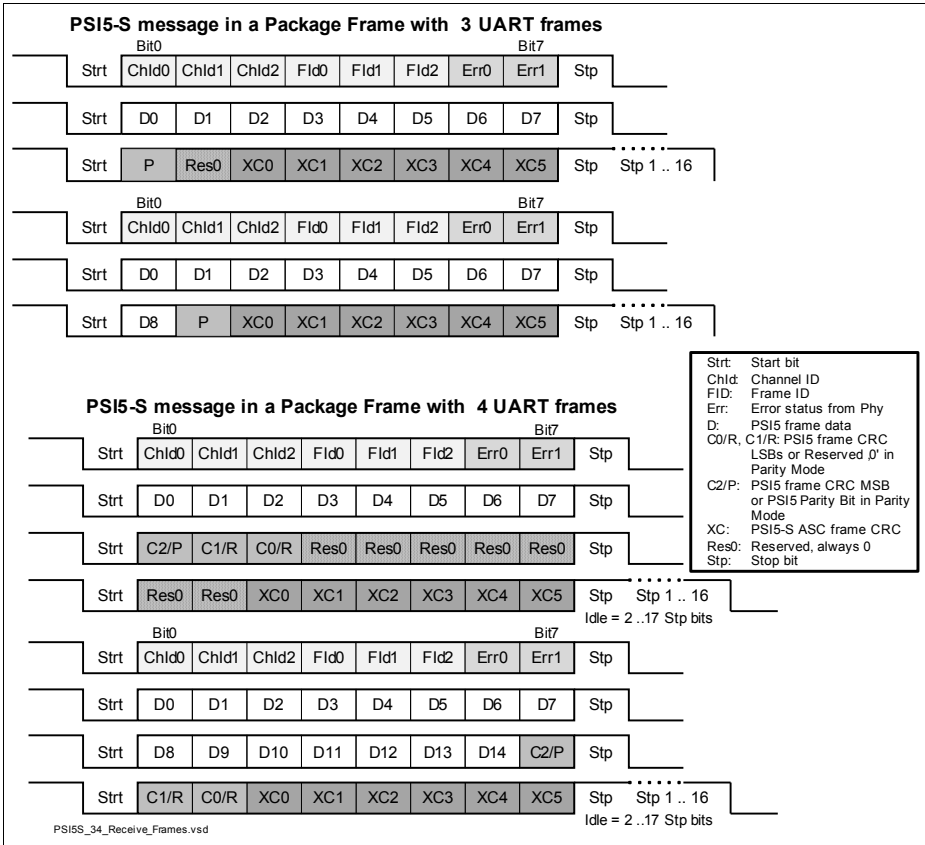


Figure 34-4 PSI5-S 3 + 4 UART Frames per Package Frame received from PHY

Peripheral Sensor Interface with Serial PHY Connection

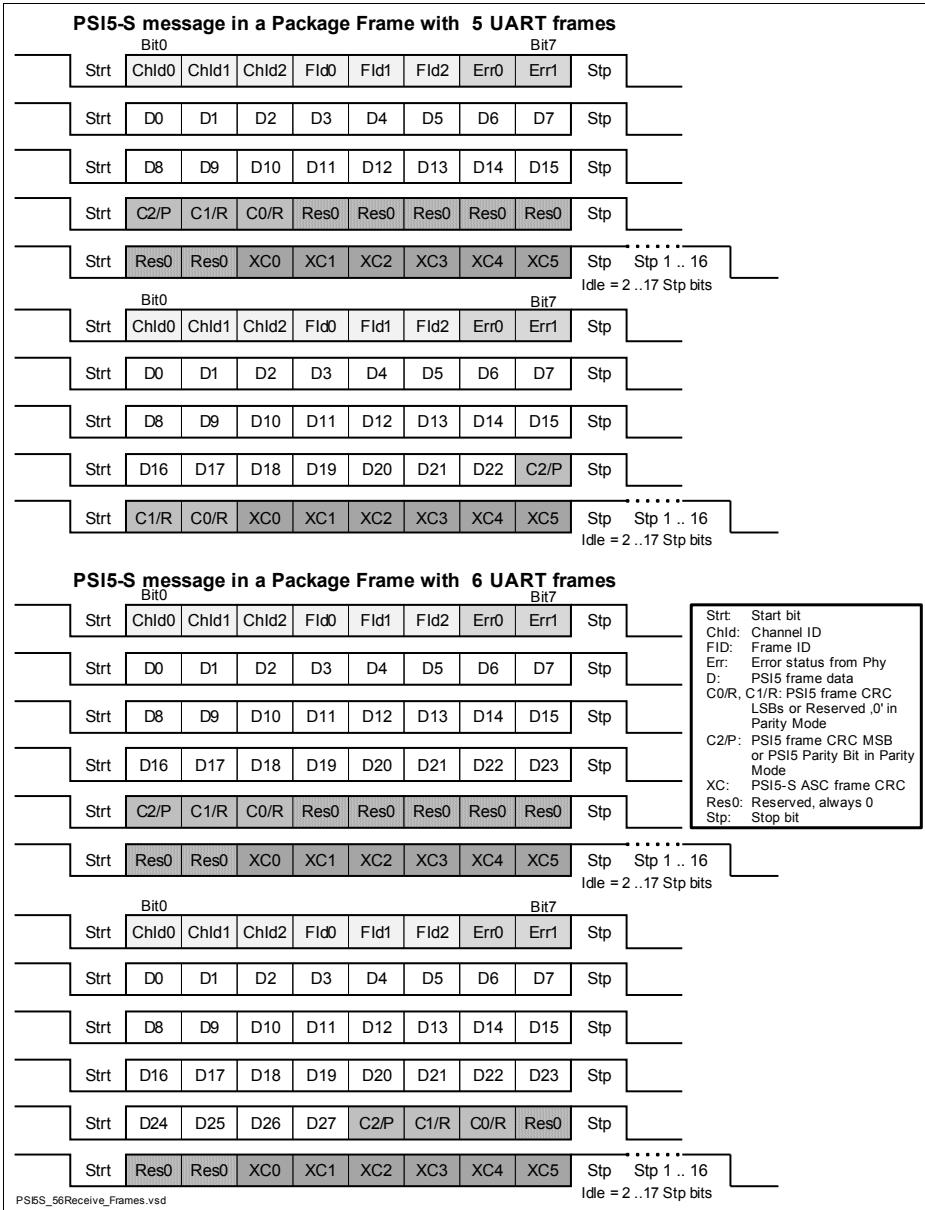


Figure 34-5 PSI5-S 5 + 6 UART Frames per Package Frame received from PHY

Peripheral Sensor Interface with Serial PHY Connection

XCRC Content

XCRC is always sent at the end of the last UART Packet. For XCRC calculation, all bits of the preceding UART Frames that belong to the Packet Frame are relevant including potential stuffing bits (Res0). UART Start / Stop bits are excluded.

In detail the XCRC shall contain the header (ChId, FID, Err0, Err1), PSI5 frame data (D0 .. Dx, PSI5 frame Parity Bit or CRC bits) and potential stuffing bits (Res0) between P/CRC and XCRC. Not included are start and stop bits of the UART packets and the 2 start bits (S0, S1) of the PSI5 frame.

The LSB of the first byte (ChId0) is fed first into the XCRC polynomial.

XCRC Calculation Method

XCRC equals the CRC defined in PSI5 Standard 2.0 “Frame format 4 ECU to Sensor Message CRC”, see **“ECU to Sensor Communication” on Page 34-12**.

34.5.1.2 PSI5-S UART Frames transmitted to PHY

Figure 34-6 shows the layout and definitions of the UART Frames sent by PSI5-S.

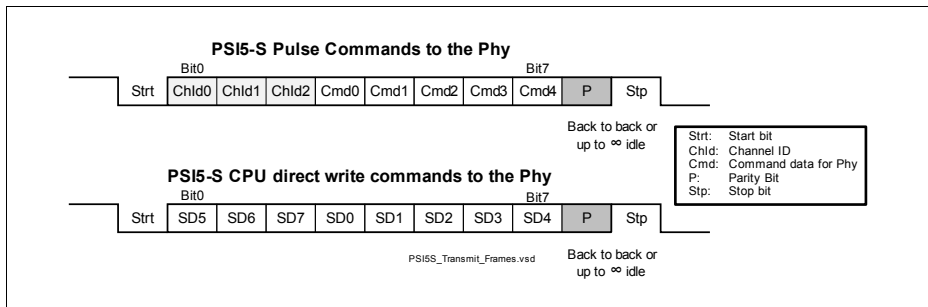


Figure 34-6 PSI5-S UART Frames transmitted to the PHY

34.5.2 Communication between PHY and Sensor (PSI5 Standard)

This chapter treats the frame formats on the PSI5 bus as short reference.

34.5.2.1 PSI5 Standard Frame Format

Figure 34-7 shows the principal layout and definitions of a PSI5 Frame. Note that the PSI5 standard specifies that the least significant bit is sent out first. See standard for CRC and Parity definition.

Peripheral Sensor Interface with Serial PHY Connection

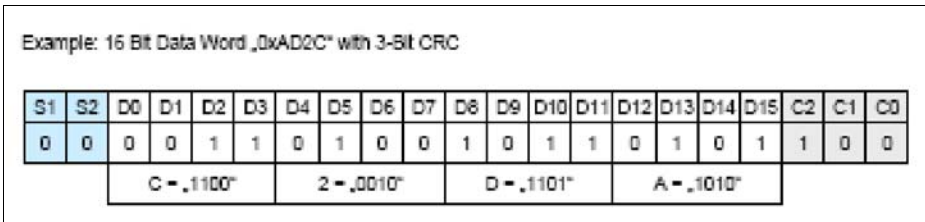


Figure 34-7 Standard PSI5 Frame Format

34.5.2.2 PSI5 Extended Frame Format

Figure 34-8 shows the layout and definitions of an Extended PSI5 Frame. Note that the PSI5 standard specifies that the least significant bit is sent out first.

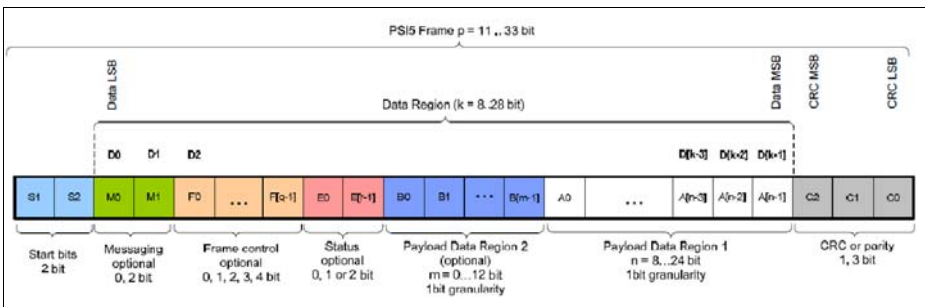


Figure 34-8 PSI5 Extended Frame Format

PSI5 Sensor to ECU Message CRC

The transmission of PSI5 data is checked by the following protection modes. The transmission error detection must be selectable:

- 1) 1-bit even parity
- 2) 3-bit CRC

The applied generator polynomial of the CRC is $g(x) = 1 + x + x^3$ with a binary start value (seed) "111". The transmitter shall extend the data bits by three zeros (as MSBs). This augmented data word shall be fed (LSB first) into the shift registers of the CRC check. Start bits shall be ignored in this check. When the last zero of the augmentation is pending on the input adder, the shift registers contain the CRC checksum. These three check bits shall be transmitted in reverse order (MSB first: C2, C1, C0)

34.5.2.3 Sync Pulses

Sync Pulses are used for two different purposes:

Peripheral Sensor Interface with Serial PHY Connection

- Triggering a data frame for data acquisition from a sensor or
- ECU to sensor communication.

Synchronous Transmission

In the “synchronous” mode, the sensor (slave) starts to transfer a complete data frame only after a sync pulse is transmitted via ASC to the external PHY. The external PHY translates this ASC command into a voltage increase. The sensor then initiates a measurement and starts to calculate the new output data value. The data follows in a standard PSI5 Frame, starting with the 2 start bits, data and Parity or CRC. The timing diagram in [Figure 34-9](#) visualizes a synchronous transmission

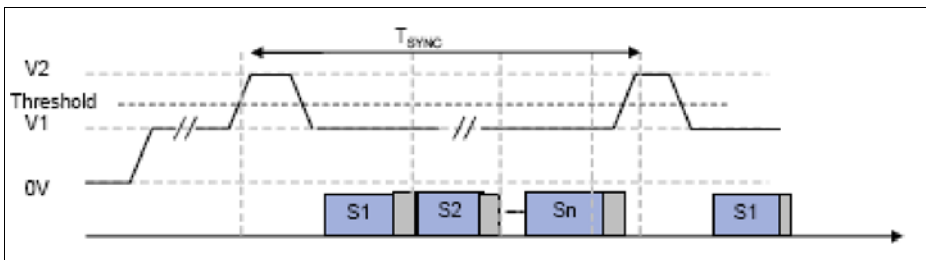


Figure 34-9 Synchronous Transmission

ECU to Sensor Communication

PSI5 V2.0 2011-06 defines 4 principle frame formats. Please refer to this standard document for more details. As the main target application is power train, only frame format 4 is cited here. This is the format for the power train sub standard.

Two methods for the ECU to Sensor Communication are supported:

Tooth Gap Method (see [Figure 34-10](#)) and
Pulse Width Method (see [Figure 34-11](#) and [Figure 34-12](#)).

Tooth Gap Method

uses usually 31 sync pulses as start condition and start bits (0 1 0) to signal the start of the ECU to Sensor Communication. From this start on, the sync pulses are interpreted by the sensor as data. See standard for CRC and Parity definition.

Peripheral Sensor Interface with Serial PHY Connection

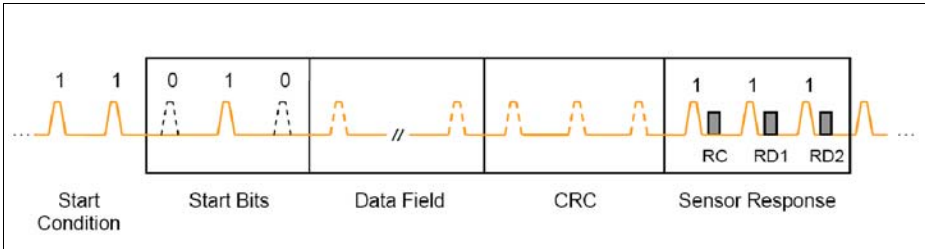


Figure 34-10 ECU to Sensor (Tooth Gap Method)

Pulse Width Method

allows for modulation of the sync pulse duration. This way bidirectional communication is possible continuously.

Sync Bits of frame format 4

A consecutive train of seven ones within the transmitted data could be misinterpreted as a frame start condition. Therefore a “sync bit” (logical 0) is inserted after every six bits of the sensor address and the payload data. With respect to the data transmission, these sync bits are treated like stuffing bits, which are removed at the receiver side.

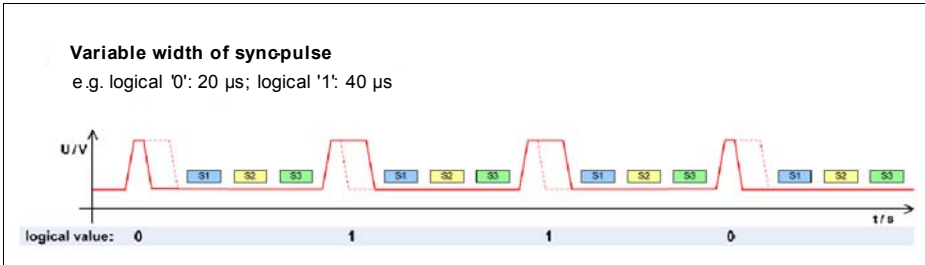


Figure 34-11 ECU to Sensor, Bit transmission (Pulse Width Method)

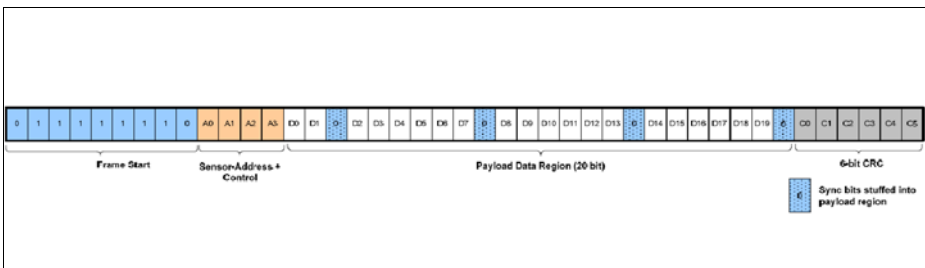


Figure 34-12 ECU to Sensor, Frame format 4 (power train)

Peripheral Sensor Interface with Serial PHY Connection

Frame format 4 ECU to Sensor Message CRC

The ECU-to-Sensor Frame has a 6-bit CRC.

The generator polynomial of the CRC is

$$g(x)=x^6 + x^4 +x^3 +1$$

with a binary CRC initialization value "010101".

The transmitter extends the data bits by six zeros (as MSBs).

This augmented data word shall be fed (LSB first) into the shift registers of the CRC check. Start bits and sync bits are ignored in this check.

When the last zero of the augmentation is pending on the input adder, the shift registers contain the CRC checksum.

These six check bits shall be transmitted LSB first [C0, C1 .. C5]. This CRC value is computed as a function of the contents of all address and data bits and Sync bits ("0" bits). For purposes of the CRC calculation, the bits shall be ordered: $m = [A0 A1 .. A3 D0 D1 0 D2 .. D19 0]$.

In this implementation the number of elements is configurable. While the standard is 4 address and 20 data bits, the module allows longer or shorter messages too.

34.6 Clock Generation

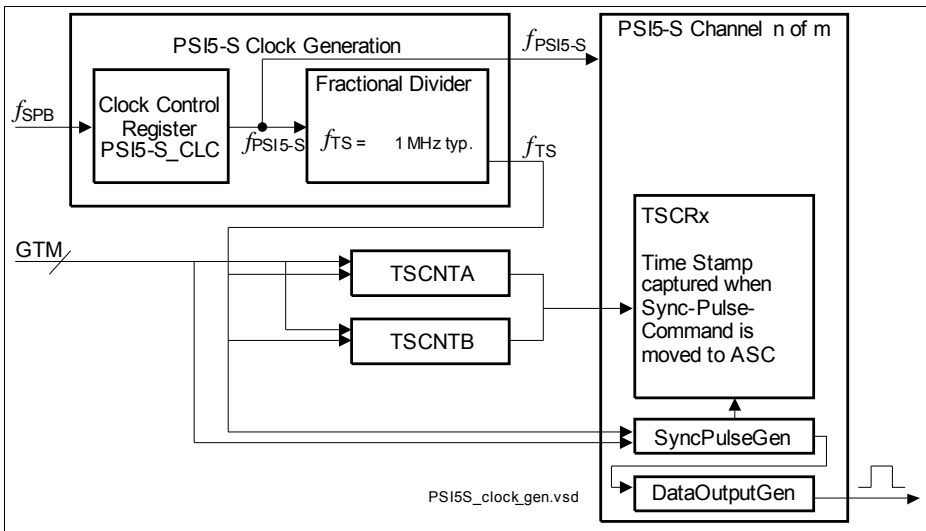
The PSI5-S module provides 2 central Time Stamp Counters and the central Time Stamp Clock f_{TS} to each channel. See [Figure 34-13](#). The clock of each Time Stamp Counter is individually selectable: either f_{TS} or any of the GTM inputs can be selected to drive them.

This chapter shows in detail how f_{TS} is adjusted.

For details on the principles of a fractional divider, please refer to the SCU chapter of TC27x section "Clock Control".

The PSI5-S module provides 2 internal clock signals centrally

Peripheral Sensor Interface with Serial PHY Connection


Figure 34-13 PSI5-S Module Clock Generation

- $f_{\text{PSI5-S}}$
 This is the module clock that is used inside the PSI5-S kernel for control purposes such as clocking of control logic and register operations. The frequency of $f_{\text{PSI5-S}}$ is controlled by register FDR and derived from the system clock frequency f_{SPB} . The clock control register CLC makes it possible to enable/disable $f_{\text{PSI5-S}}$ under certain conditions.
- f_{TS}
 This clock is usually used to drive the central Time Stamp Counters. It can be selected to drive the global sync pulse time base and inside the PSI5-S channels to drive the local sync pulse generators and watch dog timers. The fractional divider register FDRT controls the frequency of f_{TS} . This is usually adjusted to 1 MHz / 1 μ s period time.

The following two formulas define the frequency of f_{TS} :

$$f_{\text{TS}} = f_{\text{PSI5-S}} / (1024 - \text{PSI5-S_FDRT.STEP}); \text{FDRT.DM} = 01\text{B} \quad (34.1)$$

$$f_{\text{TS}} = f_{\text{PSI5-S}} \times \text{PSI5-S_FDRT.STEP} / 1024 \text{ with STEP} = 0 \dots 1023; \text{FDRT.DM} = 10\text{B} \quad (34.2)$$

34.6.1 Overview on Clocks in the System

To reduce system cost, the external PHY can be supplied with a clock from TC27x. EXTCLK1/2 (SCU), CMU_ECK (GTM) or by PSISCLK (PSI5-S). The range is limited to

Peripheral Sensor Interface with Serial PHY Connection

20 .. 25 MHz maximum for proper clock transmission. The PHY needs to provide 3 clocks from this: Sample clock for 189 kHz, and 125 kHz and the ASC oversampling clock.

A divider by 2 is added between the actual fractional divider block and the output signal f_{PSISCLK} in order to achieve a duty cycle of 50% (plus tolerances due to fractional divider jitter)! This is why the formulas contain the division by 2 at the end.

- f_{PSISCLK}
This clock is used to drive the external PHY.
The fractional divider register **FDO** controls the frequency of f_{PSISCLK} .

The following formula defines the frequency of f_{PSISCLK} :

$$f_{\text{PSISCLK}} = f_{\text{BAUD2}} / (2048 - \text{PSI5-S_FDO.STEP}) / 2; \text{DM} = 01\text{B} \tag{34.3}$$

$$f_{\text{PSISCLK}} = f_{\text{BAUD2}} \times \text{PSI5-S_FDO.STEP} / 2048 / 2 \text{ with STEP} = 0 \dots 2047; \text{DM} = 10\text{B} \tag{34.4}$$

In order to relax the requirements with respect to ASC communication with TC27x while allowing for high bit rates, some additional provisions have been made. The ASC Sub Module is supplied with a higher frequency (f_{BAUD2}) to allow for very high precision at high baud rates (4 - 12,5 MBaud). As a second measure to achieve this, the fractional divider in ASC is longer than in the standard module. **Figure 34-14** illustrates the situation on PCB. Refer to the ASC sub chapter for baud rate calculations.

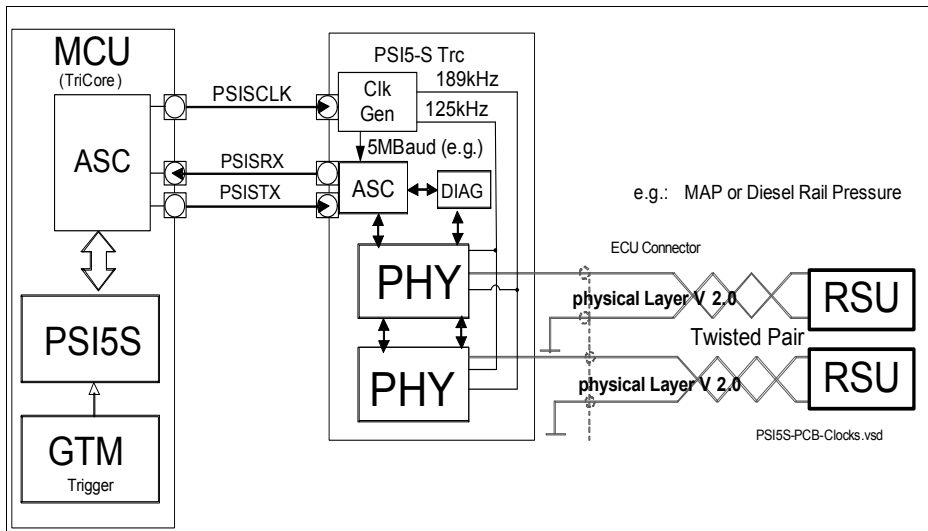


Figure 34-14 PSI5-S Clock Requirements of PHY

Peripheral Sensor Interface with Serial PHY Connection

34.7 Time Stamp Generation

Two time bases can be selected for Time Stamping:

Any of TSCNTA and TSCNTB provide one Time Stamp Counter. Each of them can be configured to be driven by either f_{TS} or by an external timer, i.e. one of the GTM inputs. The frequency of f_{TS} is controllable by the fractional divider in register FDRT.

All of them can be cleared by SW single or synchronously.

The last sync pulse sent on a channel is stored in the sync pulse time stamp capture register TSCRx. Note that if RCRAx.TSTS is set, the Sync Pulses do not trigger the Time stamp capture. In this case, the Packet Frame reception triggers the capture. Non recoverable Packet Frames will be stored with the current value of TSCNTA/B directly copied to TSM. See RCRAx on [Page 34-83](#) and on for more details.

For each channel the capture time (TSCRx) is stored.

Figure 34-15 shows the time stamp generation:

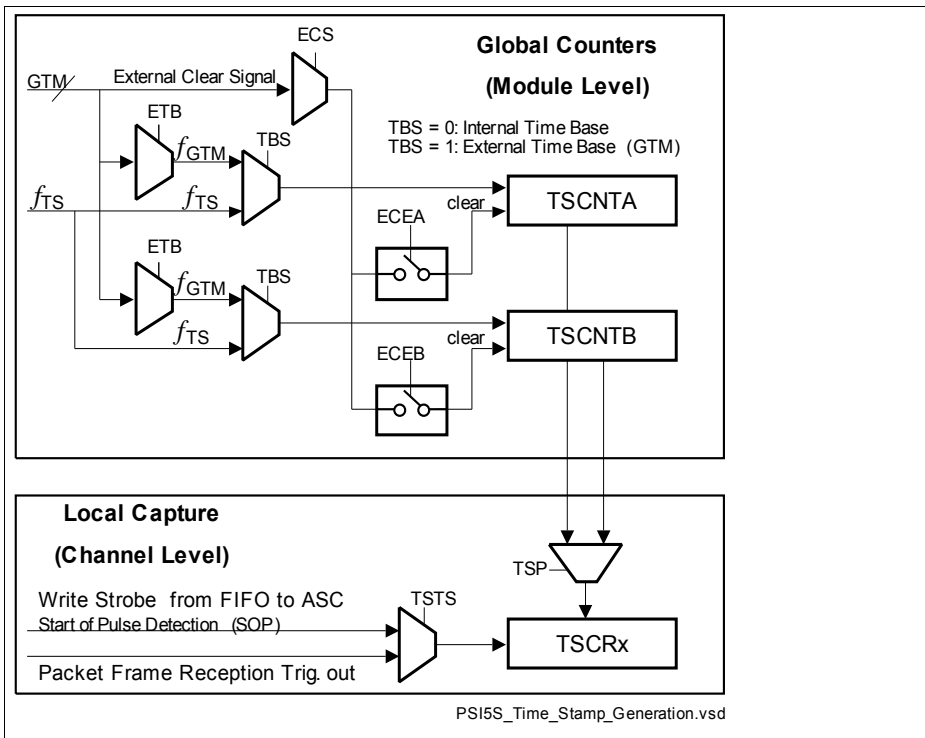


Figure 34-15 Time Stamp Generation

Peripheral Sensor Interface with Serial PHY Connection

34.8 Watch Dog Timers

The value entered in WDTx.WDL defines the time in multiples of TTS (defined in FDRT) until expiry of the watch dog.

There are two modes of operation that can be selected with bit RCRAx.WDMS.

In normal operation an interrupt (TEI) is issued if the distance between two RDIs is longer than specified in WDL. In this mode, The internal watch dog timer is started automatically when the channel is enabled.

In synchronous mode, TEI is issued if the watch dog timer expires without reception of CHCI in time. I.e. the time from issuing the sync pulse to reception of the last expected frame configured in NFC.NFx was too long.

In both modes, only completely and correctly received Packet Frames on channel x can qualify, i.e. Packet Frames with XCRC checked ok. Otherwise the channel number can not be determined - the referring WDLx not be selected.

The internal watch dog timer is compared to WDL. A match

- triggers a TEI.
- restarts the watch dog timer. I.e. it clears the internal watch dog timer and it counts on from zero. TEI is repeated if no recoverable frames are received.

If no watch dog is needed, WDL is cleared, the internal watch dog timer is stopped and no check is performed.

Asynchronous mode of the Watch Dog Timers:

If RCRAx.WDMS is cleared the internal watch dog timer y is restarted on reception of a new recoverable frame (RDIx) for the referring channel. It is not restarted by a sync pulse and not stopped by a certain number of received Packet Frames.

The example below shows the use of the watch dog. The sync pulse is completely ignored by the watch dog system.

F2 is the only frame coming too late. All others start before the expiry of the watch dog.

Peripheral Sensor Interface with Serial PHY Connection

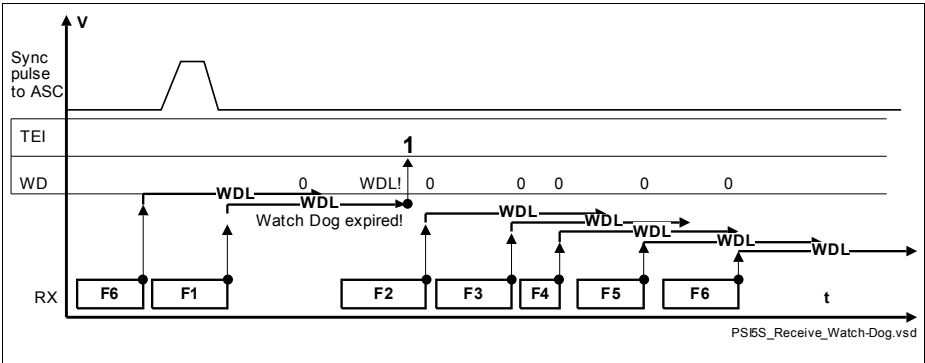


Figure 34-16 Watch Dog Timers

Synchronous mode of the Watch Dog Timers:

If RCRAx.WDMS is set the WDT is restarted on sync pulse and stopped at reception of the last frame configured in NFx. Register NFCx contains the bit fields NFx which configure the number of Packet Frames expected after the sync pulse. FCNT.FCx is the counter that is compared with NFx.

Peripheral Sensor Interface with Serial PHY Connection

The Watch Dog Timer is restarted (cleared and set to run), each time a Sync Pulse is transferred for the referring channel from FIFO to ASC. The Watch Dog Timer of this channel is forced to stop if the number of a received Packet Frames (counted in FCNT.FCx) matches the number configured in NFC.NFx. This supports in particular angle synchronous channels where the distance of packets can vary strongly with e.g. the roundings per minute of an axis.

The WDT can not stop in time if a non recoverable (and thus not countable) Packet Frame is received! If more than the configured number of Packet Frames are received, no dedicated interrupt is issued for this but the frame is still received correctly.

RCRAx.WDMS should be 0 on asynchronous channels. Otherwise, the internal Watch Dog Timer will be stopped after reception of the number of configured frames but never be restarted due to the lack of a Sync Pulse!

If a sensor is not present in a special configuration, NFx must be reduced accordingly! E.g. if out of F1 .. F6 the F2 is missing, only 5 frames are configured.

Please refer to register **"NFC"** on [Page 34-79](#) for detailed description.

Peripheral Sensor Interface with Serial PHY Connection

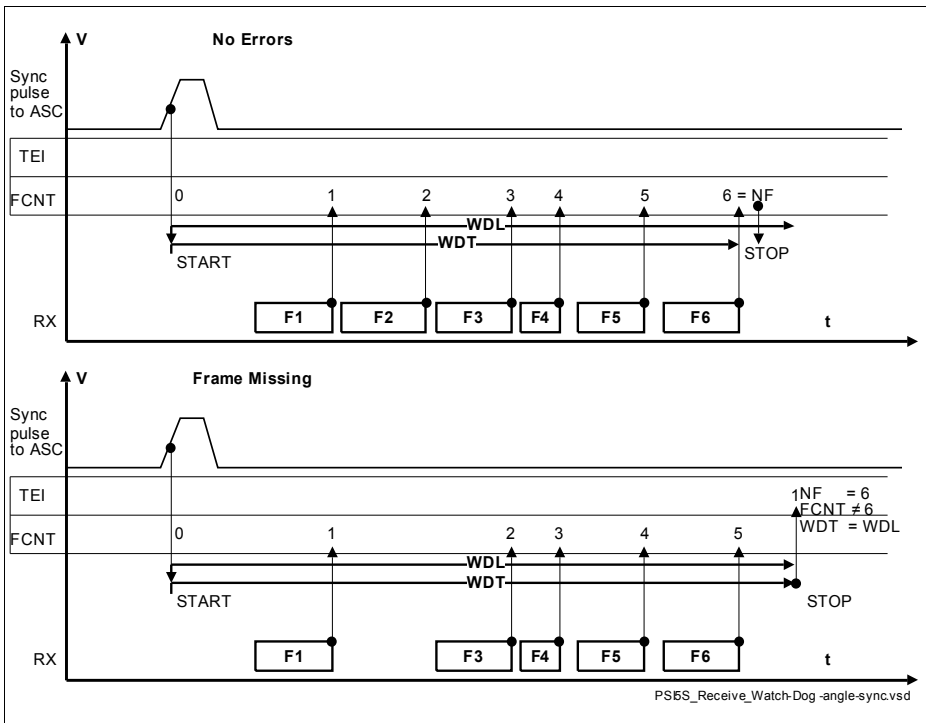


Figure 34-17 Watch Dog Timers in synchronous mode

34.9 Send Data

This chapter contains the details of how data to be sent is prepared. The scope extends to:

- Periodic Triggers
- Angle Synchronous Triggers
- ECU to Sensor communication

34.9.1 Channel Trigger

Peripheral Sensor Interface with Serial PHY Connection

The Channel Trigger Value Register CTVx contains a two cell reset counter. It allows for setting up periodic sync pulses for channel x.

The Channel Trigger Value CTV is the compare value which restarts the counter CTC on equality and only on equality. At exactly this time, a sync pulse is triggered.

The compare on equality allows to preset CTC and thus configure an offset between the CTVx.CTCs. This allows for staggering periodic sync pulses that must have frequencies with common multiples (e.g. same frequency, double etc.).

In this document OFFSET = 65535 - "CTC preset value"

I.e. CTC preset is > CTV so that the the time to to roll over is the offset.

Of course CTC preset can be < CTV so that this CTC has some advance.

Note that CTC can be written only if it is disabled by clearing GCR.ETCx.

Staggering the pulses might be required if the current source for the external PHYs can provide only limited current so that not all channels can be provided with the additional current that is required to apply the voltage increase representing a proper sync pulse.

For a synchronization of all counters CTVx.CTC, bits GCR.ETC0 .. 7 can be used.

Peripheral Sensor Interface with Serial PHY Connection

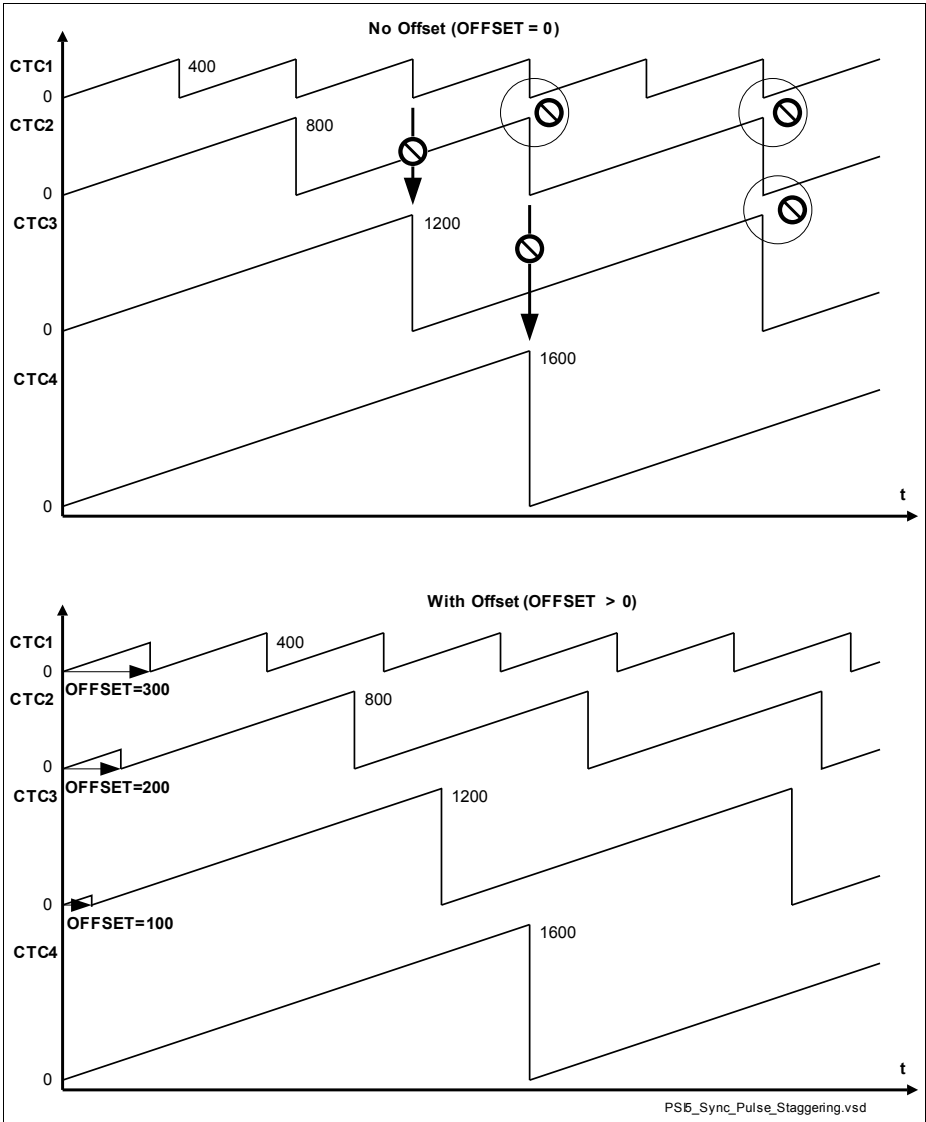


Figure 34-18 PSI5-S Pulse Staggering

Peripheral Sensor Interface with Serial PHY Connection

34.9.2 Sync Pulse Control

The sync pulse generation provides the following functionality:

- periodic sync pulses, driven by a common module time base (PTE is set)
- angle synchronous sync pulses, driven by GTM (ETE is set)
- sync pulses for ECU to Sensor communication (Tooth Gap Method, EPS[0]=0)
- sync pulses for ECU to Sensor communication (PWM Method, EPS[0]=1)

It is possible to switch on and off the periodic trigger or the angle synchronous trigger at any time. The pulses will be modulated according to the selected standard. EPS selects between PWM method and Tooth Gap Method with pulse dropping.

An ECU to Sensor communication can be stopped at any point in time by setting the flush bit. This will clear the send register and flag TPF.

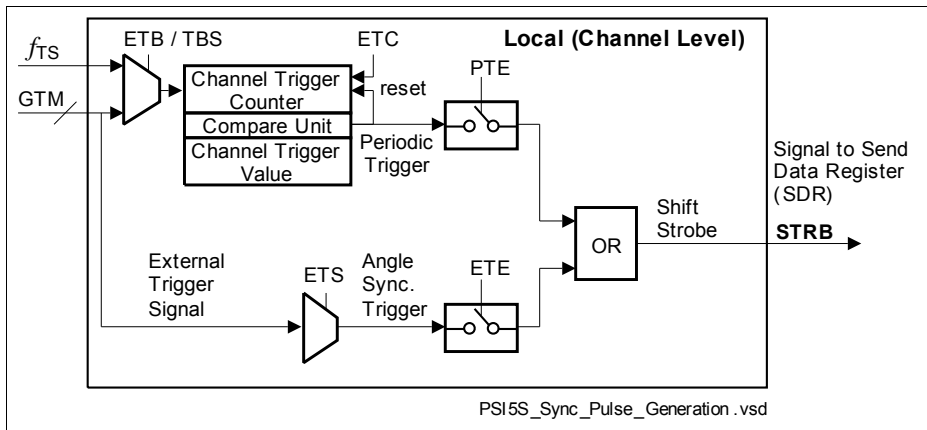


Figure 34-19 PSI5-S Sync Pulse Generation

34.9.3 Send Data Preparation

Figure 34-20 shows the mechanisms how ECU to sensor communication data is treated:

Peripheral Sensor Interface with Serial PHY Connection

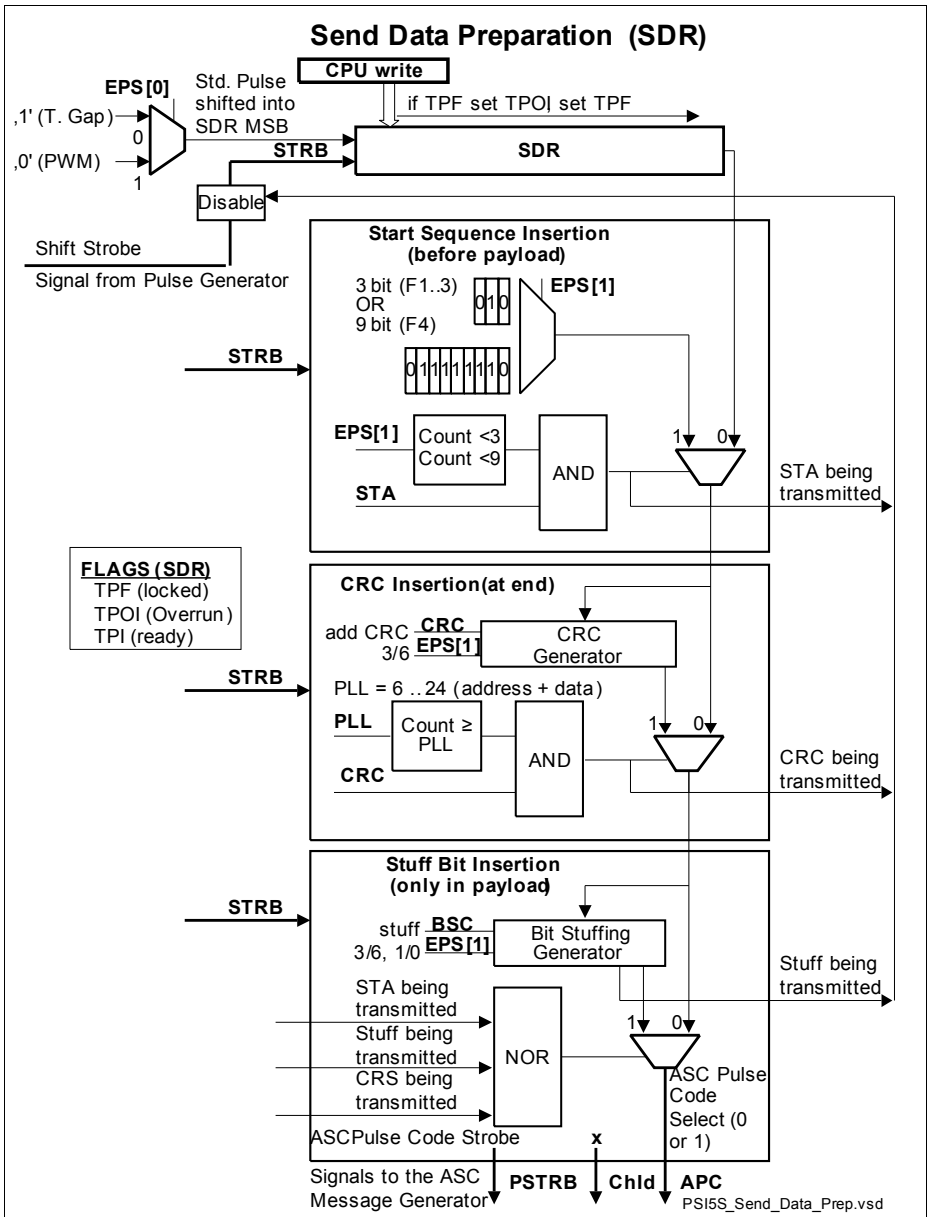


Figure 34-20 PSI5-S Send Data Preparation

Peripheral Sensor Interface with Serial PHY Connection

34.10 Message Generation

Figure 34-21 shows the message generation logic. The generated ASC commands are sorted into an internal FIFO of 8 entries of 9 bits width. The MSB stores the information, if a command is generated from the sync pulse generation (ASC Pulse Code Strobe issued the write into FIFO) or if the CPU wrote a command. This is important for the Time Stamp generation. Usually the Time Stamp is captured when a Sync Pulse Command leaves the FIFO i.e. is copied to the ASC TX buffer. Commands stored in the FIFO can only trigger a Time Stamp capture if they are marked as Sync Pulse in bit 8. Nevertheless, the CPU can set MSB (bit 8) in Register CDW if the command is to be treated like a Sync Pulse Command and a Time Stamp should be captured. Note that if RCRAx.TSTS is set, the Sync Pulses do not trigger the Time stamp capture. In this case, the Packet Frame reception triggers the capture and bit 9 is ignored.

There is no further arbitration of the CPU and Message Generation Unit. The PHY answers CPU commands with standard frames of fixed predefined length in channel 0 frame 0.

Interrupt FOI is issued if a transfer to the FIFO was generated by the message generation unit or by CDW (CPU direct write register) while the FIFO was full. In a correct setup, this will never be the case as the bandwidth of the ASC is assumed to be by far higher than the write bandwidth in order to have short delays. In most cases too many write accesses to CDW will be the root cause. This exceptional condition can be handled either by waiting until the FIFO has transferred at least one command or by a module reset by SW. It should not occur in the application but FOI should help SW generation and debugging.

The transfer path of a command to the ASC could add jitter to sync pulses.

In case several channels request a pulse at exactly the same time, the channel with the lower channel number is served first and CDW has highest priority.

Peripheral Sensor Interface with Serial PHY Connection

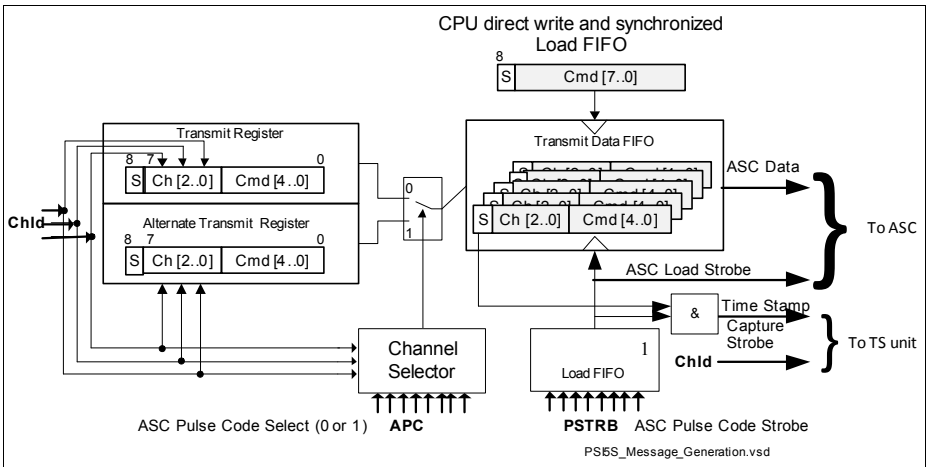


Figure 34-21 PSI5-S Message Generation Logic

34.11 DMA Support

The received frames come in sequentially and are XCRC checked. The complete reassembled frame including Channel Id (ChId), Frame ID (FId) and status are available at RDR and RDS. TSM shows the latest time stamp of the currently received channel.

RDR, RDS and TSM are located in 3 sequentially addressed registers. This way, they are easily readable by DMA. See [Figure 34-22](#).

34.11.1 Single DMA, 8 dedicated DMAs

With one single DMA channel, one common frame buffer in the system memory can be build up. All frames can be collected from RDR, RDS and TSM. The DMA source address is programmed to be always RDS, block transfer is set to 3 words with auto address increment. The target is set to be automatically incremented. This results in one buffer in system memory with all frames from all channels.

Individual buffers for each channel can be build up if more DMA channels are used. Each channel can generate its own receive triggers RDI and RSI. Thus individual DMAs can be set up for each channel similarly to the single DMA use case but with individual target addresses for each channel.

Peripheral Sensor Interface with Serial PHY Connection

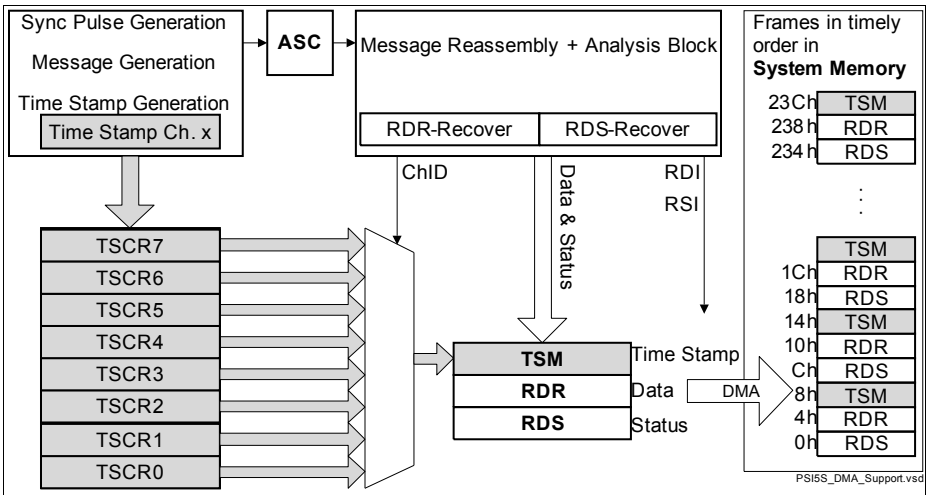


Figure 34-22 PSI5-S Single DMA Support

34.11.2 Two daisy chained DMAs

The PSI5-S supports sorting by channel with only two DMA channels. Register TAR contains the target address that is to be copied by the first DMA into the target address register of the second DMA. Source and target of the first DMA are fixed (DMA register SADR and DADR). The target address offered in TAR is calculated by PSI5-S by adding an offset value to the programmable base address defined in BAR. This allows for locating the channel buffers at any location in system memory.

The resulting base + offset is presented in TAR. The target address depends on:

- BAR (base address)
- ChID: offset of channel number currently received (channel number x 6 (buffer lines) x 3 (words per frame) x 4 Bytes per word = 72 Bytes x channel number. This allows to receive 6 frames per channel which equals the maximum number of PSI5 Frames per Sync Pulse.
- FID: offset of frame ID currently received. (3 words i.e. 12 Bytes per frame)
Note that the frame ID used for this sorting depends from RCRAx.FIDS! In particular in asynchronous mode, FIDS needs to be set so that the FID from the Packet Frame is replaced by a rolling number.
- Note that the message reassembly unit forces non recoverable messages to be stored at the address for ChID = 0, FID = 1 but with original ChID and FID! I.e. RDS contains the CHID and FID as received in the first UART Frame for better debugging!
- The source address of the second DMA needs to be programmed to be fixed to the address of RDS. Use shadow control to “wrap around” the source address.

Peripheral Sensor Interface with Serial PHY Connection

(DMA_MExADICR.SHCT must be set to 0x0101b).

Block transfer with address auto incrementing must be selected and the number of transactions programmed to 3 (see DMA chapter, DMA_MExCHCR.BLKM) 32 bit (the data width is defined in DMA_MExCHCR.CHDW) accesses.

Use shadow control to “wrap around” the source address. (DMA_MExADICR.SHCT must be set to 0x0101b).

Peripheral Sensor Interface with Serial PHY Connection

The source address of the second DMA needs to be programmed to be fixed (see DMA chapter, register ADRCR_{xz}) to the address of RDS.

Block transfer with address auto incrementing must be selected and the number of transactions programmed to 3 (see DMA chapter, DMA_MExCHCR.BLKM) 32 bit (the data width is defined in DMA_MExCHCR.CHDW) accesses.

Use shadow control to “wrap around” the source address. (DMA_MExADICR.SHCT must be set to 0x0101b).

Peripheral Sensor Interface with Serial PHY Connection

The second DMA must be of the next lower priority than the first. Selecting hardware request input via $CHCRxz.PRSEL$ allows for daisy chaining the two DMA channels. See **Figure 34-23**.

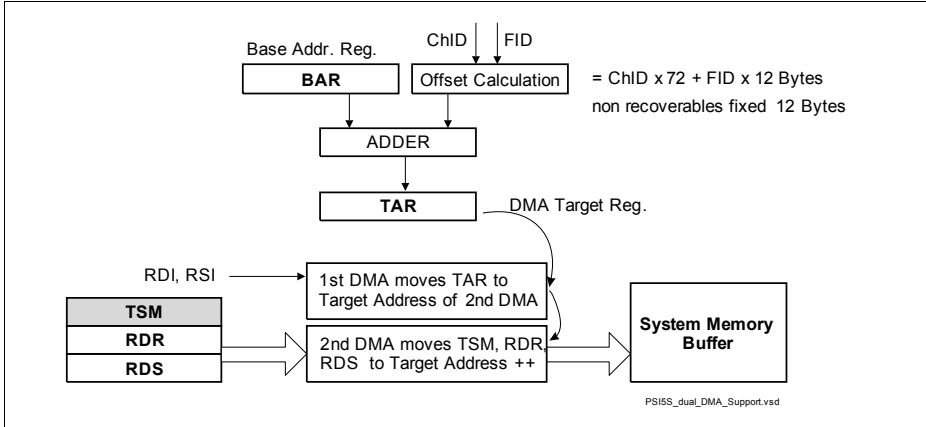


Figure 34-23 PSI5-S Dual DMA Support (daisy chained)

Table 34-1 shows the detailed mapping of the received frames resulting from the described calculation rules:

Table 34-1 Frame Mapping

Offset (dec)	Offset (hex)	Channel/Frame	Content
0	0h	channel 0 frame 0	Status (Reserved for Transceiver Messages)
4	4h	channel 0 frame 0	Data (Reserved for Transceiver Messages)
8	8h	channel 0 frame 0	Time Stamp (Reserved for Transceiver Messages)
12	Ch	channel 0 frame 1	Status (Reserved for non recoverable Messages)
16	10h	channel 0 frame 1	Data (Reserved for non recoverable Messages)

Peripheral Sensor Interface with Serial PHY Connection
Table 34-1 Frame Mapping

Offset (dec)	Offset (hex)	Channel/Frame	Content
20	14h	channel 0 frame 1	Time Stamp (Reserved for non recoverable Messages)
24	18h	channel 0 frame 2	Status
28	1Ch	channel 0 frame 2	Data
32	20h	channel 0 frame 2	Time Stamp
36	24h	channel 0 frame 3	Status
40	28h	channel 0 frame 3	Data
44	2Ch	channel 0 frame 3	Time Stamp
48	30h	channel 0 frame 4	Status
52	34h	channel 0 frame 4	Data
56	38h	channel 0 frame 4	Time Stamp
60	3Ch	channel 0 frame 5	Status
64	40h	channel 0 frame 5	Data
68	44h	channel 0 frame 5	Time Stamp
72	48h	channel 1 frame 0	Status
76	4Ch	channel 1 frame 0	Data
80	50h	channel 1 frame 0	Time Stamp
84	54h	channel 1 frame 1	Status
88	58h	channel 1 frame 1	Data
92	5Ch	channel 1 frame 1	Time Stamp
96	60h	channel 1 frame 2	Status
100	64h	channel 1 frame 2	Data
104	68h	channel 1 frame 2	Time Stamp
108	6Ch	channel 1 frame 3	Status
112	70h	channel 1 frame 3	Data
116	74h	channel 1 frame 3	Time Stamp
120	78h	channel 1 frame 4	Status
124	7Ch	channel 1 frame 4	Data
128	80h	channel 1 frame 4	Time Stamp
132	84h	channel 1 frame 5	Status

Peripheral Sensor Interface with Serial PHY Connection
Table 34-1 Frame Mapping

Offset (dec)	Offset (hex)	Channel/Frame	Content
136	88h	channel 1 frame 5	Data
140	8Ch	channel 1 frame 5	Time Stamp
144	90h	channel 2 frame 0	Status
148	94h	channel 2 frame 0	Data
152	98h	channel 2 frame 0	Time Stamp
156	9Ch	channel 2 frame 1	Status
160	A0h	channel 2 frame 1	Data
164	A4h	channel 2 frame 1	Time Stamp
168	A8h	channel 2 frame 2	Status
172	ACh	channel 2 frame 2	Data
176	B0h	channel 2 frame 2	Time Stamp
180	B4h	channel 2 frame 3	Status
184	B8h	channel 2 frame 3	Data
188	BCh	channel 2 frame 3	Time Stamp
192	C0h	channel 2 frame 4	Status
196	C4h	channel 2 frame 4	Data
200	C8h	channel 2 frame 4	Time Stamp
204	CCh	channel 2 frame 5	Status
208	D0h	channel 2 frame 5	Data
212	D4h	channel 2 frame 5	Time Stamp
216	D8h	channel 3 frame 0	Status
220	DCh	channel 3 frame 0	Data
224	E0h	channel 3 frame 0	Time Stamp
228	E4h	channel 3 frame 1	Status
232	E8h	channel 3 frame 1	Data
236	ECh	channel 3 frame 1	Time Stamp
240	F0h	channel 3 frame 2	Status
244	F4h	channel 3 frame 2	Data
248	F8h	channel 3 frame 2	Time Stamp
252	FCh	channel 3 frame 3	Status

Peripheral Sensor Interface with Serial PHY Connection
Table 34-1 Frame Mapping

Offset (dec)	Offset (hex)	Channel/Frame	Content
256	100h	channel 3 frame 3	Data
260	104h	channel 3 frame 3	Time Stamp
264	108h	channel 3 frame 4	Status
268	10Ch	channel 3 frame 4	Data
272	110h	channel 3 frame 4	Time Stamp
276	114h	channel 3 frame 5	Status
280	118h	channel 3 frame 5	Data
284	11Ch	channel 3 frame 5	Time Stamp
288	120h	channel 4 frame 0	Status
292	124h	channel 4 frame 0	Data
296	128h	channel 4 frame 0	Time Stamp
300	12Ch	channel 4 frame 1	Status
304	130h	channel 4 frame 1	Data
308	134h	channel 4 frame 1	Time Stamp
312	138h	channel 4 frame 2	Status
316	13Ch	channel 4 frame 2	Data
320	140h	channel 4 frame 2	Time Stamp
324	144h	channel 4 frame 3	Status
328	148h	channel 4 frame 3	Data
332	14Ch	channel 4 frame 3	Time Stamp
336	150h	channel 4 frame 4	Status
340	154h	channel 4 frame 4	Data
344	158h	channel 4 frame 4	Time Stamp
348	15Ch	channel 4 frame 5	Status
352	160h	channel 4 frame 5	Data
356	164h	channel 4 frame 5	Time Stamp
360	168h	channel 5 frame 0	Status
364	16Ch	channel 5 frame 0	Data
368	170h	channel 5 frame 0	Time Stamp
372	174h	channel 5 frame 1	Status

Peripheral Sensor Interface with Serial PHY Connection
Table 34-1 Frame Mapping

Offset (dec)	Offset (hex)	Channel/Frame	Content
376	178h	channel 5 frame 1	Data
380	17Ch	channel 5 frame 1	Time Stamp
384	180h	channel 5 frame 2	Status
388	184h	channel 5 frame 2	Data
392	188h	channel 5 frame 2	Time Stamp
396	18Ch	channel 5 frame 3	Status
400	190h	channel 5 frame 3	Data
404	194h	channel 5 frame 3	Time Stamp
408	198h	channel 5 frame 4	Status
412	19Ch	channel 5 frame 4	Data
416	1A0h	channel 5 frame 4	Time Stamp
420	1A4h	channel 5 frame 5	Status
424	1A8h	channel 5 frame 5	Data
428	1ACh	channel 5 frame 5	Time Stamp
432	1B0h	channel 6 frame 0	Status
436	1B4h	channel 6 frame 0	Data
440	1B8h	channel 6 frame 0	Time Stamp
444	1BCh	channel 6 frame 1	Status
448	1C0h	channel 6 frame 1	Data
452	1C4h	channel 6 frame 1	Time Stamp
456	1C8h	channel 6 frame 2	Status
460	1CCh	channel 6 frame 2	Data
464	1D0h	channel 6 frame 2	Time Stamp
468	1D4h	channel 6 frame 3	Status
472	1D8h	channel 6 frame 3	Data
476	1DCh	channel 6 frame 3	Time Stamp
480	1E0h	channel 6 frame 4	Status
484	1E4h	channel 6 frame 4	Data
488	1E8h	channel 6 frame 4	Time Stamp
492	1ECh	channel 6 frame 5	Status

Peripheral Sensor Interface with Serial PHY Connection
Table 34-1 Frame Mapping

Offset (dec)	Offset (hex)	Channel/Frame	Content
496	1F0h	channel 6 frame 5	Data
500	1F4h	channel 6 frame 5	Time Stamp
504	1F8h	channel 7 frame 0	Status
508	1FCh	channel 7 frame 0	Data
512	200h	channel 7 frame 0	Time Stamp
516	204h	channel 7 frame 1	Status
520	208h	channel 7 frame 1	Data
524	20Ch	channel 7 frame 1	Time Stamp
528	210h	channel 7 frame 2	Status
532	214h	channel 7 frame 2	Data
536	218h	channel 7 frame 2	Time Stamp
540	21Ch	channel 7 frame 3	Status
544	220h	channel 7 frame 3	Data
548	224h	channel 7 frame 3	Time Stamp
552	228h	channel 7 frame 4	Status
556	22Ch	channel 7 frame 4	Data
560	230h	channel 7 frame 4	Time Stamp
564	234h	channel 7 frame 5	Status
568	238h	channel 7 frame 5	Data
572	23Ch	channel 7 frame 5	Time Stamp

34.11.3 Interrupts for DMA support

Interrupts for Single DMA usage

Interrupts RDI or RSI can be used as trigger for the DMA transfer from TSM/RDS/RDR.

Interrupts for two daisy chained DMA usage

If one of the buffers is full, this is signalled to SW by interrupt CHCI. The “status buffer full” is defined as follows: current number of received frames counted in FCNT.FCx equals the number of expected frames configured in NFC.NFx. In synchronous mode this is the case, if all expected PSI5 frames after a Sync Pulse are received by PSI5-S. The number of expected frames is looked up in register NFC.NFx for channel x. In Asynchronous mode, NFC.NFx should be programmed to 6 while RCRAx.FIDS is set. This way the FID for an asynchronous channel is a rolling number 0 .. 5 and the 6 buffer locations are used completely before CHCI is issued. See [Figure 34-24](#).

Peripheral Sensor Interface with Serial PHY Connection

There is a delay between CHCI and the completion of the DMA transfer of the last set of RDS, RDR and TSM to the system memory! SW needs special checks to grant data consistency, e.g. evaluating the DMA interrupts or comparison of the Time Stamps in the System Memory!

CHCI is not granting that all data is completely moved to system memory by DMA!

Improved Data Consistency with Packet Frame Count (PFC)

For data consistency, RDR, RDS and TSM are tagged with a Packet Frame count value. For each channel there is a separate internal packet counter. This allows to read a triplet of RDR, RDS and TSM from the system memory without looking at interrupts. If the PFC is identical all three values belong together. Otherwise they need to be read again until PFC is identical.

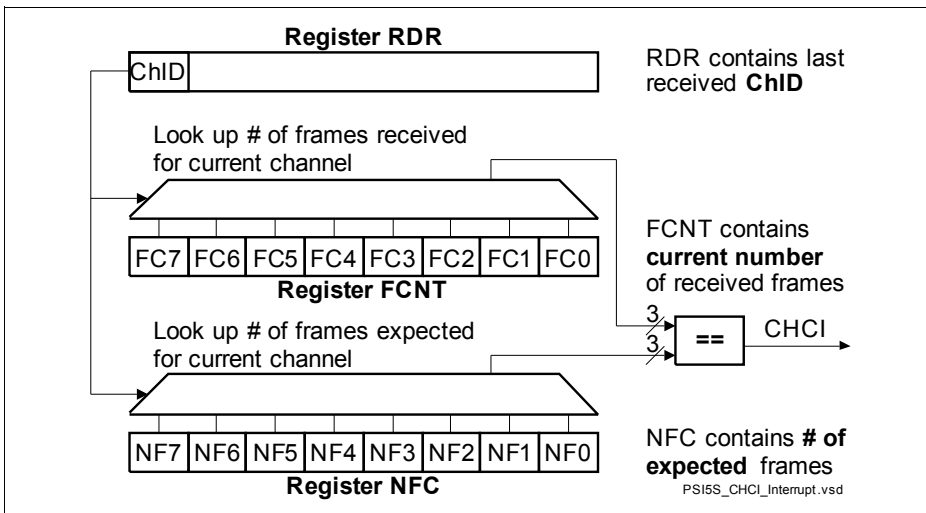


Figure 34-24 PSI5-S Generation of Interrupt CHCI

34.12 Error Detection Capabilities

Each PSI5-S channel can detect and signal the following error conditions:

Protocol Level:

- Packet Frame CRC Error (XCRC)
- PSI5 Frame Checksum error (CRCI)
- Frame not sent in time / UART Frames missing (TEI)
- Error Bits Set in Packet Frame (HDI)
- Errors Signalled by ASC Sub Module (i.e. PE, FE, OE)

Peripheral Sensor Interface with Serial PHY Connection

Transfer Management Level:

- Receive Data Buffer Overrun (RBI)
- ECU to Sensor Data Buffer Underrun (TBI)

34.13 Special use of Channel 0

Note that data that is received in incomplete or too long Packet Frames is always copied - as is - to Channel 0 Frame ID 1. This can happen, if UART Frames are lost, the PHY sends non allowed data or idle before the actual end of the Packet Frame. See [““Packet Frames” received from PHY” on Page 34-6](#).

This is the only actual reservation / limitation in functionality of Channel 0. I.e. all frames can be received and sorted correctly in Channel 0 as long as the Packet Frame is configured with: ChID = 0. It is recommended not to use ChID = 0 AND FID = 1 for normal sensor frames but to keep FID = {2, 3, 4, 5}.

The target application assumes, that the PHY sends messages for its own house keeping with ChID = 0, FID = 0. This is assumed in this document and e.g. shown in [Table 34-1 “Frame Mapping” on Page 34-31](#). Nevertheless, a different application may not have this functionality or may use a different ChID / FID configuration.

34.14 ASC Kernel Description

[Figure 34-25](#) shows a global view of the ASC interface. The ASC is fully integrated into the PSI5-S module. It can be switched to dedicated internal use of the PSI5-S module or to ASC only mode where all PSI5 functionality is disabled. In the last case, the whole PSI5-S module represents one additional ASC in the system.

Peripheral Sensor Interface with Serial PHY Connection

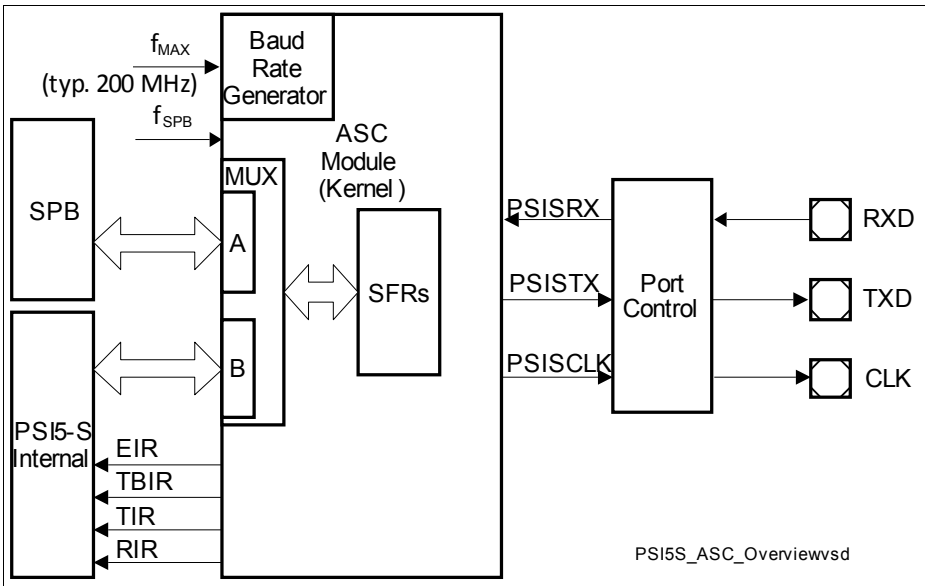


Figure 34-25 General Block Diagram of the ASC Interface

The ASC module communicates with the external world via two I/O lines. The RXD line is the receive data input signal (and also output signal in Synchronous Mode), and TXD is the transmit output signal.

Clock control, address decoding, and interrupt service request control are managed outside the ASC module kernel.

34.14.1 Overview

The ASC provides serial communication between the TC27x and other micro controllers, microprocessors, or external peripherals.

The ASC supports full-duplex asynchronous communication and half-duplex synchronous communication. In Synchronous Mode, data is transmitted or received synchronous to a shift clock that is generated by the ASC internally. In Asynchronous Mode, 8-bit or 9-bit data transfer, parity generation, and the number of stop bits can be selected. Parity, framing, and overrun error detection are provided to increase the reliability of data transfers. Transmission and reception of data is double-buffered in ASC only mode. During internal PSI5-S mode, double-buffering is switched off for the transmitter. The receiver stays double buffered. For multiprocessor communication, a mechanism is included to distinguish address Bytes from data Bytes. Testing is supported by a loop-back option. A 13-bit baud rate generator provides the ASC with a

Peripheral Sensor Interface with Serial PHY Connection

separate serial clock signal, which can be accurately adjusted by a prescaler implemented as fractional divider.

Features

- Full-duplex asynchronous operating modes
 - 8-bit or 9-bit data frames, LSB first
 - Parity-bit generation/checking
 - Separate Parity control for receive and transmit
 - One or two stop bits
 - Baud rate from 12,5 Mbit/s to 3 bit/s (@ 200 MHz module clock)
 - Multiprocessor mode for automatic address/data Byte detection
 - Loop-back capability
- Half-duplex 8-bit synchronous operating mode
 - Baud rate from 25 Mbit/s to 2034,5 bit/s (@ 200 MHz module clock)
- Double-buffered transmitter (in ASC-only mode)/receiver
- Interrupt generation
 - On a transmit buffer empty condition
 - On a transmit last bit of a frame condition
 - On a receive buffer full condition
 - On an error condition (frame, parity, overrun error)
- Implementation features
 - Connections to DMA Controller
 - Connections of receiver input to GTM for baud rate detection and LIN break signal measuring

Peripheral Sensor Interface with Serial PHY Connection

34.14.2 General Operation

The ASC supports full-duplex asynchronous communication and half-duplex synchronous communication. In Synchronous Mode, data is transmitted or received synchronously to a shift clock generated by the microcontroller. In Asynchronous Mode, 8-bit or 9-bit data transfer, parity generation, and the number of stop bits can be selected. Parity, framing, and overrun error detection are provided to increase the reliability of data transfers. Transmission and reception of data are double-buffered. For multiprocessor communication, a mechanism is included to distinguish address Bytes from data Bytes. Testing is supported by a loop-back option. A 13-bit baud rate generator provides the ASC with a separate serial clock signal, which can be accurately adjusted by a prescaler implemented as fractional divider.

A transmission is started by writing to the Transmit Buffer Register, TBUF. Only the number of data bits determined by the selected operating mode will actually be transmitted; that is, bits written to positions 9 through 15 of register TBUF are always insignificant. Data transmission is double-buffered, so a new character may be written to TBUF before the transmission of the previous character is complete. This allows a back-to-back transmission of characters to take place without gaps.

Data reception is enabled by the receiver enable bit CON.REN. After a reception has been completed, the received data and, if provided by the selected operating mode, the received parity bit can be read from the (read-only) receive buffer register RBUF. Unused bits in the upper half of RBUF that are not required in the selected operating mode will be read as zeros.

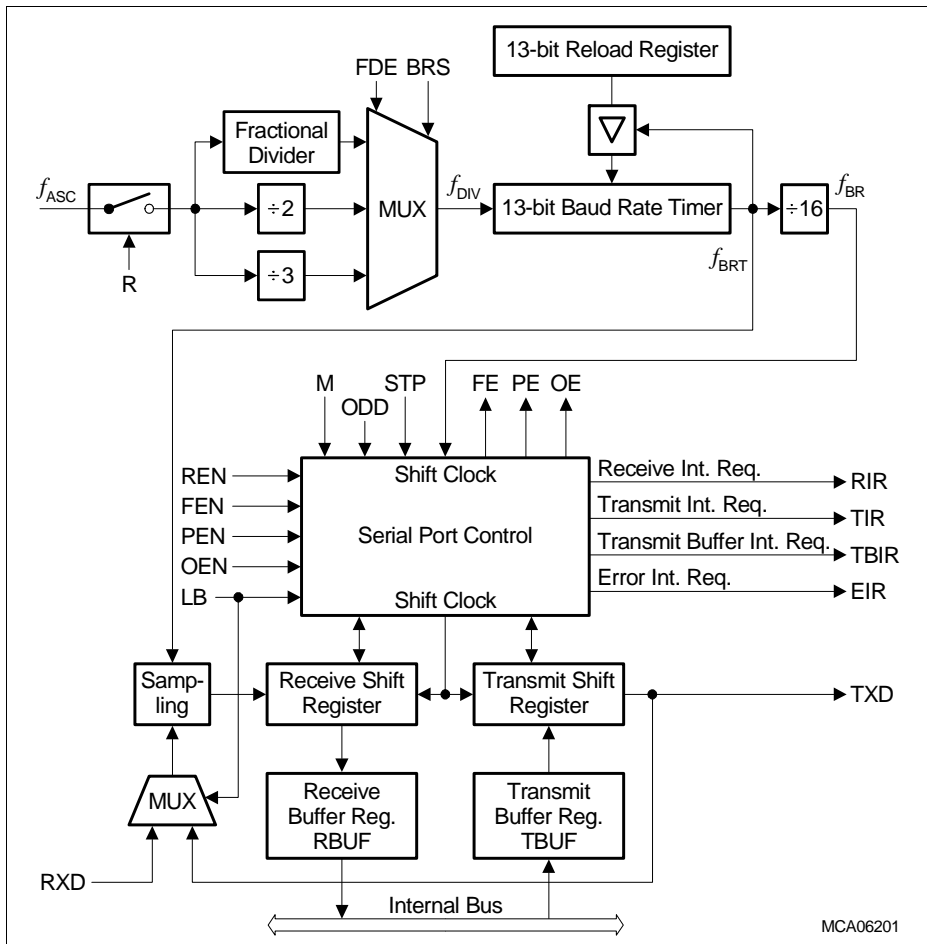
Data reception is double-buffered, so that reception of a second character may already begin before the previously received character has been read out of the receive buffer register. During the exclusive use by the PSI5-S, double buffering is switched off. The PSI5-S internal output FIFO is used instead. As commands for sync pulses are time stamped inside PSI5-S as they leave the internal FIFO, the double buffering would add unwanted jitter. In all modes, receive buffer overrun error detection can be selected through bit CON.OEN. When enabled, the overrun error status flag CON.OE and the error interrupt request line EIR will be activated when the receive buffer register has not been read by the time reception of a second character is complete. In this case, the previously received character in the receive buffer is overwritten.

The loop-back option (selected by bit CON.LB) allows the data currently being transmitted to be received simultaneously in the receive buffer. This may be used to test serial communication routines at an early stage without having to provide an external network. In loop-back mode, the alternate input/output function of port pins is not required.

Peripheral Sensor Interface with Serial PHY Connection

34.14.3 Asynchronous Operation

Asynchronous Mode supports full-duplex communication, in which both transmitter and receiver use the same data frame format and have the same baud rate. Data is transmitted on pin TXD and received on pin RXD. **Figure 34-26** shows the block diagram of the ASC when operating in Asynchronous Mode.


Figure 34-26 Asynchronous Mode of the ASC

Peripheral Sensor Interface with Serial PHY Connection

34.14.3.1 Asynchronous Data Frames

Asynchronous data frames can consist of 8-bit or 9-bit data frames.

8-bit Data Frames

The 8-bit data frames consist of either eight data bits D7 ... D0 (CON.M = 001_B), or of seven data bits D6 ... D0 plus an automatically generated parity bit (CON.M = 011_B). Parity may be odd or even, depending on bit CON.ODD. An even parity bit will be set if the modulo-2 sum of the seven data bits is 1. An odd parity bit will be cleared in this case. Parity checking is enabled via bit CON.PEN (always OFF in 8-bit data mode). The parity error flag CON.PE will be set, along with the error interrupt request flag, if a wrong parity bit is received. The received parity bit itself will be stored in RBUF too.

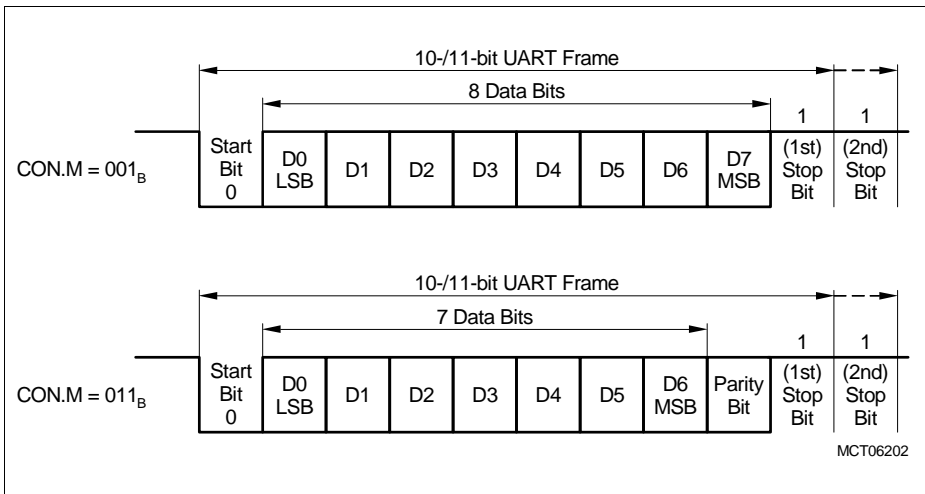


Figure 34-27 Asynchronous 8-bit Frames

Peripheral Sensor Interface with Serial PHY Connection

9-bit Data Frames

The 9-bit data frames consist of nine data bits D8 ... D0 (CON.M = 100_B), or of eight data bits D7 ... D0 plus an automatically generated parity bit (CON.M = 111_B) or of eight data bits D7 ... D0 plus wake-up bit (CON.M = 101_B). Parity may be odd or even, depending on bit CON.ODD. An even parity bit will be set if the modulo-2-sum of the eight data bits is 1. An odd parity bit will be cleared in this case. Parity checking is enabled via bit CON.PEN (always OFF in 9-bit data and wake-up mode). The parity error flag CON.PE will be set along with the error interrupt request flag if a wrong parity bit is received. The received parity bit itself will be stored in RBUF too.

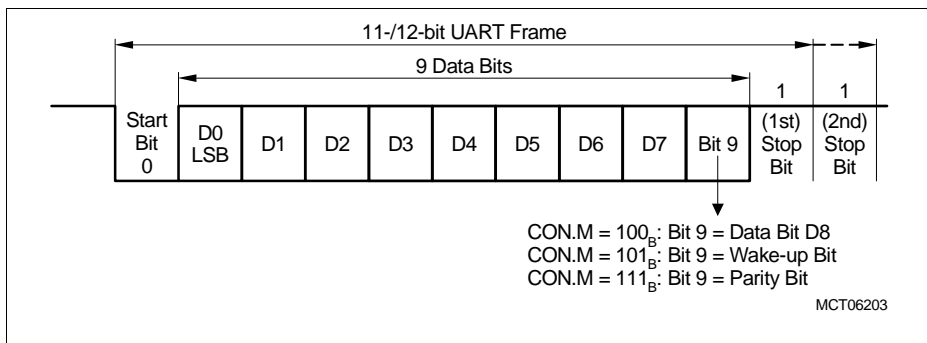


Figure 34-28 Asynchronous 9-bit Frames

In Wake-up Mode (CON.M = 101_B), received frames are transferred to the receive buffer register only if the 9th bit (the wake-up bit) of the frame is 1. If this bit is 0, no receive interrupt request will be activated and no data will be transferred.

This feature can be used to control communication in multi-processor systems, for example:

When the master processor aims to transmit a block of data to one of several slaves, it first sends out an address 'Byte' (in this case, a 'Byte' consists of nine bits) that identifies the target slave. An address 'Byte' differs from a data 'Byte' in that the additional 9th bit is a 1 for an address 'Byte' but is a 0 for a data 'Byte', so, no slave will be interrupted by a data 'Byte'. An address 'Byte' will interrupt all slaves (operating in 8-bit data + wake-up bit mode), so each slave can examine the eight LSBs of the received character (the address). The addressed slave will switch to 9-bit data mode (for example, by clearing bit CON.M[0]), which enables it to also receive the data Bytes that will be coming (having the wake-up bit cleared). The slaves that were not being addressed remain in 8-bit data + wake-up bit mode, ignoring the following data 'Bytes'.

Peripheral Sensor Interface with Serial PHY Connection

34.14.3.2 Asynchronous Transmission

Asynchronous transmission begins when the next overflow of the divide-by-16 baud rate timer (transition of the baud rate clock f_{BR}) occurs, if bit CON.R is set and data has been loaded into TBUF. The transmitted data frame consists of three elements:

1. The start bit
2. The data field (8 or 9 bits, LSB first, including a parity bit, if selected)
3. The delimiter (1 or 2 stop bits)

Data transmission is double-buffered. When the transmitter is idle, the transmit data loaded into TBUF is immediately moved to the transmit shift register; thus, freeing TBUF for the next transmit data to be loaded. This is indicated by the transmit buffer interrupt request line TBIR being activated. TBUF may then be loaded with the next transmit data while transmission of the previous one continues.

The Transmit Interrupt Request line TIR will be activated before the last bit of a frame is transmitted, that is, before the first or the second stop bit is shifted out of the transmit shift register.

Note: A dedicated GPIO device pin which is connected to the module output pin TXD must be configured by software as alternate data output for asynchronous transmission.

34.14.3.3 Asynchronous Reception

Asynchronous reception is initiated by a falling edge (1-to-0 transition) on pin RXD, on the condition that bits CON.R and CON.REN are set. The receive data input pin RXD is sampled at sixteen times the rate of the selected baud rate. A majority decision of the 7th, 8th and 9th sample determines the effective sampled bit value. This avoids erroneous results that may be caused by noise.

If the detected value is not a 0 when the start bit is sampled, the receive circuit is reset and waits for the next 1-to-0 transition at pin RXD. If the start bit proves valid, the receive circuit continues sampling and shifts the incoming data frame into the receive shift register.

When the last stop bit has been received, the contents of the receive shift register are transferred to the Receive Data Buffer Register RBUF. Simultaneously, the receive interrupt request line RIR is activated after the 9th sample in the last stop bit time-slot (as programmed), regardless whether valid stop bits have been received or not. The receive circuit then waits for the next start bit (1-to-0 transition) at the receive data input line.

Note: A dedicated GPIO pin that is connected to the module input pin RXD must be configured by software as input for asynchronous reception.

Asynchronous reception is stopped by clearing bit CON.REN. A currently received frame is completed including generation of the receive interrupt request and an error interrupt request, if appropriate. Start bits that follow this frame will not be recognized.

Peripheral Sensor Interface with Serial PHY Connection

Note: In wake-up mode, received frames are transferred to the receive buffer register only if the 9th bit (the wake-up bit) is 1. If this bit is 0, no receive interrupt request will be activated and no data will be transferred.

34.14.3.4 RXD/TXD Data Path Selection in Asynchronous Modes

The data paths for the serial input and output data in Asynchronous Modes are affected by control bit CON.LB (loop-back) as shown in [Figure 34-29](#).

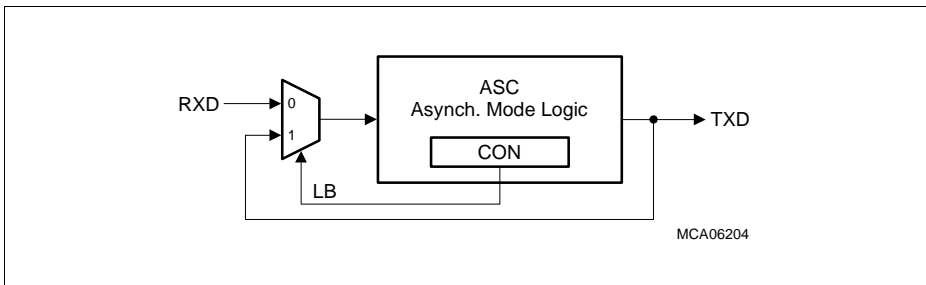


Figure 34-29 RXD/TXD Data Path Selection in Asynchronous Modes

Peripheral Sensor Interface with Serial PHY Connection

34.14.4 Synchronous Operation

Synchronous Mode supports half-duplex communication, usable for simple I/O expansion via shift registers. Data is transmitted and received via pin RXD while pin TXD outputs the shift clock. These signals are typically connected as alternate functions with GPIO port pins. Synchronous Mode is selected with CON.M = 000_B.

Eight data bits are transmitted or received synchronous to a shift clock generated by the internal baud rate generator. The shift clock is active only as long as data bits are transmitted or received.

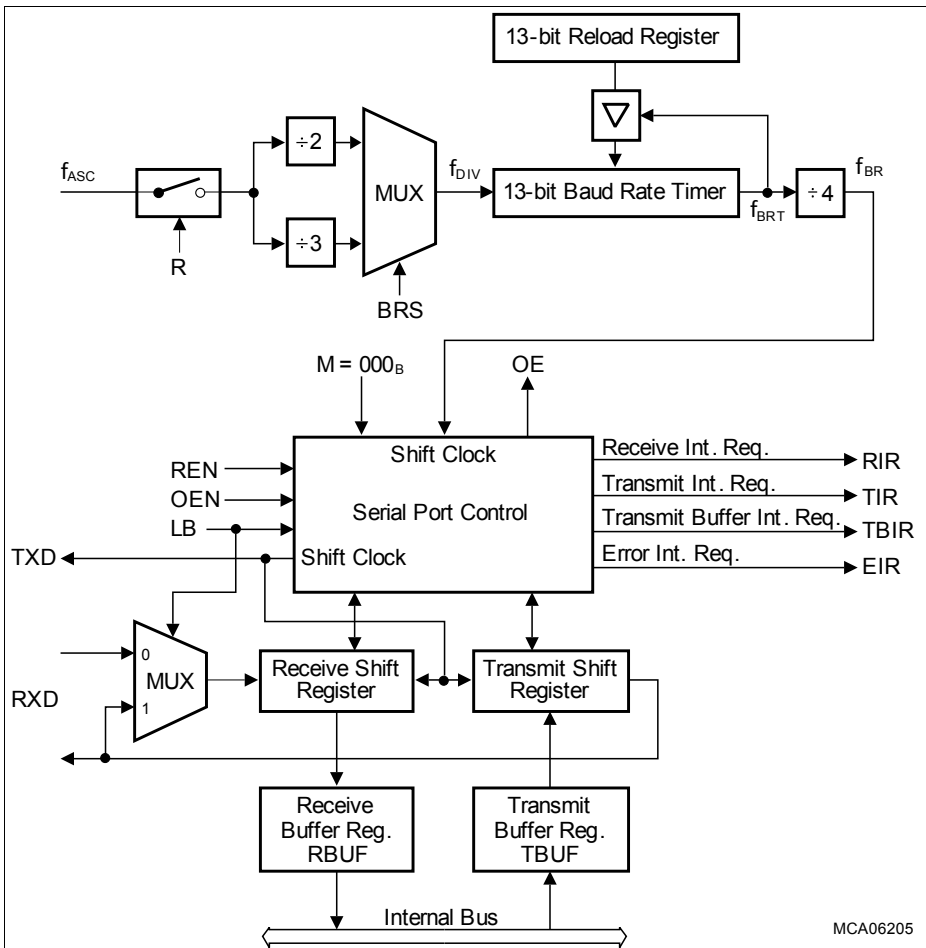


Figure 34-30 Synchronous Mode of Serial Channel ASC

Peripheral Sensor Interface with Serial PHY Connection

34.14.4.1 Synchronous Transmission

Synchronous transmission begins after data has been loaded into TBUF, provided that CON.R is set and CON.REN = 0 (half-duplex, no reception), with one exception: in asynchronous Loop-back Mode (bit CON.LB set), CON.REN must be set for reception of the transmitted Byte. In synchronous Loop-back Mode, the transmitted Byte can still be received if CON.REN=0. Data transmission is double-buffered. When the transmitter is idle, the transmit data loaded into TBUF is immediately moved to the transmit shift register, thus freeing TBUF for the next data to be sent. This is indicated by the transmit buffer interrupt request line TBIR being activated. TBUF may now be loaded with the next data, while transmission of the previous one continues. The data bits are transmitted synchronously with the shift clock. After the bit time for the 8th data bit, both TXD and RXD will be set to high level, the transmit interrupt request line TIR is activated, and serial data transmission stops.

Note: The dedicated GPIO device pins that are connected to TXD and RXD must be configured by software as alternate data outputs in order to provide the shift clock and the output data during synchronous transmission.

34.14.4.2 Synchronous Reception

Synchronous reception is initiated by setting bit CON.REN = 1. If bit CON.R = 1, the data applied at RXD is clocked into the receive shift register synchronously to the clock which is output at TXD. After the 8th bit has been shifted in, the contents of the receive shift register are transferred to the receive data buffer RBUF, the receive interrupt request line RIR is activated, the receiver enable bit CON.REN is reset, and serial data reception stops.

Synchronous reception is stopped by clearing bit CON.REN. Any Byte that is currently being received is completed, including the generation of the receive interrupt request and an error interrupt request, if appropriate. Writing to the transmit buffer register while a reception is in progress has no effect on reception and will not start a transmission.

If a previously received Byte has not been read out of the receive buffer register by the time the reception of the next Byte is complete, both the error interrupt request line EIR and the overrun error status flag CON.OE will be activated/set, provided that the overrun check has been enabled by bit CON.OEN.

Note: The dedicated GPIO device pin that is connected to TXD must be configured by software as alternate data output in order to provide the shift clock. The dedicated GPIO device pin that is connected to RXD must be configured by software as input during synchronous reception.

Peripheral Sensor Interface with Serial PHY Connection

34.14.4.3 Synchronous Timing

Figure 34-31 shows timing diagrams of the ASC Synchronous Mode data reception and data transmission. In idle state, the shift clock is at high level. With the beginning of a synchronous transmission of a data Byte, the data is shifted out at RXD with the falling edge of the shift clock. If a data Byte is received through RXD, data is latched with the rising edge of the shift clock.

One shift clock cycle (f_{BR}) delay is inserted between two consecutive receive or transmit data Bytes.

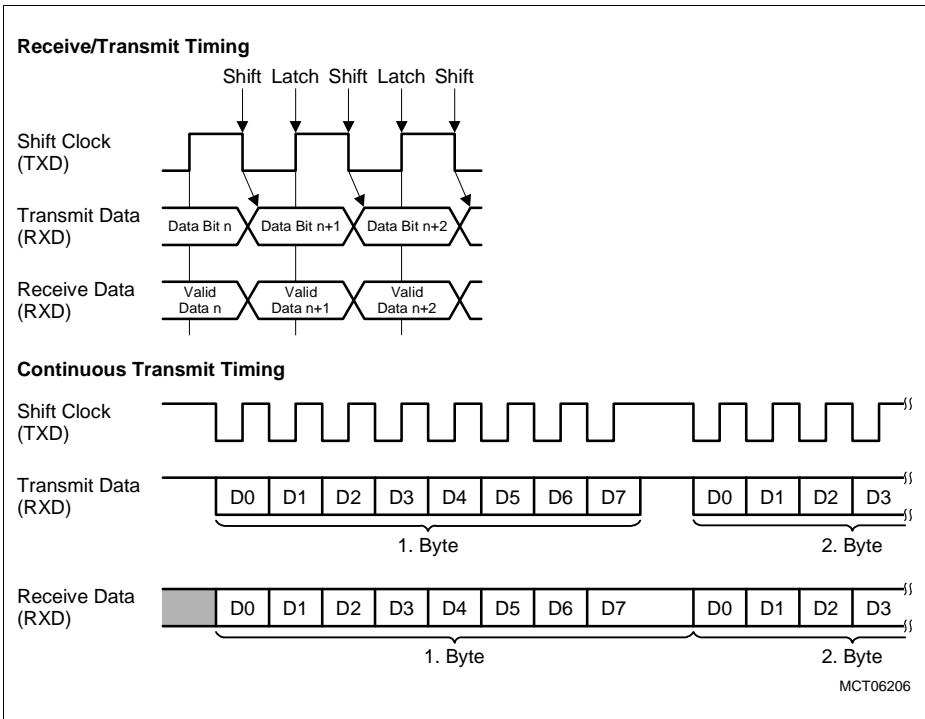


Figure 34-31 ASC Synchronous Mode Waveforms

Peripheral Sensor Interface with Serial PHY Connection

34.14.5 Baud Rate Generation

The ASC has its own dedicated 13-bit baud rate generator with 13-bit reload capability, allowing baud rate generation independent of other timers.

The baud rate generator is clocked with a clock (f_{DIV}) which is derived via a prescaler from the ASC module clock f_{ASC} . The baud rate timer is counting downwards and can be started or stopped through the baud rate generator run bit CON.R. Each underflow of the timer generates one clock pulse to the serial channel. The timer is reloaded with the value stored in its 13-bit reload register at each underflow. The resulting clock f_{BRT} is again divided by a factor for the baud rate clock ($\div 16$ in asynchronous operating modes and $\div 4$ in synchronous operating mode). The prescaler is selected by the bits CON.BRS and CON.FDE. In the asynchronous operating modes, a fractional divider prescaler unit is available (in addition to the two fixed dividers) that allows selection of prescaler divider ratios of $n/2048$ with $n = 0-2047$. Therefore, the baud rate of ASC is determined by the module clock, the content of register FDV, the reload value in register BG, and the operating mode (asynchronous or synchronous).

Register BG is the dual-function baud rate generator/reload register. Reading BG returns the contents of the timer in bit field BR_VALUE (bits 31:13 return zero), while writing to BG always updates the reload register (bits 31:13 are insignificant).

An auto-reload of the timer with the contents of the reload register is performed each time BG is written to. However, if CON.R = 0 at the time the write operation to BG is performed, the timer will not be reloaded until the first instruction cycle after CON.R = 1. For a clean baud rate initialization, BG should only be written if CON.R = 0. If BG is written with CON.R = 1, an unpredictable behavior of the ASC may occur during running transmit or receive operations.

Peripheral Sensor Interface with Serial PHY Connection

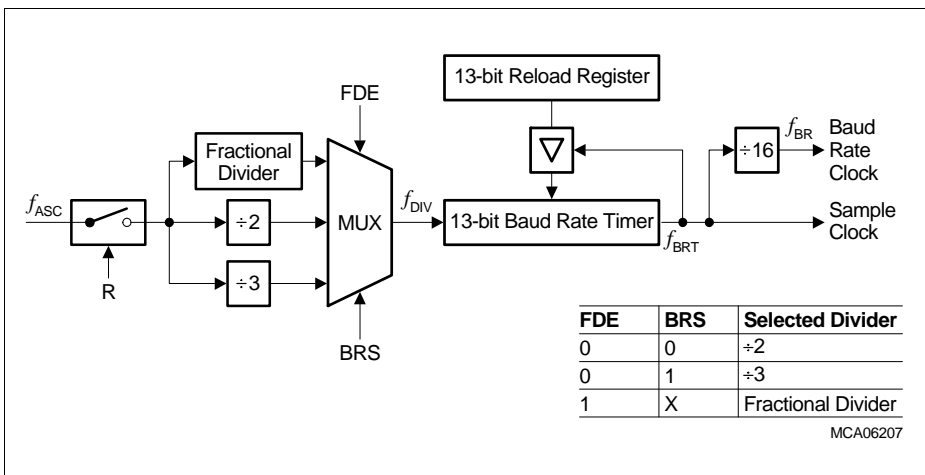
34.14.5.1 Baud Rates in Asynchronous Mode

For asynchronous operation, the baud rate generator provides a clock f_{BRT} with sixteen times the rate of the established baud rate. Every received bit is sampled on the 7th, 8th and 9th cycle of this clock. The clock divider circuitry, which generates the input clock f_{DIV} for the 13-bit baud rate timer, is extended by a fractional divider circuitry that allows the adjustment of more accurate baud rates and the extension of the baud rate range.

The baud rate of the baud rate generator depends on the settings of the following bits and register values:

- Input clock f_{ASC}
- Selection of the baud rate timer input clock f_{DIV} by bits CON.FDE and CON.BRS
- If bit CON.FDE = 1 (fractional divider): value of register FDV
- Value of the 13-bit reload register BG

The output clock of the baud rate timer with the reload register is the sample clock in the asynchronous operating modes of the ASC. For baud rate calculations, this baud rate clock f_{BR} is derived from the sample clock f_{BRT} by a division of sixteen.


Figure 34-32 ASC Baud Rate Generator Circuitry in Asynchronous Modes

Peripheral Sensor Interface with Serial PHY Connection
Using the Fixed Input Clock Divider

The baud rate for asynchronous operation of the serial channel ASC when using the fixed input clock divider ratios (CON.FDE = 0) and the required BG reload value for a given baud rate can be determined by the following formulas:

Table 34-2 Asynchronous Baud Rate Formulas using the Fixed Input Clock Dividers

FDE	BRS	BG	Formula
0	0	0 ... 8191	Baud rate = $f_{ASC} / (32 \times (BG + 1))$ BG = $f_{ASC} / (32 \times \text{Baud rate}) - 1$
	1		Baud rate = $f_{ASC} / (48 \times (BG + 1))$ BG = $f_{ASC} / (48 \times \text{Baud rate}) - 1$

BG represents the content of the reload register bit field BG.BR_VALUE, taken as an unsigned 13-bit integer.

The maximum baud rate that can be achieved for the asynchronous operating modes when using the two fixed clock dividers and a module clock of 200 MHz is 6.25 Mbit/s.

Peripheral Sensor Interface with Serial PHY Connection
Using the Fractional Divider

When the fractional divider is selected, the input clock f_{DIV} for the baud rate timer is derived from the module clock f_{ASC} by a programmable fractional divider. If $CON.FDE = 1$, the fractional divider is activated. It divides f_{ASC} by a fraction of $n/2048$ for any value of n from 0 to 2047. If $n = 0$, the divider ratio is 1, which means that $f_{DIV} = f_{ASC}$. In general, the fractional divider allows the baud rate to be programmed with much better accuracy than with the two fixed prescaler divider stages.

Note: In fractional divider mode, the clock f_{DIV} can have a maximum period jitter of one f_{ASC} clock period.

Table 34-3 Asynchronous Baud Rate Formulas using the Fractional Input Clock Divider

FDE	BRS	BG	FDV	Formula
1	–	0 ... 8191	1 ... 2047	Baud rate = $(FDV / 2048) \times (f_{ASC} / (16 \times (BG + 1)))$
			0	Baud rate = $f_{ASC} / (16 \times (BG + 1))$

BG represents the content of the reload register bit field BG.BR_VALUE, taken as an unsigned 13-bit integer. FDV represents the contents of the fractional divider register bit field FDV.FD_VALUE, taken as an unsigned 11-bit integer.

Peripheral Sensor Interface with Serial PHY Connection

34.14.5.2 Baud Rates in Synchronous Mode

For synchronous operation, the baud rate generator provides a clock f_{BRT} that runs with four times the established baud rate (see [Figure 34-33](#)).

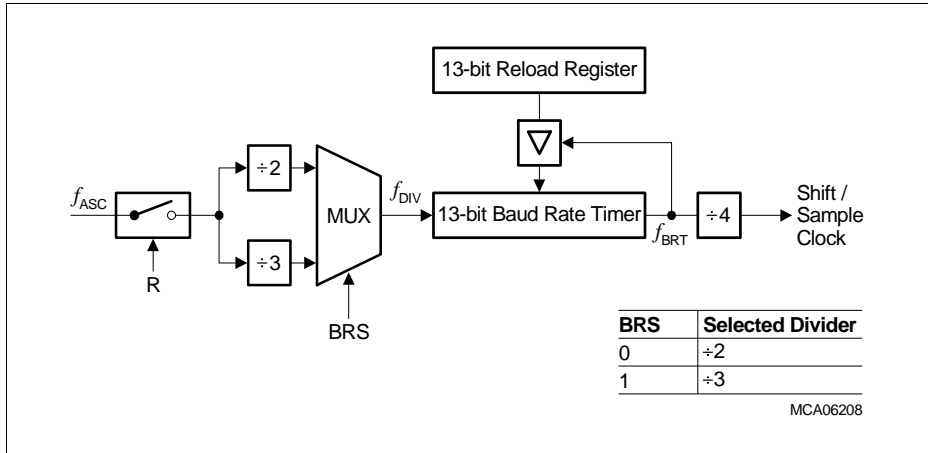


Figure 34-33 ASC Baud Rate Generator Circuitry in Synchronous Mode

The baud rate for synchronous operation of the serial channel ASC can be determined by the formulas as shown in [Table 34-4](#).

Table 34-4 Synchronous Baud Rate Formulas

BRS	BG	Formula
0	0 ... 8191	Baud rate = $f_{ASC} / (8 \times (BG + 1))$ BG = $f_{ASC} / (8 \times \text{Baud rate}) - 1$
1		Baud rate = $f_{ASC} / (12 \times (BG + 1))$ BG = $f_{ASC} / (12 \times \text{Baud rate}) - 1$

BG represents the content of the reload register bit field BG.BR_VALUE, taken as an unsigned 13-bit integer.

Peripheral Sensor Interface with Serial PHY Connection

34.14.6 Hardware Error Detection Capabilities

To improve the reliability of serial data exchange, the serial channel ASC provides an error interrupt request flag that indicates the presence of an error and three (selectable) error status flags in register CON that indicate which error has been detected during reception. Upon completion of a reception, the error interrupt request line EIR will be activated simultaneously with the receive interrupt request line RIR, if one or more of the following conditions are met:

- If the framing error detection enable bit CON.FEN is set and any of the expected stop bits is not high, the framing error flag CON.FE is set, indicating that the error interrupt request is due to a framing error (asynchronous operating modes only).
- If the parity error detection enable bit CON.PEN is set in the modes where a parity bit is received and the parity check on the received data bits proves false, the parity error flag CON.PE is set, indicating that the error interrupt request is due to a parity error (asynchronous operating modes only).
- If the overrun error detection enable bit CON.OEN is set and the last character received was not read out of the receive buffer by software or DMA transfer at the time the reception of a new frame is complete, the overrun error flag CON.OE is set indicating that the error interrupt request is due to an overrun error (Asynchronous and Synchronous Modes).

34.14.7 Interrupts

Four interrupt sources are provided for serial channel ASC. Line TIR indicates a transmit interrupt, TBIR indicates a transmit buffer interrupt, RIR indicates a receive interrupt, and EIR indicates an error interrupt of the serial channel. The interrupt output lines TBIR, TIR, RIR, and EIR are synchronized to the PSI5-S kernel and routed by the INPs to the trigger output lines.

The cause of an error interrupt request EIR (framing, parity, overrun error) can be identified by the error status flags CON.FE, CON.PE, and CON.OE.

Note: By contrast to the error interrupt request line EIR, the error status flags CON.FE/CON.PE/CON.OE are not reset automatically but must be cleared by software.

For normal operation (that is, other than error interrupt), the ASC provides three interrupt requests to control data exchange via this serial channel:

- TBIR is activated when data is moved from TBUF to the transmit shift register.
- TIR is activated before the last bit of an asynchronous frame is transmitted, or after the last bit of a synchronous frame has been transmitted.
- RIR is activated when the received frame is moved to RBUF.

While the task of the receive interrupt handler is quite clear, the transmitter is serviced by two interrupt handlers. This provides advantages for the servicing software.

Peripheral Sensor Interface with Serial PHY Connection

For single transfers, it is sufficient to use the transmitter interrupt (TIR), which indicates that the previously loaded data has been transmitted, except for the last bit of an asynchronous frame.

For multiple back-to-back transfers, it is necessary to load the following piece of data at least before the last bit of the previous frame has been transmitted. In Asynchronous Mode, this leaves just one bit-time for the handler to respond to the transmitter interrupt request; in Synchronous Mode, it is entirely impossible.

Using the Transmit Buffer Interrupt (TBIR) to reload transmit data provides the time necessary to transmit a complete frame for the service routine, as TBUF may be reloaded while the previous data is still being transmitted.

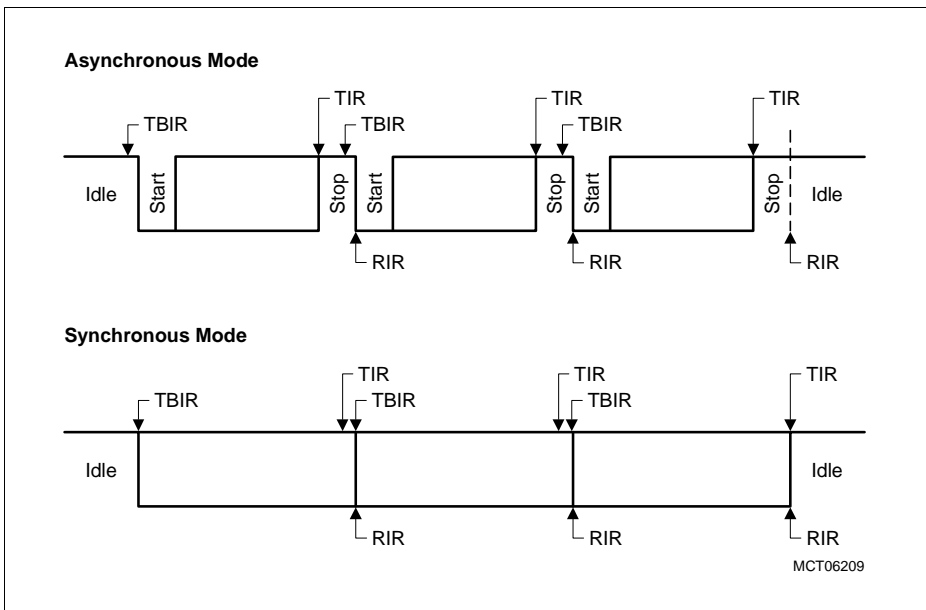


Figure 34-34 ASC Interrupt Generation

As shown in [Figure 34-34](#), TBIR is an early trigger for the reload routine, while TIR indicates the completed transmission. Software using handshake should, therefore, rely on TIR at the end of a data block to ensure that all data has been transmitted.

34.15 Interrupts

8 Interrupt sources are available for the PSI5-S module. For each trigger source of a channel x one interrupt can be selected in register INPx.

Peripheral Sensor Interface with Serial PHY Connection

RDI indicates a receive data interrupt. It is activated when a received frame is moved to Receive Data Register RDR.

RSI indicates a receive frame success interrupt, i.e. non of the interrupts/errors XCRCI, CRCI, TEI, PE, FE, OE, RBI or HDI was detected. The referring interrupt must be enabled in GCR to be considered for RSI. Both RDI and RSI will be issued together in normal use cases where reception is correct.

RBI indicates a receive buffer overrun interrupt. It is activated when a new frame is transferred to a Receive Data Register RDR while the old value was still not read by the host ("overwrite"), i.e. the kernel wants to set interrupt RDI and finds RDI already set.

TPI indicates a transmit interrupt. It is activated when data is completely moved from a SDR to ASC (TPI). Thus it can be used to trigger the next write access to SDR.

TPOI indicates a transfer (send) register (SDR) overrun interrupt. It is set if data is written to SDR while the referring register is locked (occupied), signalled by TPF. The data written is ignored in this case. FOI indicates that the output FIFO was overrun.

In addition the protocol error interrupts are available:

CRCI, XCRC. If CRC interrupt is activated, data is to be treated as invalid according to standard V2.0 2011-06. Note that the wrong data is still available in all buffers. For XCRCI, these frames are always stored in Channel 0 Frame 1.

TEI is issued by the watch dog timer if the timely distance between two frames is too long.

CHCI indicates that the last frame expected for a channel is received.

The interrupt request or the corresponding interrupt set bit (in register INTSET) can trigger the interrupt generation at the selected interrupt node. The service request pulse is generated independently from the interrupt flag in register INTSTATx. The interrupt flag can be cleared by software by writing to the corresponding bit in register INTCLR.

If more than one interrupt source is connected to the same interrupt node pointer (in register INPx all node pointer point to the same line), the requests are combined to one common line.

34.16 Trigger Outputs

Any interrupt source can be used as trigger output TRIGO outside the module. Each TRIGO is connected to the interrupt router. See [Table 34-7 "Service Request Lines of PSI5-S" on Page 34-140](#).

Peripheral Sensor Interface with Serial PHY Connection
34.17 PSI5-S Kernel Registers

This section describes the kernel registers of the PSI5-S module. All PSI5-S kernel register names described in this section will be referenced in other parts of the TC27x User's Manual by the module name prefix "PSI5S_" for the PSI5-S interface.

All registers in the PSI5-S address spaces are reset with the application reset (definition see SCU section "Reset Operation").

The complete and detailed address map of the PSI5-S module is described in [Table 34-5](#) on [Page 34-58](#).

x can take the values 0 ... 7 (PSI5-S channels)

List of Access Protection Abbreviations

- U - User Mode
- SV - Supervisor Mode
- BE - Bus Error
- nBE - no Bus Error
- P - Access Protection, as defined by the ACCEN Register
- E - ENDINIT
- SE - Safety ENDINIT

Reset Classes

All registers belong to reset class 3 except OCS, which belongs to reset class 1.

Table 34-5 Registers Address Space - PSI5-S Kernel Registers

Module	Base Address	End Address	Note
PSI5S	F000 7000 _H	F000 7FFF _H	–

Table 34-6 Registers Overview - PSI5-S Kernel Registers

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
Module Control Registers					
-	reserved	004 _H	BE	BE	
PSI5S_ID	Module Identification Register	008 _H	SV, U	BE	Page 34-63
PSI5S_FDR	Module Fractional Divider	00C _H	SV, U	U, SV,P	Page 34-65
Global Registers					

Peripheral Sensor Interface with Serial PHY Connection
Table 34-6 Registers Overview - PSI5-S Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
PSI5S_FDR T	Fractional Divider Register for Time Stamp Counter	010 _H	SV, U	U, SV,P	Page 34-72
PSI5S_TSC NTA	Module Time Stamp Count Register A	014 _H	SV, U	U, SV,P	Page 34-74
PSI5S_TSC NTB	Module Time Stamp Count Register B	018 _H	SV, U	U, SV,P	Page 34-76
PSI5S_GCR	Global Control Register	01C _H	SV, U	U, SV,P	Page 34-77
PSI5S_NFC	Number of Frames Control Register	020 _H	SV, U	U, SV,P	Page 34-79
PSI5S_FCN T	Number of Frames Count Register	024 _H	SV, U	U, SV,P	Page 34-80
PSI5S_IOC R	Input and Output Control Register	028 _H	SV, U	U, SV,P	Page 34-82
Channel Registers					
-	reserved	02C _H	BE	BE	-
Receive Control Registers					
PSI5S_RCR Ax	Receiver Control Register x	030 _H + x*4	SV, U	U, SV,P	Page 34-83
PSI5S_RCR Bx	Receiver Control Register x	050 _H + x*4	SV, U	U, SV,P	Page 34-85
PSI5S_WDT x	Watch Dog Timer Register x (x=0..7)	070 _H + x*4	SV, U	U, SV,P	Page 34-86
Receive Data and Status Registers					
PSI5S_TSC Rx	Time Stamp Capture Register x	090 _H + x*4	SV, U	BE	Page 34-89
PSI5S_RDS	Receive Data Status Register	0B0 _H	SV, U	BE	Page 34-90
PSI5S_RDR	Receive Data Register	0B4 _H	SV, U	BE	Page 34-94
PSI5S_TSM	Time Stamp Mirror Register	0B8 _H	SV, U	BE	Page 34-95
-	reserved	0BC _H	BE	BE	-
PSI5S_TAR	Target Address Register	0D0 _H	SV, U	BE	Page 34-96
PSI5S_BAR	Base Address Register	0D4 _H	SV, U	U, SV,P	Page 34-97

Peripheral Sensor Interface with Serial PHY Connection
Table 34-6 Registers Overview - PSI5-S Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
-	reserved	0D8 _H - 0EC _H	BE	BE	-
Sync Pulse Control Registers					
PSI5S_PGC x	Pulse Generation Control Register x	0F0 _H + x*4	SV, U	U, SV,P	Page 34-98
PSI5S_CTV x	Channel Trigger Value Register	110 _H + x*4	SV, U	U, SV,P	Page 34-99
Send Registers					
PSI5S_SCR x	Send Control Register x	130 _H + x*4	SV, U	U, SV,P	Page 34-100
PSI5S_SDR x	Send Data Register x	150 _H + x*4	SV, U	U, SV,P	Page 34-103
PSI5S_CD W	CPU Direct Write Register	170 _H	SV, U	U, SV,P	Page 34-104
-	reserved	174 _H - 20C _H	BE	BE	-
ASCRegisters					
PSI5S_CON	Control Register	210 _H	SV, U	U, SV,P	Page 34-105
PSI5S_BG	Baud Rate Timer Reload Register	214 _H	SV, U	U, SV,P	Page 34-110
PSI5S_FD V	Fractional Divider Register	218 _H	SV, U	U, SV,P	Page 34-110
PSI5S_FDO	Fractional Divider for Clock Output	21C _H	SV, U	U, SV,P	Page 34-111
PSI5S_TBU F	Transmit Buffer Register	220 _H	SV, U	U, SV,P	Page 34-112
PSI5S_RBU F	Receive Buffer Register	224 _H	SV, U	BE	Page 34-113
-	reserved	228 _H - 24C _H	BE	BE	-

Peripheral Sensor Interface with Serial PHY Connection
Table 34-6 Registers Overview - PSI5-S Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
PSI5S_WHB CON	Write Hardware Bits Control Register	250 _H	SV, U	U, SV,P	Page 34-10 8
Interrupt Status and Control Registers					
PSI5S_INT OV	Interrupt Overview Reg.	300 _H	SV, U	BE	Page 34-11 4
PSI5S_INTS TATx	Interrupt Status Reg. x	260 _H + x*4	SV, U	BE	Page 34-11 6
PSI5S_INTS ETx	Interrupt Set Register x	280 _H + x*4	SV, U	U, SV,P	Page 34-12 0
PSI5S_INTC LRx	Interrupt Clear Reg. x	2A0 _H + x*4	SV, U	U, SV,P	Page 34-12 2
PSI5S_INTE Nx	Interrupt Enable Reg. x	2C0 _H + x*4	SV, U	U, SV,P	Page 34-12 4
PSI5S_INPx	Int. Node Pointer Reg. x	2E0 _H + x*4 _H	SV, U	U, SV,P	Page 34-12 6
PSI5S_INTS TATG	Interrupt Status Reg. Glob.	304 _H	SV, U	BE	Page 34-12 8
PSI5S_INTS ETG	Interrupt Set Register Glob.	308 _H	SV, U	U, SV,P	Page 34-13 1
PSI5S_INTC LRG	Interrupt Clear Reg. Glob.	30C _H	SV, U	U, SV,P	Page 34-13 3
PSI5S_INTE NG	Interrupt Enable Reg. Glob.	310 _H	SV, U	U, SV,P	Page 34-13 5
PSI5S_INP G	Int. Node Pointer Reg. Glob	314 _H	SV, U	U, SV,P	Page 34-13 7
-	reserved	318 _H - 3C8 _H	BE	BE	-
BPI Registers					
PSI5S_CLC	Clock Control Register	000 _H	SV, U	SV, E, P	Page 34-64
PSI5S_OCS	OCDS Contr. + Stat. Reg.	3CC _H	U, SV	SV, P	Page 34-66
PSI5S_ACC EN0	Access Enable Register 0	3D0 _H	U, SV	SV, SE	Page 34-67

Peripheral Sensor Interface with Serial PHY Connection

Table 34-6 Registers Overview - PSI5-S Kernel Registers (cont'd)

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
PSI5S_ACC EN1	Access Enable Register 1	3D4 _H	U, SV	SV, SE	Page 34-68
PSI5S_KRS T0	Reset Control Register 0	3D8 _H	U, SV	SV, E, P	Page 34-69
PSI5S_KRS T1	Reset Control Register 1	3DC _H	U, SV	SV, E, P	Page 34-70
PSI5S_KRS TCLR	Reset Status Clear Reg.	3E0 _H	U, SV	SV, E, P	Page 34-71
-	reserved	3E4 _H - FFC _H	BE	BE	-

1) The absolute register address is calculated as follows:
 Module Base Address ([Table 34-5](#)) + Offset Address (shown in this column)

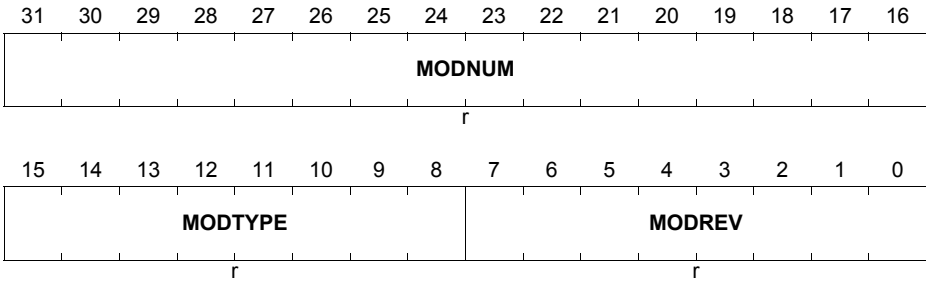
34.17.1 Module Control

Peripheral Sensor Interface with Serial PHY Connection
Module Identification Register

The PSI5-S Module Identification Register ID contains read-only information about the module version.

ID

Module Identification Register (008_H) **Reset Value: 00D3 C0XX_H**



Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MODTYPE	[15:8]	r	Module Type This bit field defines the module as a 32-bit module: C0 _H
MODNUM	[31:16]	r	Module Number Value This bit field defines the module identification number for the PSI5-S: 00D3 _H

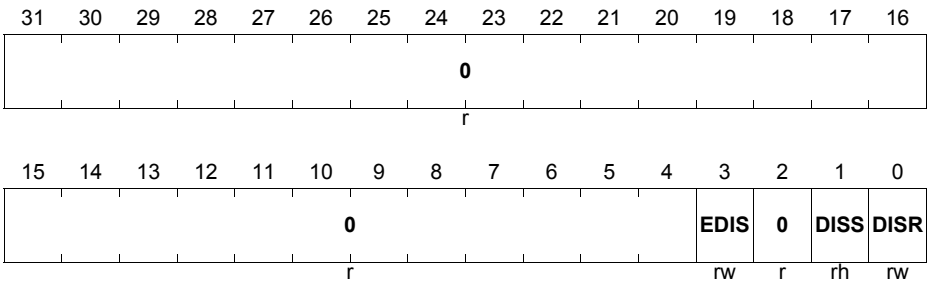
Clock Control Register (CLC)

The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI_FPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the module clock, sleep mode and disable mode for the module.

Peripheral Sensor Interface with Serial PHY Connection

CLC
Clock Control Register

 (000_H)

 Reset Value: 0000 0003_H


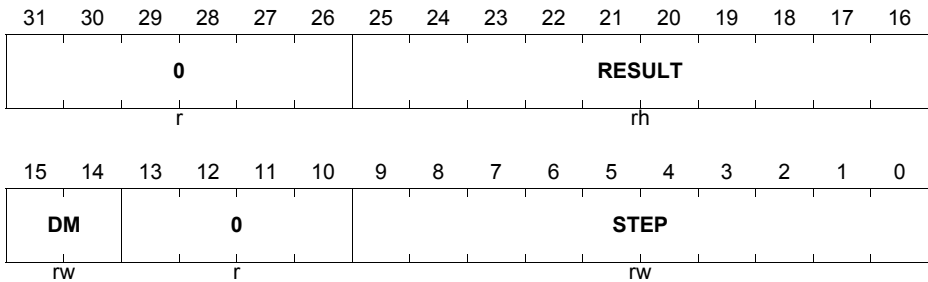
Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module. <i>Note: This bit disables the kernel clocks f_{CLC_PSI5-S} and the ASC clock f_{BAUD2}.</i>
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
EDIS	3	rw	External Sleep Mode Request Disable Bit Used to control module's sleep mode. <i>Note: If this bit is cleared the kernel clock f_{CLC_PSI5-S} and the ASC clock f_{BAUD2} are disabled during System Sleep Mode.</i>
0	[31:4], 2	r	Reserved Read as 0; should be written with 0.

Note: After a hardware reset operation, the clocks f_{PSI5-S} and f_{BAUD2} are switched off and the PSI5-S module and the included ASC module are disabled (DISS set).

Fractional Divider Register

The Fractional Divider Register controls the clock f_{PS5-S} .

This bit field only controls the kernel clock f_{PSI5-S} and not the sampling clock f_{BAUD2} .

Peripheral Sensor Interface with Serial PHY Connection
FDR
PSI5-S Fractional Divider Register
(00C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
STEP	[9:0]	rw	Step Value Reload or addition value for RESULT.
DM	[15:14]	rw	Divider Mode DM selects normal or fractional divider mode. 00 _B Fractional divider is switched off; no output clock is generated. The Reset External Divider signal is 1. RESULT is not updated (default after System Reset). 01 _B Normal Divider Mode selected. 10 _B Fractional Divider Mode selected. 11 _B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	[25:16]	rh	Result Value Bit field for the addition result.
0	[13:10], [31:26]	r	Reserved Read as 0; should be written with 0.

OCDS Control and Status Register (OCS)

The OCDS Control and Status (OCS) register is cleared by Debug Reset. The OCS register can only be written when the OCDS is enabled. If OCDS is being disabled, the OCS register value will not change. When OCDS is disabled the OCS suspend control is ineffective. Write access is 32 bit wide only and requires Supervisor Mode.

Peripheral Sensor Interface with Serial PHY Connection
OCS
OCDS Control and Status
(3CC_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		SUS STA	SUS _P	SUS				0							
r		rh	w	rw				r							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															
r															

Field	Bits	Type	Description
SUS	[27:24]	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS) 0 _H Will not suspend 1 _H Hard suspend. Clock is switched off immediately. 2 _H Soft suspend option A (Suspend after end of current send or receive transfers, if any are ongoing) others , reserved <i>Note: Suspend disables the kernel clocks f_{CLC_PSI5-S} and the ASC clock f_{BAUD2}.</i>
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended
0	[23:0], [31:30]	r	Reserved Read as 0; must be written with 0.

Access Enable Register (ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, OCS, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

Peripheral Sensor Interface with Serial PHY Connection

TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... , EN31 -> TAG ID 011111B.

ACCEN0
Access Enable Register 0
(3D0_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN3	EN3	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN1	EN1	EN1	EN1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN1	EN1	EN1	EN1	EN1	EN1	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
5	4	3	2	1	0										
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
ENn (n = 0-31)	n	rw	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register (ACCEN1)

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000B, EN1 -> TAG ID 100001B, ... , EN31 -> TAG ID 111111B.

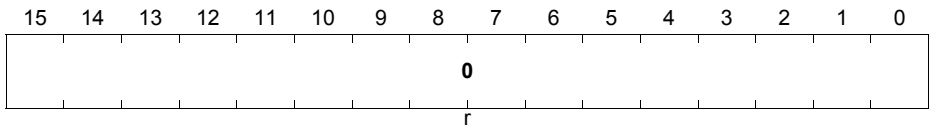
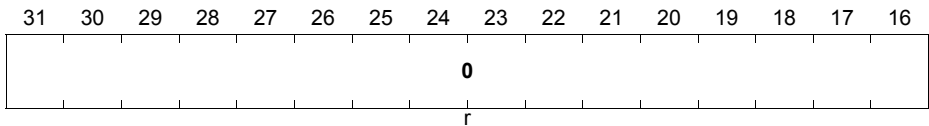
Peripheral Sensor Interface with Serial PHY Connection

ACCEN1

Access Enable Register 1

(3D4_H)

Reset Value: 0000 0000_H



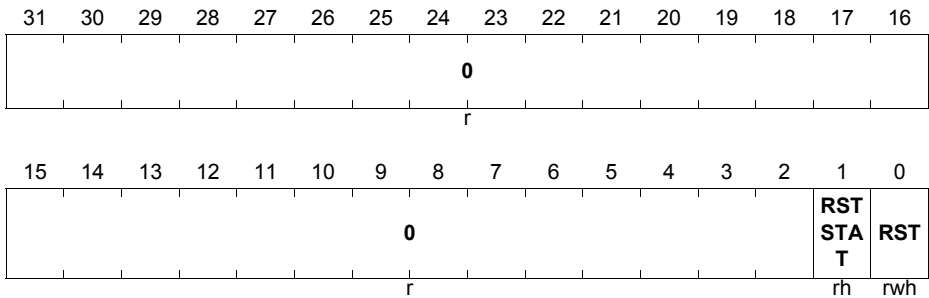
Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLE.CLR register bit.

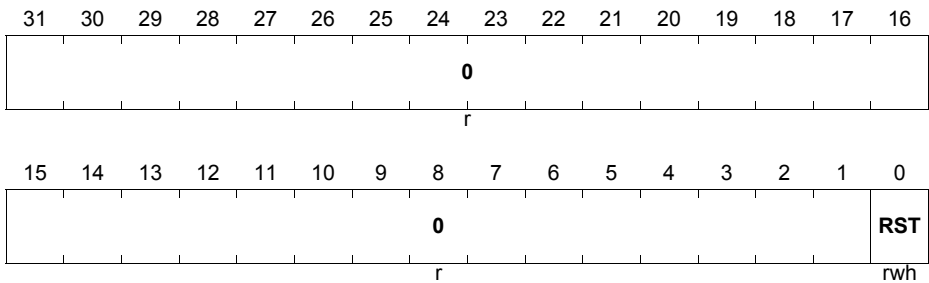
During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

Peripheral Sensor Interface with Serial PHY Connection
KRST0
Kernel Reset Register 0
(3D8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
RSTSTAT	1	rh	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits. 0 _B No kernel reset was executed 1 _B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

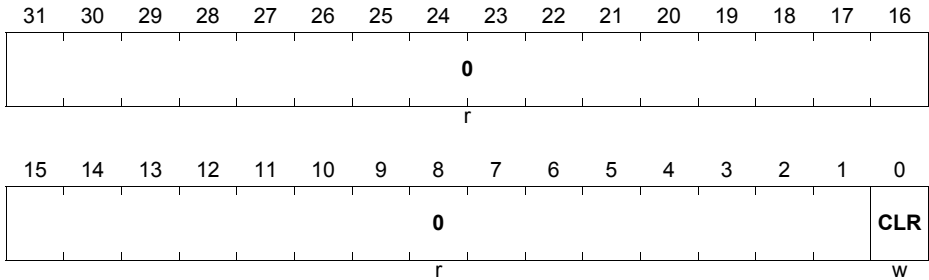
Peripheral Sensor Interface with Serial PHY Connection
KRST1
Kernel Reset Register 1
(3DC_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Status Clear Register (KRSTCLR)

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

Peripheral Sensor Interface with Serial PHY Connection

KRSTCLR
Kernel Reset Status Clear Register (3E0_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection
Fractional Divider Register for Time Stamp

The PSI5-S Module Time Stamp Predivider Register contains the pre divider that defines the time resolution of the Time Stamp Registers TSCNTA/B and the Sync Pulse Time Base Counter SBC. It divides f_{PSI5-S} by a factor as given in [Equation \(34.1\)](#) and [Equation \(34.2\)](#). It contains as well the bits for reset control of the Time Stamp Counters.

FDRT
Fractional Divider Register for Time Stamp (010_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0		ECE B	ECE A	ECS			RESULT									
r	rw		rw		rw			rh								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DM			0			STEP										
rw			r			rw										

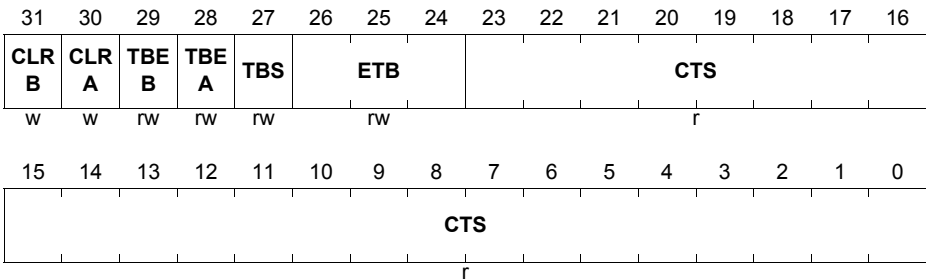
Field	Bits	Type	Description
STEP	[9:0]	rw	Step Value Reload or addition value for RESULT.
DM	[15:14]	rw	Divider Mode DM selects normal or fractional divider mode. 00 _B Fractional divider is switched off; no output clock is generated. 01 _B Normal Divider Mode selected. 10 _B Fractional Divider Mode selected. 11 _B Fractional divider is switched off; no output clock is generated. RESULT is not updated.
RESULT	[25:16]	rh	Result Value Bit field for the addition result.
ECS	[28:26]	rw	External Time Stamp Clear Source Select Selects the external trigger line that clears the global Time Stamp Counters TSCNTA/B.CTS if this is enabled by ECEA/ECEB. 000 _B TRIG0 001 _B TRIG1 ... _B ... 111 _B TRIG7

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
ECEA	29	rw	External Time Stamp Clear Enable A Enables the external trigger line selected by ECS to clear the global Time Stamp Counter TSCNTA.CTS on rising edge of the external trigger. 0 _B disabled 1 _B enabled
ECEB	30	rw	External Time Stamp Clear Enable B Enables the external trigger line selected by ECS to clear the global Time Stamp Counter TSCNTB.CTS on rising edge of the external trigger. 0 _B disabled 1 _B enabled
0	31,[13:10]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection
Module Time Stamp Count Register A

This PSI5-S Module Time Stamp Counter Register contains read-only information about the current time given in clock cycles of f_{TS} or f_{TRIGx} since last reset while module was clocked. It also contains the control bits for selection of input clock and reset of this counter.

TSCNTA
Time Stamp Count Register A (014_H) Reset Value: 0000 0000_H


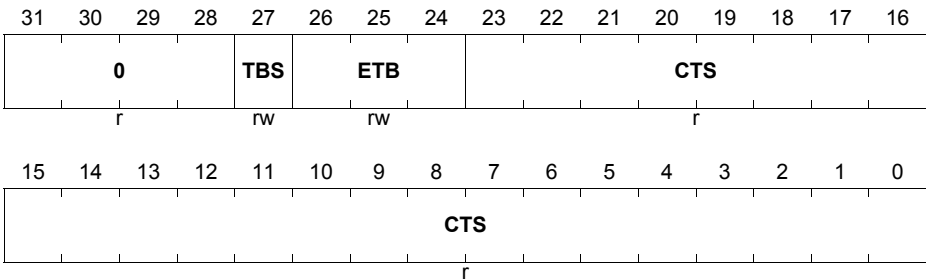
Field	Bits	Type	Description
CTS	[23:0]	r	Current Time Stamp for the Module This bit field shows the current time stamp.
ETB	[26:24]	rw	External Time Base Select Selects the external clock line for counter CTS if TBS is set. 000 _B TRIG0 001 _B TRIG1 ... _B ... 111 _B TRIG7
TBS	27	rw	Time Base Select This bit selects the clock source for CTS 0 _B Internal, CTS counts in clock cycles of f_{TS} 1 _B External, CTS counts in clock cycles received on external clock line selected by ETB
TBEA	28	rw	Time Base Enable TSCNTA This bit starts/stops TSCNTA.CTS 0 _B CTS stopped (w/o clear of CTS) 1 _B CTS started (w/o clear of CTS)

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
TBEB	29	rw	Time Base Enable TSCNTB This bit starts/stops TSCNTB.CTS 0 _B CTS stopped (w/o clear of CTS) 1 _B CTS started (w/o clear of CTS)
CLRA	30	w	Clear Time Stamp Counter A This bit clears TSCNTA.CTS. TSCNTA.CTS counts on, starting from 0.
CLRB	31	w	Clear Time Stamp Counter B This bit clears TSCNTB.CTS. TSCNTB.CTS counts on, starting from 0.

Peripheral Sensor Interface with Serial PHY Connection
Module Time Stamp Count Register B

This PSI5-S Module Time Stamp Counter Register contains read-only information about the current time given in clock cycles of f_{TS} or f_{TRIGx} since last reset while module was clocked. It also contains the control bits for selection of input clock and reset of this counter.

TSCNTB
Time Stamp Count Register B (018_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
CTS	[23:0]	r	Current Time Stamp for the Module This bit field shows the current time stamp.
ETB	[26:24]	rw	External Time Base Select Selects the external clock line for counter CTS. 000 _B TRIG0 001 _B TRIG1 ... _B ... 111 _B TRIG7
TBS	27	rw	Time Base Select This bit selects the clock source for CTS 0 _B Internal, CTS counts in clock cycles of f_{TS} 1 _B External, CTS counts in clock cycles received on external clock line selected by ETB
0	[31:28]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection
Global Control Register

The Global Control Register defines module wide settings.

The first 8 bits define, which error flags are regarded at RSI. RSI indicates that the referring frame is free of the selected errors. Each of these errors can be selected: CRCI, XCRCI, TEI, PE, FE, OE, RBI, HDI.

GCR
Global Control Register
(01C_H)
Reset Value: 0000 001F_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ASC	0			IDT				CEN 7	CEN 6	CEN 5	CEN 4	CEN 3	CEN 2	CEN 1	CEN 0
rw	r			rw				rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETC 7	ETC 6	ETC 5	ETC 4	ETC 3	ETC 2	ETC 1	ETC 0	HDI	RBI	OE	FE	PE	TEI	XCRCI	CRCI
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
CRCI	0	rw	CRCI is selected if bit is set.
XCRCI	1	rw	XCRCI is selected if bit is set.
TEI	2	rw	TEI is selected if bit is set.
PE	3	rw	PE is selected if bit is set.
FE	4	rw	FE is selected if bit is set.
OE	5	rw	OE is selected if bit is set.
RBI	6	rw	RBI is selected if bit is set.
HDI	7	rw	HDI is selected if bit is set.

Peripheral Sensor Interface with Serial PHY Connection

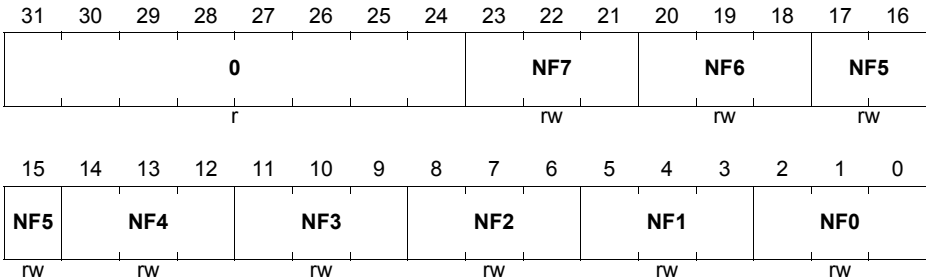
Field	Bits	Type	Description
ETC_x (x = 0-7)	8+x	rw	Enable Channel Trigger Counter CTV_x.CTC This bit enables CTV _x .CTC. The bits ETC0 ..x can be set with one write access to synchronously start all counters. This is required for proper sync pulse staggering. If set, CTC _x counts on, starting from its current value. CTC _x can be written only if ETC _x is cleared (stopped).
CEN_x (x = 0-7)	16+x	rw	Enable Channel x This bit enables PSI5-S Channel x. If cleared, all internal state machines of the receiver and the sender are forced to default idle state while all registers can be read and written. Used for configuration of a channel. Frames received for a disabled channel are copied to ChID 0, FID 1 with original IDs.
IDT	[27:24]	rw	Idle Time (GLOBAL VALUE FOR ALL CHANNELS) determines the number of stop bits in addition to the stop bit of the last UART Frame that is required for SOF detection. (IDT-1) idle bit times are allowed/tolerated within one Packet Frame. Default is IDT = 0, i.e. back to back transfer. 0000 _B 1 bit time idle 0001 _B 2 bit times idle 1111 _B 16 bit times idle
ASC	31	rw	ASC only Mode is selected if bit is set. The ASC registers are fully controllable by SW via SPB. If cleared, the ASC is controlled by the message reassembly unit and the message generation unit. RBUF and TBUF are no longer writable by SW and interrupts are handled by the message reassembly block automatically.
0	[30:28]	r	Reserved Read as 0; should be written with 0.

Number of Frames Control Register

Peripheral Sensor Interface with Serial PHY Connection

This register contains the number of Packet Frames expected after a Sync Pulse. It stores the compare values for the counters in register FCNT.

See **“FCNT” on Page 34-80**.

NFC
Number of Frames Control Register (020_H)
Reset Value: 0024 9249_H


Field	Bits	Type	Description
NF_x (x = 0-7)	[3*x+2: 3*x]	rw	Number of expected Frames on Channel x 000 _B reserved, do not use! Reads back 1. 001 _B 1 ... _B ... 110 _B 6 111 _B reserved, do not use! Reads back 6.
0	[31:24]	r	Reserved Read as 0; should be written with 0.

Number of Frames Count Register

Peripheral Sensor Interface with Serial PHY Connection

This register contains the number of Packet Frames actually received after the last Sync Pulse. A Sync Pulse on channel x will clear FCx.

If FCNT.FCx equals NFC.NFx interrupt CHCI is issued and FCNT.FCx will roll over to '1' with the reception of the next recoverable message on channel x. The roll over happens in any case, even if no Sync Pulse clears FCx after CHCI.

If RCRAx.WDMS is set, NFC and FCNT affect the operation of the watch dog timer. See ["Watch Dog Timers" on Page 34-18](#).

NFC and FCNT also affect the generation of interrupt CHCI. See ["Interrupts for DMA support" on Page 34-36](#).

FCNT

Frame Counter Register (024_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFC LR7	NFC LR6	NFC LR5	NFC LR4	NFC LR3	NFC LR2	NFC LR1	NFC LR0	FC7			FC6			FC5	
w	w	w	w	w	w	w	w	r			r			r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FC5		FC4		FC3			FC2			FC1			FC0		
r		r		r			r			r			r		

Field	Bits	Type	Description
FCx (x = 0-7)	[3*x+2: 3*x]	r	Frame Counter for Channel x Contains the number of received frames on Channel x. Copied to RDR.FID if RCRAx.FIDS is set. 000 _B 0 (after reset, Sync (if WDMS is set only) or setting NFCLR _x) 001 _B 1 ... _B ... 110 _B 6 111 _B not valid
NFCLR_x (x = 0-7)	x+24	w	Clear Number of Frame Counter for Channel x Clears the referring counter FCNT.FCx. Intended for use during recovery from TEI. If set while a frame is being received, this action results in FCx = '1'. Thus FCx will never be '0' when RDI/RSI signal a new receive frame. Use with care!

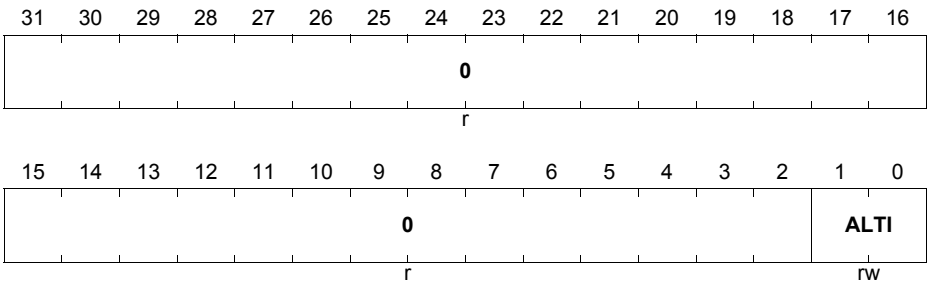
34.17.2 Input and Output Control

Input and Output Control Register Functions

Peripheral Sensor Interface with Serial PHY Connection

The Input and Output Control Register IOCRx determines for the PSI5-S channel x:

- the alternate input for the receiver

IOCR
Input and Output Control Register (028_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
ALTI	[1:0]	rw	Alternate Input Select Selects the alternate input for RX of the ASC: 00 _B Alternate Input 0 selected 01 _B Alternate Input 1 selected ... _B ... 11 _B Alternate Input 3 selected
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection
34.17.3 Receiver Control Registers
Receiver Control Register Ax

The Receiver Control Registers RCRAx contain control bits/bit fields that are related to the PSI5-S receiver operation. It enables the channel and determines the use of CRC or parity. CENx must be clear for write access.

RCRAx (x = 0-7)

Receiver Control Register Ax (030_H+x*4) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			UFC5		UFC4		UFC3		UFC2		UFC1		UFC0		
r			rw		rw		rw		rw		rw		rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				WD MS	FIDS	TST S	TSP	TSE N	CRC 5	CRC 4	CRC 3	CRC 2	CRC 1	CRC 0	
r				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Field	Bits	Type	Description
CRCy (y = 0-5)	y	rw	CRC or Parity Selection If set, a 3 bit CRC checksum is expected for the PSI5-S channel x in slot y. Else, 1 bit Parity is assumed. This bit field is looked up before potential modification of FID according to FIDS. 0 _B 1 Parity Bit is configured (default) 1 _B 3 CRC bits are configured
TSEN	6	rw	Time Stamp Enable Enables the time stamping for channel x 0 _B off (default) TSCRx and thus TSM are forced to 0x0000 0000! 1 _B on, see TSP and TSTS
TSP	7	rw	Time Stamp Select For non recoverable Packets (stored in ChID '0', FID '1' with original IDs); TSCNTA is captured in TSM and not in TSCR0 (independent from TSP). 0 _B TSCNTA.CTS is captured in TSCRx 1 _B TSCNTB.CTS is captured in TSCRx

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
TSTS	8	rw	<p>Time Stamp Trigger Select</p> <p>0_B On Sync Pulse. TSCRx is updated on Sync Pulse. TSM is updated from TSCRx on Packet Frame reception after ChID extraction. For non recoverable Packets (stored in ChID '0', FID '1' with original IDs); TSCR0 is not updated and TSM is updated with the current value of TSCNTA (independent from TSP). This happens at the time the FSM assumes the Packet Frame to be non recoverable. This allows following good packets to be time stamped correctly.</p> <p>1_B On any Frame. TSCRx and TSM are updated simultaneously with the current value of TSCNTA/B depending from RCRAx.TSP on Packet Frame reception after ChID extraction. (RDI) This allows SW to read the time of reception on the referring channel from TSCRx. For non recoverable Packets (stored in ChID '0', FID '1' with original IDs), TSM is updated with the current value of TSCNTA (independent from TSP and TSEN). This happens at the time the FSM assumes the Packet Frame to be non recoverable. TSCR0 is updated only if RCRA0.TSTS is set ('1')</p>
FIDS	9	rw	<p>Frame ID Select</p> <p>0_B Frame ID is updated from Packet Frame Header (sync mode). Channel 0 should have FIDS cleared. This avoids that the module overwrites non recoverable messages with Transceiver Messages. Note that non recoverable messages are always stored in ChID '0', FID '1' with original IDs.</p> <p>1_B Frame ID is a rolling number 0 .. 5 copied from FCNT.(FCx-1) (async mode) Non recoverable messages are still stored in ChID '0', FID '1' with original IDs.</p>

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
WDMS	10	rw	Watch Dog Timer Mode Select 0 _B Watch Dog Timer is restarted on reception of each recoverable frame on Channel x (async mode). 1 _B Watch Dog Timer is restarted on Sync Pulse and stopped at reception of the last frame configured in NFC.NFx.(sync mode)
UFCy (y = 0-5)	[2*y+17 :2*y+16]	rw	UART Frame Count per Packet Frame in Slot y This bit field defines the number of UART Frames per Packet Frame that are expected for Slot y. This bit field is looked up before potential modification of FID according to FIDS. 00 _B 3 UART Frames 01 _B 4 UART Frames 10 _B 5 UART Frames 11 _B 6 UART Frames
0	[31:28], [15:11]	r	Reserved Read as 0; should be written with 0.

Receiver Control Register Bx

The Receiver Control Registers RCRBx configures the number of payload bits and implicitly the number of UART Frames (Bytes) each of the up to 6 slots in channel x. CENx must be clear for write access.

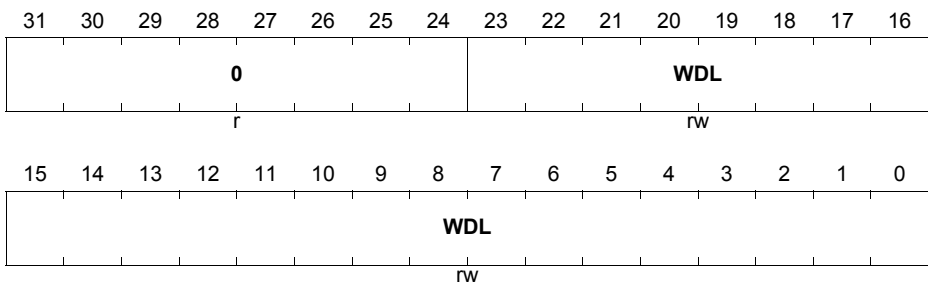
RCRBx (x = 0-7)

Receiver Control Register Bx (050_H+x*4) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		PDL5				PDL4				PDL3					
r		rw				rw				rw					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDL3		PDL2				PDL1				PDL0					
rw		rw				rw				rw					

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
PDL_y (y = 0-5)	[5*y+4: 5*y]	rw	Payload Data Length PDL determines the number of bits in a Packet Frame frame for Slot y. It also determines the position of the CRC/Parity bit. E.g. 8 defines 8 data bits on position [7:0]. On bit position 8 the CRC/Parity is located. See Figure 34-4 . If PDL _y is cleared ('0') no frame is expected for this slot. Packet Frames received for a slot with PDL = '0' are copied to ChID 0, FID 1 with original IDs without further processing. This bit field is looked up before potential modification of FID according to RCRAx.FIDS. 00000 _B No Frame expected! 00001 _B 8 bits 01000 _B 8 bits 01001 _B 9 bits 11100 _B 28 bits 11111 _B 28 bits
0	[31:30]	r	Reserved Read as 0; should be written with 0.

Watch Dog Timer Register
WDTx (x = 0-7)
Watch Dog Timer Register x (070_H+x*4) Reset Value: 0000 0000_H


Peripheral Sensor Interface with Serial PHY Connection

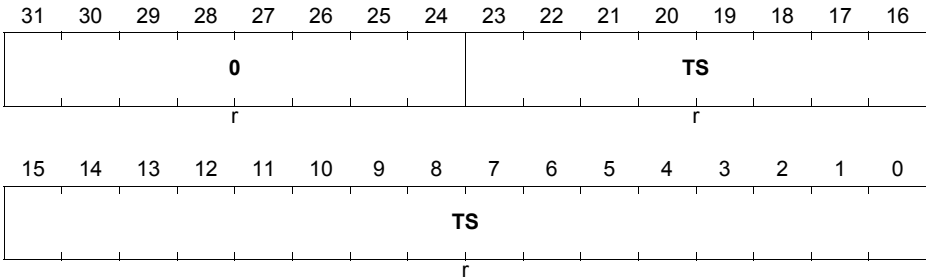
Field	Bits	Type	Description
WDL	[23:0]	rw	Watch Dog Timer Limit for channel x. If no watch dog is needed, WDL is cleared, the internal watch dog timer is stopped and no check is performed. CENx must be clear for write access.
0	[31:24]	r	Reserved Read as 0; should be written with 0.

34.17.4 Receive Data and Status Registers

Start of Pulse Time Stamp Capture Register TSCRx

Peripheral Sensor Interface with Serial PHY Connection

If RCRAx.TSEN is set, the time stamp is captured in this register each time a sync pulse is sent from the ASC FIFO to the ASC TX buffer for channel x. RCRAx.TSP selects, if TSCNTA or B is used.

TSCRx (x = 0-7)
Capture Register TSCRx **(090_H+x*4)** **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
TS	[23:0]	r	Time Stamp of the last sync pulse sent for channel x.
0	[31:24]	r	Reserved Read as 0; should be written with 0.

Receive Status Registers RDS

Peripheral Sensor Interface with Serial PHY Connection

The Receive Status Register RDS shows the status of a received data frame.

The internal receive buffer is always cleared (0x0000 0000_H) at each frame start. Thus unused bits are always read as zero.

The bits CRCl, XCRCI, HDI, PE, FE, OE, TEI and RBI reflect, whether the conditions are met to issue an interrupt signal and set the referring interrupt status bit in INTSTAT or CON (ASC sub module) for the latest frame. Thus it is independent from the old status of the referring sticky bit in INTSTAT or CON before latest frame reception.

RDS

Receive Status Register (0B0H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PFC			AFC			CID			FID			RBI	TEI	OE	
r			r			r			r			r	r	r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE	PE	HDI	ERR 1	ERR 0	CRCl	CRC 2	CRC 1	CRC 0	XCRC I	XCRC 5	XCRC 4	XCRC 3	XCRC 2	XCRC 1	XCRC 0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
XCRCy (y = 0-5)	y	r	XCRC CRC of last Packet Frame. XCRC0 is on bit position 0.
XCRCI	6	r	XCRC Error Flag This bit is set if the CRC check on the enveloping Packet Frame fails including the case where XCRC check can not be performed (non recoverable frames) see Chapter 34.5.1.1 . 0 _B correct XCRC 1 _B wrong XCRC
CRCy (y = 0-2)	7+y	r	CRC of last frame. CRC0 / Parity is on bit position 7. If Parity is used, CRC1/2 are always 0.

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
CRCI	10	r	CRC Error Flag This bit is set if the CRC or Parity check on the transported PSI5 frame fails. 0 _B correct CRC/Parity 1 _B wrong CRC/Parity
ERR0	11	r	Error signalling Flag 0 This bit represents the status of the error signalling flag Err0 in the enveloping Packet Frame.
ERR1	12	r	Error signalling Flag 1 This bit represents the status of the error signalling flag Err1 in the enveloping Packet Frame.
HDI	13	r	Header Error Signalled Flag This bit is set if at least one of the error signalling flags in the enveloping Packet Frame Err0 and Err1 is set. 0 _B (Err0 OR Err1) = false (0) 1 _B (Err0 OR Err1) = true (1)
PE	14	r	ASC Parity Error Flag This bit is set if the error flag signalling a parity error was set during reception of one of the ASC Bytes transporting this PSI5 frame.
FE	15	r	ASC Framing Error Flag This bit is set if the error flag signaling a framing error was set during reception of one of the ASC Bytes transporting this PSI5 frame.
OE	16	r	ASC Overrun Error Flag This bit is set if the error flag signaling an overrun error was set during reception of one of the ASC Bytes transporting this PSI5 frame.

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
TEI	17	r	<p>Time Error Flag</p> <p>This bit is set if the watch dog timer expired. Depending from RCRAx.WDMS either the distance between two RDIs is longer than specified in WDL or it expired without reception of CHCI in time. I.e. the time from issuing the sync pulse to reception of the last expected frame configured in NFC.NFx was too long.</p> <p>If WDMS is in synchronous mode, all frames after TEI have TEI set in RDS until either CHCI is issued or INSTATx.TEI is cleared by SW. RDS.TEI flag is independently from INTENx.TEI.</p> <p>0_B no error 1_B error</p>
RBI	18	r	<p>Receive Buffer Overflow Flag</p> <p>This bit is set after a frame has been received while the old one was not read from RDR. I.e. the kernel wants to set interrupt RDI and finds RDI already set. The old data is overwritten by the new data.</p> <p>0_B No overflow 1_B Overflow</p>
FID	[21:19]	r	<p>Frame ID (Frame Number)</p> <p>See bit RCRAx.FIDS for actual content.</p> <p>000_B Slot 0 001_B Slot 1 ..._B ... 101_B Slot 5 110_B not valid, frame is copied to ChID 0, FID 1 with original IDs 111_B not valid, frame is copied to ChID 0, FID 1 with original IDs</p>
CID	[24:22]	r	<p>Channel ID (Channel Number)</p> <p>000_B channel 0 001_B channel 1 ..._B ... 110_B channel 6 111_B channel 7</p>
AFC	[27:25]	r	<p>Actual UART Frame Count</p> <p>This bit field shows the number of UART frames actually received. This is used to further analyze non recoverable frames by SW or during debugging.</p>

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
PFC	[31:28]	r	<p>Packet Frame Count</p> <p>For data consistency, RDR, RDS and TSM are tagged with a Packet Frame count value. For each channel there is a separate internal packet counter. This allows to read a triplet of RDR, RDS and TSM from the system memory without looking at interrupts. If the PFC is identical all three values belong together. Otherwise they need to be read again until PFC is identical.</p>

Receive Data Registers RDR

Peripheral Sensor Interface with Serial PHY Connection

The Receive Data Register RDR shows the data content of a received data frame. If Messaging bits are used, they are located on bit position [1:0].

The internal receive buffer is always cleared (0x0000 0000_H) at each frame start. Thus unused bits are always read as zero.

RDR
Receive Data Register
(0B4H)
Reset Value: 0000 0000_H

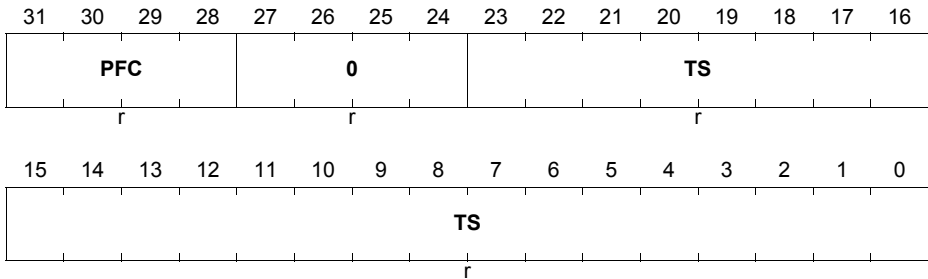
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
PFC				RD2 7	RD2 6	RD2 5	RD2 4	RD2 3	RD2 2	RD2 1	RD2 0	RD1 9	RD1 8	RD1 7	RD1 6	
r				r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RD1 5	RD1 4	RD1 3	RD1 2	RD1 1	RD1 0	RD9	RD8	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	

Field	Bits	Type	Description
RDy (y = 0-27)	y	r	PSI5 Receive Data of last frame. D0 is on bit position 0.
PFC	[31:28]	r	Packet Frame Count For data consistency, RDR, RDS and TSM are tagged with a Packet Frame count value. For each channel there is a separate internal packet counter. This allows to read a triplet of RDR, RDS and TSM from the system memory without looking at interrupts. If the PFC is identical all three values belong together. Otherwise they need to be read again until PFC is identical.

Time Stamp Mirror Register TSM

Peripheral Sensor Interface with Serial PHY Connection

Each time a new frame is received, the value from TSCRx for the referring channel is mirrored here. This register is updated, after XCRC was checked ok, as the referring channel number can not be securely determined before.

TSM
Time Stamp Mirror Register TSM (0B8_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
TS	[23:0]	r	Time Stamp of the last sync pulse sent on channel x.
PFC	[31:28]	r	Packet Frame Count For data consistency, RDR, RDS and TSM are tagged with a Packet Frame count value. For each channel there is a separate internal packet counter. This allows to read a triplet of RDR, RDS and TSM from the system memory without looking at interrupts. If the PFC is identical all three values belong together. Otherwise they need to be read again until PFC is identical.
0	[27:24]	r	Reserved Read as 0; should be written with 0.

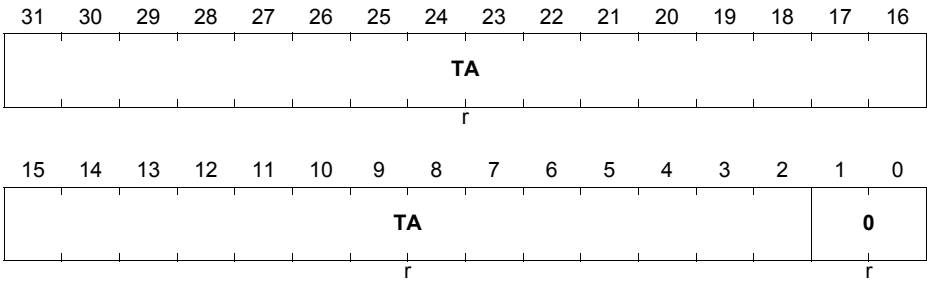
Target Address Registers TAR

Peripheral Sensor Interface with Serial PHY Connection

TAR contains the address that is to be copied by the first DMA into the target address register of the second DMA. This is required for the use case with 2 DMAs. See **“DMA Support” on Page 34-27**

TAR

Target Address Register (0D0_H) Reset Value: 0000 0000_H



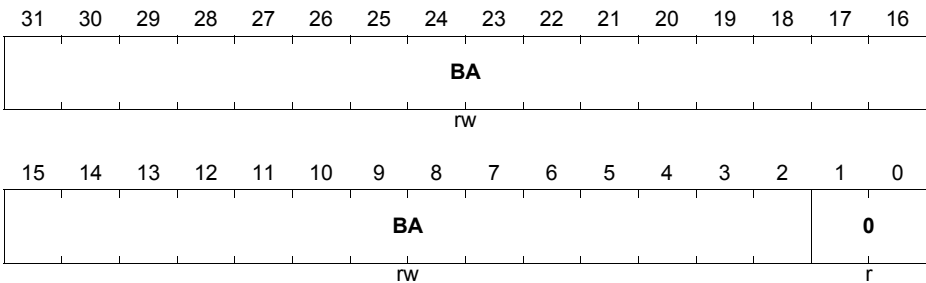
Field	Bits	Type	Description
0	[1:0]	r	Reserved Read as 0; should be written with 0.
TA	[31:2]	r	Target Address Contains the upper 30 bit of the target address for the next DMA transfer. The 32 bit target address must be word aligned. Thus the 2 LSBs are fixed to 0. It is updated each time a new Packet Frame is received completely.

Base Address Registers BAR

Peripheral Sensor Interface with Serial PHY Connection

BAR contains the address in the system memory where the PSI5-S buffer for all channels is built up.

BAR must be configured by the application for an address space that is big enough for all channels that are enabled. TAR will wrap around after the upper limit (0xFFFF FFFF) if adding the offsets (based on ChID and FID) exceeds this limit.

BAR
Base Address Register (0D4_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
0	[1:0]	r	Reserved Read as 0; should be written with 0.
BA	[31:2]	rw	Base Address Contains the upper 30 bits of the base address for the DMA transfers. The 32 bit base address must be word aligned. Thus the 2 LSBs are fixed to 0.

34.17.5 Sync Pulse Control
Pulse Generation Control Registers PGCx

The Pulse Generation Control Register PGC contains control data for the sync pulse generation. It contains as well the trigger control bits required for sending data from the PSI5-S module to the sensor / external PHY.

Peripheral Sensor Interface with Serial PHY Connection
PGCx (x = 0-7)
Pulse Generation Control Register x(0F0_H+x*4)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ETE	ETS			PTE		ETB	
r								rw	rw			rw		rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBS		0		ATXCMD				0			TXCMD				
rw		r		rw				r			rw				

Field	Bits	Type	Description
TXCMD	[4:0]	rw	TX Command Defines the value that is copied to the ASC FIFO for coding a '0'.
ATXCMD	[12:8]	rw	Alternate TX Command Defines the value that is copied to the ASC FIFO for the alternate pulse width i.e. for coding a '1'.
TBS	15	rw	Time Base Select This bit selects the clock source for CTVx 0_B Internal, CTV counts in clock cycles of f_{TS} 1_B External, CTV counts in clock cycles of f_{TRIGx} according to the setting of bit ETB.
ETB	[18:16]	rw	External Time Base Select Selects the external clock line for counter CTVx. 000_B TRIG0 001_B TRIG1 \dots_B \dots 111_B TRIG7
PTE	19	rw	Periodic Trigger Enable Periodic trigger is defined by CTVx. Should be 0 if ETE is set. 0_B disabled 1_B enabled

Peripheral Sensor Interface with Serial PHY Connection

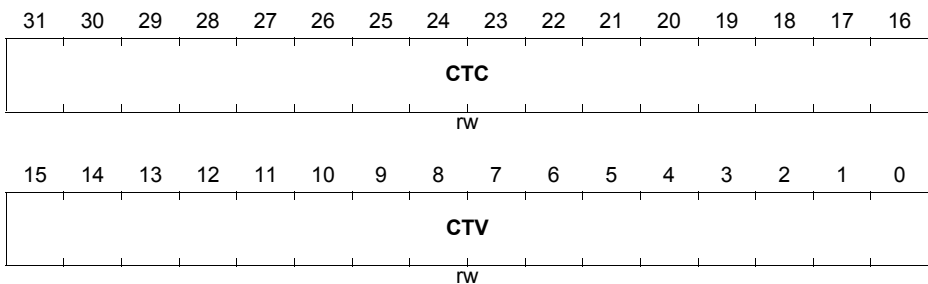
Field	Bits	Type	Description
ETS	[22:20]	rw	External Trigger Select Selects the external trigger line for pulse generation (e.g. angle synchronous). 000 _B TRIG0 001 _B TRIG1 ... _B ... 111 _B TRIG7
ETE	23	rw	External Trigger Enable "Angle sync. trigger", external line is selected by ETS. Should be 0 if PTE is set. 0 _B disabled 1 _B enabled
0	[31:24], [14:13], [7:5]	r	Reserved Read as 0; should be written with 0.

Channel Trigger Value Register CTVx

CTV contains the value that determines the period of periodic triggers for each channel. It contains as well the counter that can be initialized with an offset so that the phase of harmonic trigger frequencies on different channels can be staggered.

CTVx (x = 0-7)

Channel Trigger Value Register x ($110_H + x \cdot 4$) **Reset Value: 0000 0000_H**



Peripheral Sensor Interface with Serial PHY Connection

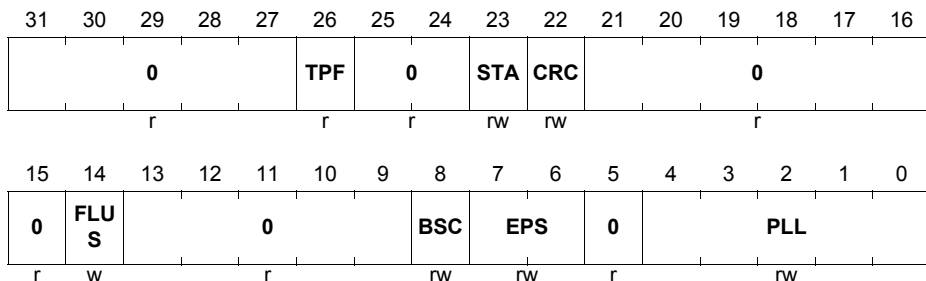
Field	Bits	Type	Description
CTV	[15:0]	rw	Channel Trigger Value CTV Contains the compare value (exact match) of Channel Trigger CTC at which a sync pulse is triggered for channel x and the counter CTC is cleared. If cleared, CTC is stopped and no pulse triggered.
CTC	[31:16]	rw	Channel Trigger Counter This bit field allows to read the current counter value of the reset timer cell CTVx. If GCR.ETCx is cleared, CTC can be written.

Send Control Registers SCR_x

The Send Control Register SCR_x contains control data bits required for sending data from the PSI5-S module to the sensor / external PHY.

Data is collected in SDR_x and the control bits are collected in SCR_x.

The send control unit is build to transmit complete ECU to sensor frames.

SCR_x (x = 0-7)
Send Control Register x (130_H+x*4) Reset Value: 0000 0000_H


Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
PLL	[4:0]	rw	Pay Load Length of Registers SDRx Defines the length that is taken into account: 00000 _B length is 6 00001 _B length is 6 00110 _B length is 6 11000 _B length is 24 11111 _B length is 24 PLL needs to be written before SDRx is used for proper operation.
EPS	[7:6]	rw	Enhanced Protocol Selection EPS[0] controls the Bit Format and MSB Fill bit. (0: Tooth Gap, MSB Fill 1) (1: PWM, MSB Fill 0) EPS[1] controls the Frame Format (0: Frame Format 1 - 3) (1: Frame Format 4) 00 _B Tooth Gap Method, '1' for filling the shift register from MSB; Frame format 1 -3 (3 bit start sequence, 3 bit stuffing distance, '1' for stuffing, 3 bit CRC). 01 _B Pulse Width Method, '0' for filling the shift register from MSB; Frame format 1 -3 (3 bit start sequence, 3 bit stuffing distance, '1' for stuffing, 3 bit CRC). 10 _B reserved, do not use! 11 _B Pulse Width Method, '0' for filling the shift register from MSB; Frame Format 4 (9 bit start sequence, 6 bit stuffing distance, '0' for stuffing, 6 bit CRC).
BSC	8	rw	Bit Stuffing Control 0 _B No automatic bit stuffing 1 _B Automatic bit stuffing is enabled. Depending from bit EPS[1] after 3 bits a '1' is inserted (Frame Format 1 -3) or after 6 bits a '0' is inserted (Frame Format 4)

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
FLUS	14	w	Flush SDRx Setting this bit stops shifting out the data in SDRx, the start sequence or CRC if any and clears the referring FSMs and counters, if EPS[0]=0 SDRx is flushed by setting all bits if EPS[0]=1 SDRx is flushed by clearing all bits clears TPF TPIx is issued at the end of successful flushing. Reads always as zero.
CRC	22	rw	CRC Generation Control 0 _B CRC is not generated automatically, it still can be written by SW together with the data (e.g. to test the remote CRC) 1 _B CRC is automatically generated by HW (according to EPS[1])
STA	23	rw	Start Sequence Generation Control 0 _B no start sequence generated, shifting out payload starts at bit 0! 1 _B automatically generated by HW (according to EPS[1]) shifting out payload starts after 3/9 bits (EPS[1] = 0/1)
TPF	26	r	Transmit in Progress Flag If set, data preparation and transmission is in progress: start sequence or CRC or stuffing bits or data from SDRx are being transferred. It is cleared automatically after preparation and transmitting is finished. If set, write access to SDRx will not change any data and issue TPOI.
0	[31:27],[25:24],[21:15],[13:9],5	r	Reserved Read as 0; should be written with 0.

Send Data Registers SDRx

Peripheral Sensor Interface with Serial PHY Connection

The Send Data Register SDR_x shows the data content of a data frame to be sent. CEN_x must be set for write access. For initialization SCR_x.FLUS can be used.

SDR_x (x = 0-7)

Send Data Register x (150_H+x*4) **Reset Value: 0000 0000_H**

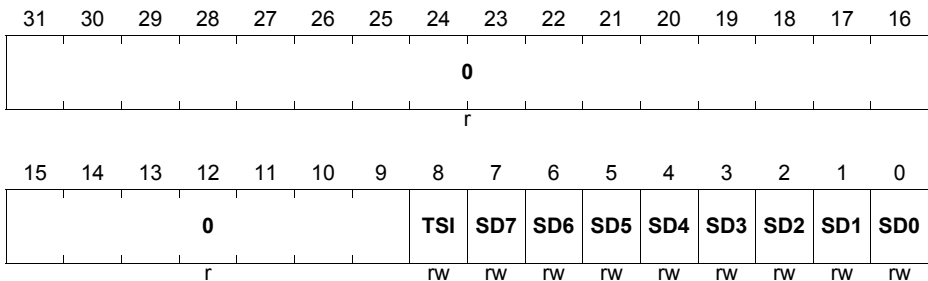
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								SD2 3	SD2 2	SD2 1	SD2 0	SD1 9	SD1 8	SD1 7	SD1 6
r								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SD1 5	SD1 4	SD1 3	SD1 2	SD1 1	SD1 0	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
SD_y (y = 0-23)	y	rw	SD_y Send data of next ECU to Sensor frame. Each time a bit is shifted out, the whole content is shifted right by 1 position and the MSB is filled with '1' or '0' depending from EPS[0]. This allows to read back SDR and determine the status of the shift process by SW. The unused MSBs (bit position SCR _x .PLL and higher) must be written with '0' if EPS[0] is set (PWM Method) and with '1' if EPS[0] is cleared (Tooth Gap Method). This is required, as the respective standard value is shifted into the register from MSB during shift out operation. SDR _x will be filled with this value after shift out process.
0	[31:24]	r	Reserved Read as 0.

CPU Direct Write Registers CDW

Peripheral Sensor Interface with Serial PHY Connection

The CPU Direct Write Register CDW allows the CPU to insert a command into the ASC FIFO. Note that Bits SD[7:5] select the channel for which the time stamp is captured. These bits are copied to the UART Frame Bit positions [2:0] (ChID) while SD[4:0] are copied to UART Frame Bit positions [7:3] (Command). See [Figure 34-6](#).

CDW
CPU Direct Write Register (170_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
SDy (y = 0-7)	y	rw	SDy Send data of next ECU to Sensor frame.
TSI	8	rw	Trigger Pulse Indicator If this bit is set, a sync pulse is assumed and thus a time stamp is captured when the command leaves the FIFO and is written to ASC TX Register. Bits SD[7:5] select the channel for which the time stamp is captured. RCRAx.TSTS must be cleared. If set, the time stamp is captured on Packet Frame reception only.
0	[31:9]	r	Reserved Read as 0.

34.17.6 ASC Registers

Peripheral Sensor Interface with Serial PHY Connection

The serial operating modes of the ASC module are controlled by its Control Register CON. This register contains control bits for mode and error check selection, and status flags for error identification.

CON
Control Register
(210_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			ODD TX	0								MTX			
r			rw	r								rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LB	BRS	ODD	FDE	OE	FE	PE	OEN	FEN	PEN	REN	STP	M		
rw	rw	rw	rw	rw	r	r	r	rw	rw	rw	r	rw	rw		

Field	Bits	Type	Description
M	[2:0]	rw	Mode Selection 000 _B 8-bit data Synchronous Mode. MTX needs to be cleared as well for proper operation. 001 _B 8-bit data Asynchronous Mode 010 _B Reserved. Do not use this combination. 011 _B 7-bit data + parity Asynchronous Mode 100 _B 9-bit data Asynchronous Mode 101 _B 8-bit data + wake up bit Asynchronous Mode 110 _B Reserved. Do not use this combination. 111 _B 8-bit data + parity Asynchronous Mode
STP	3	rw	Number of Stop Bit Selection 0 _B One stop bit 1 _B Two stop bits
REN	4	r	Receiver Enable Control 0 _B Receiver disabled 1 _B Receiver enabled Bit is reset by hardware after reception of a Byte in Synchronous Mode.
PEN	5	rw	Parity Check Enable (asynchronous mode only) 0 _B Ignore parity 1 _B Check parity

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
FEN	6	rw	Framing Check Enable (asynchronous mode only) 0 _B Ignore framing errors 1 _B Check framing errors
OEN	7	rw	Overrun Check Enable 0 _B Ignore overrun errors 1 _B Check overrun errors
PE	8	r	ASC Parity Error Flag Set by hardware on a parity error (PEN = 1). Must be reset by software.
FE	9	r	ASC Framing Error Flag Set by hardware on a framing error (FEN = 1). Must be reset by software.
OE	10	r	ASC Overrun Error Flag Set by hardware on an overrun error (OEN = 1). Must be reset by software.
FDE	11	rw	Fractional Divider Enable 0 _B Fractional divider disabled 1 _B Fractional divider is enabled and used as prescaler for baud rate timer (bit BRS is don't care) FDE is don't care and assumed '0' in Synchr. Mode.
ODD	12	rw	Parity Selection 0 _B Even parity selected (parity bit = 1 on odd number of 1s in data, parity bit = 0 on even number of 1s in data) 1 _B Odd parity selected (parity bit = 1 on even number of 1s in data, parity bit = 0 on odd number of 1s in data)
BRS	13	rw	Baud Rate Selection 0 _B Baud rate timer prescaler divide-by-2 selected 1 _B Baud rate timer prescaler divide-by-3 selected BRS is don't care if FDE = 1 (fractional divider enabled) FDE is don't care and assumed '0' in Synchr. Mode.
LB	14	rw	Loop-back Mode Enable 0 _B Loop-Back mode disabled 1 _B Loop-Back mode enabled

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
R	15	rw	Baud Rate Generator Run Control 0 _B Baud rate generator disabled (ASC inactive) 1 _B Baud rate generator enabled Register BG should only be written if R = 0.
MTX	[18:16]	rw	Mode Selection TX direction While bit field M controls the RX path, MTX controls the mode for the TX path. 000 _B 8-bit data Synchronous Mode. M needs to be cleared as well for proper operation. 001 _B 8-bit data Asynchronous Mode 010 _B Reserved. Do not use this combination. 011 _B 7-bit data + parity Asynchronous Mode 100 _B 9-bit data Asynchronous Mode 101 _B 8-bit data + wake up bit Asynchronous Mode 110 _B Reserved. Do not use this combination. 111 _B 8-bit data + parity Asynchronous Mode
ODDTX	28	rw	Parity Selection TX direction While bit field ODD controls the RX path, ODDTX controls the mode for the TX path. 0 _B Even parity selected (parity bit = 1 on odd number of 1s in data, parity bit = 0 on even number of 1s in data) 1 _B Odd parity selected (parity bit = 1 on even number of 1s in data, parity bit = 0 on odd number of 1s in data)
0	[31:29] , [27:19]	r	Reserved Read as 0; should be written with 0.

Serial data transmission or reception is possible only when the run bit CON.R is set to 1. Otherwise, the serial interface is idle. To avoid unpredictable behavior of the serial interface, do not program the mode control field CON.M to one of the reserved combinations.

Write Hardware Control Bits

The three error flags in register CON and the REN bit can be set or cleared by software via register WHBCON. WHBCON is a write-only register. Reading WHBCON always returns 0000 0000_H.

Peripheral Sensor Interface with Serial PHY Connection

WHBCON
Write Hardware Bits Control Register (250_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	SET OE	SET FE	SET PE	CLR OE	CLR FE	CLR PE	0	SET REN	CLR REN				0		
r	w	w	w	w	w	w	r	w	w				r		

Field	Bits	Type	Description
CLRREN	4	w	Clear Receiver Enable Bit 0 _B No effect 1 _B Bit CON.REN is cleared. Bit is always read as 0.
SETREN	5	w	Set Receiver Enable Bit 0 _B No effect 1 _B Bit CON.REN is set. Bit is always read as 0.
CLRPE	8	w	Clear Parity Error Flag 0 _B No effect 1 _B Bit CON.PE is cleared. Bit is always read as 0.
CLRFE	9	w	Clear Framing Error Flag 0 _B No effect 1 _B Bit CON.FE is cleared. Bit is always read as 0.
CLROE	10	w	Clear Overrun Error Flag 0 _B No effect 1 _B Bit CON.OE is cleared. Bit is always read as 0.
SETPE	11	w	Set Parity Error Flag 0 _B No effect 1 _B Bit CON.PE is set. Bit is always read as 0.

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
SETFE	12	w	Set Framing Error Flag 0 _B No effect 1 _B Bit CON.FE is set. Bit is always read as 0.
SETOE	13	w	Set Overrun Error Flag 0 _B No effect 1 _B Bit CON.OE is set. Bit is always read as 0.
0	[3:0], [7:6], [31:14]	r	Reserved Read as 0; should be written with 0.

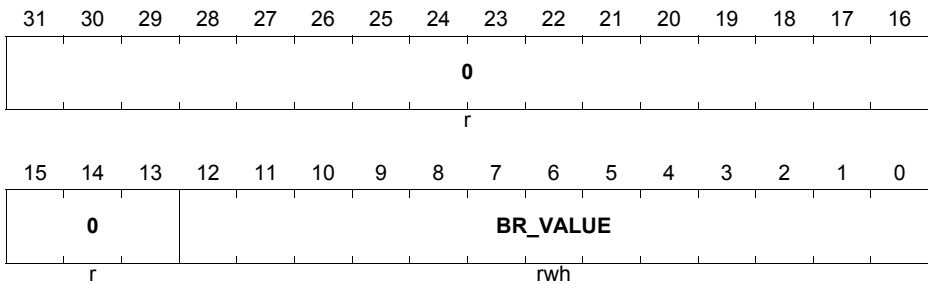
Note: When the set and clear bits for an error flag are set at the same time during a WHBCON write operation (e.g SETPE = CLRPE = 1), the error flag in CON is not affected.

Peripheral Sensor Interface with Serial PHY Connection

The Baud Rate Timer Reload Register BG of the ASC module contains the 13-bit reload value for the baud rate timer in Asynchronous and Synchronous Modes.

BG

Baud Rate Timer/Reload Register (214_H) **Reset Value: 0000 0000_H**

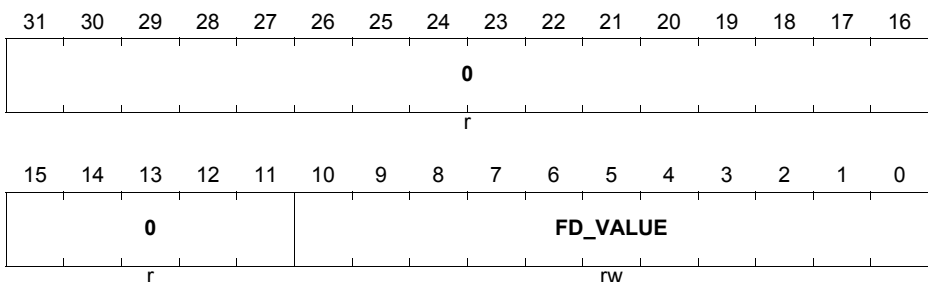


Field	Bits	Type	Description
BR_VALUE	[12:0]	rwh	Baud Rate Timer/Reload Register Value Reading BR_VALUE returns the 13-bit content of the baud rate timer. Writing BR_VALUE loads the baud rate timer reload register. BG should only be written if CON.R = 0.
0	[31:13]	r	Reserved Read as 0; should be written with 0.

The Fractional Divider Register FDV of the ASC module contains the 11-bit divider value for the fractional divider (asynchronous mode only).

FDV

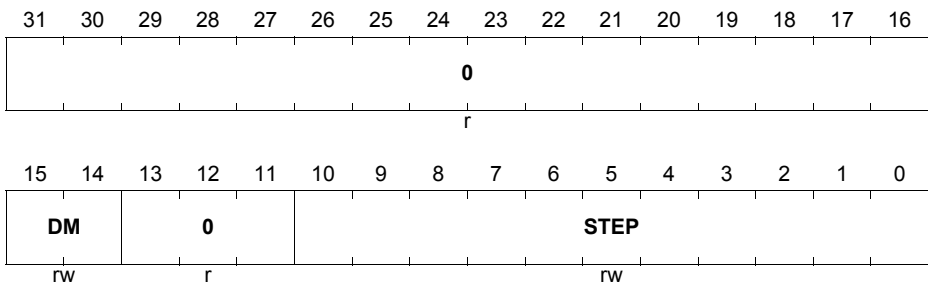
Fractional Divider Register (218_H) **Reset Value: 0000 0000_H**



Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
FD_VALUE	[10:0]	rw	Fractional Divider Register Value FD_VALUE contains the 11-bit value n of the fractional divider which determines the fractional divider ratio $n/2048$ ($n = 0-2047$). With $n = 0$, the fractional divider is switched off (divider ratio = 1).
0	[31:1 1]	r	Reserved Read as 0; should be written with 0.

The Fractional Divider Register FDO contains the 11-bit divider value for the fractional divider that is generating f_{PSISCLK} from f_{BAUD2} . PSISCLK is a clock that can be used on a pin to drive the external PHY.

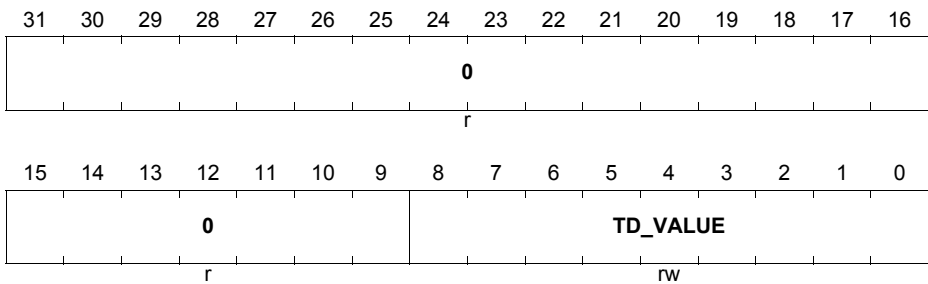
FDO
Fractional Divider for Output CLK Register (21C_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
STEP	[10:0]	rw	Step Value Reload or addition value for internal accumulator.
DM	[15:14]	rw	Divider Mode DM selects normal or fractional divider mode. 00 _B Fractional divider is switched off; no output clock is generated. The Reset External Divider signal is 1. (default after System Reset). 01 _B Normal Divider Mode selected. 10 _B Fractional Divider Mode selected. 11 _B Fractional divider is switched off; no output clock is generated.

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
0	[31:16], [13:11]	r	Reserved Read as 0; should be written with 0.

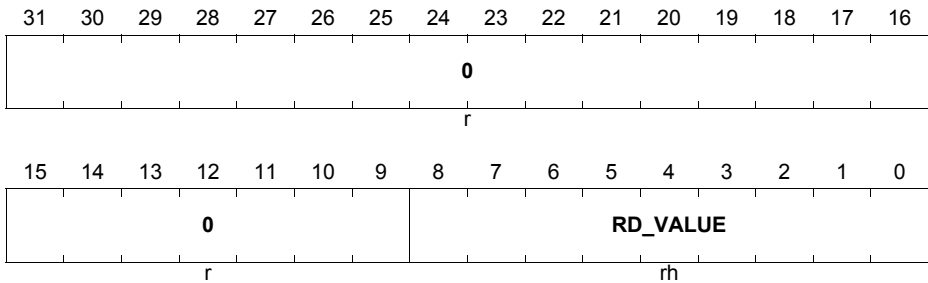
The Transmit Buffer Register TBUF of the ASC module contains the transmit data value in Asynchronous And Synchronous Modes. If GCR.ASC is cleared, TBUF is no longer writable by SW and interrupts are handled by the message reassembly block automatically.

TBUF
Transmit Buffer Register (220_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
TD_VALUE	[8:0]	rw	Transmit Data Register Value TBUF contains the data to be transmitted in the asynchronous and synchronous operating modes of the ASC. Data transmission is double-buffered; therefore, a new value can be written to TBUF before the transmission of the previous value is complete.
0	[31:9]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection

The receive buffer register RBUF of the ASC module contains the receive data value in Asynchronous and Synchronous Modes.

RBUF
Receive Buffer Register
(224_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RD_VALUE	[8:0]	rh	Receive Data Register Value RBUF contains the received data bits and, depending on the selected mode, the parity bit in the asynchronous and synchronous operating modes of the ASC. In Asynchronous Mode, with CON.M = 011 _B (7-bit data + parity), the received parity bit is written into RBUF.7. In Asynchronous Mode, with CON.M = 111 _B (8-bit data + parity), the received parity bit is written into RBUF.8.
0	[31:9]	r	Reserved Read as 0.

Peripheral Sensor Interface with Serial PHY Connection
34.17.7 Interrupt Control Registers
INTOV
Interrupt Overview Register
(300_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	FOI	XCR CI	TBIR	EIR	RIR	TIR	HDI	TPOI	TPI	CRCI	CHCI	TEI	RBI	RDI	RSI
r	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh	rh

Field	Bits	Type	Description
RSI	0	rh	Interrupt Pending on Node Pointer RSI If any interrupt requested flag is set for this Node Pointer in register (INTSTATx or INTSTATG) AND the referring interrupt is enabled in (INTENx or INTENG) then this bit is set. It is automatically reset if all flags in INTSTATx/G are cleared for which the referring interrupt is enabled in INTENx/G.
RDI	1	rh	Interrupt Pending on Node Pointer RDI See details of INTOV.RSI.
RBI	2	rh	Interrupt Pending on Node Pointer RBI See details of INTOV.RSI.
TEI	3	rh	Interrupt Pending on Node Pointer TEI See details of INTOV.RSI.
CHCI	4	rh	Interrupt Pending on Node Pointer CHCI See details of INTOV.RSI.
CRCI	5	rh	Interrupt Pending on Node Pointer CRCI See details of INTOV.RSI.
TPI	6	rh	Interrupt Pending on Node Pointer TPI See details of INTOV.RSI.
TPOI	7	rh	Interrupt Pending on Node Pointer TPOI See details of INTOV.RSI.

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
HDI	8	rh	Interrupt Pending on Node Pointer HDI See details of INTOV.HDI.
TIR	9	rh	Interrupt Pending on Node Pointer TIR If any interrupt requested flag is set for this Node Pointer in register INTSTATG AND the referring interrupt is enabled in INTENG then this bit is set. It is automatically reset if all flags in INTSTATG are cleared for which the referring interrupt is enabled in INTENG.
RIR	10	rh	Interrupt Pending on Node Pointer RIR See details of INTOV.TIR.
EIR	11	rh	Interrupt Pending on Node Pointer EIR See details of INTOV.TIR.
TBIR	12	rh	Interrupt Pending on Node Pointer TBIR See details of INTOV.TIR.
XCRCI	13	rh	Interrupt Pending on Node Pointer XCRCI See details of INTOV.TIR.
FOI	14	rh	Interrupt Pending on Node Pointer FOI See details of INTOV.TIR.
0	[31:15]	r	Reserved Read as 0.

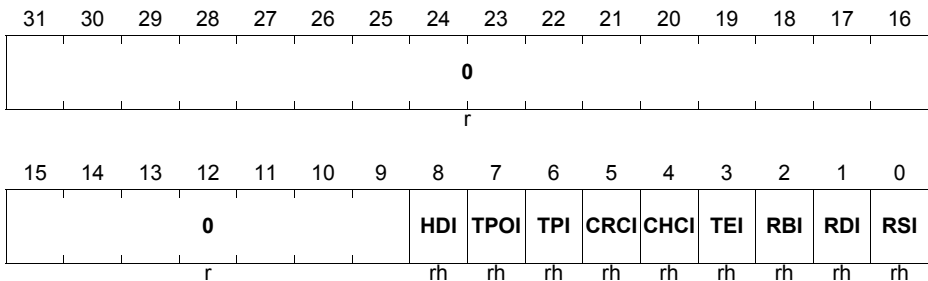
Peripheral Sensor Interface with Serial PHY Connection
Interrupt Status Registers x

The Interrupt Status Registers INTSTATx contains status bits that show the status of any interrupt of PSI5-S channel x.

The bits are set independently from the referring Interrupt Enable in Register INTENx. Thus they can be used as status bits as well e.g. by a SW based on polling.

INTSTATx (x = 0-7)

Interrupt Status Register x **(260_H+x*4)** **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RSI	0	rh	<p>Receive Success Interrupt Request Flag</p> <p>This bit is set at the successfully received end of a frame. It indicates that this frame is free of the errors CRCI, XCRCI, TEI, PE, FE, OE, RBI, HDI if selected to be taken into account in register GCR.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.RSI. This bit can be set by bit INTSET_x.RSI. This bit is set independently from INTEN_x.</p>

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
RDI	1	rh	<p>Receive Data Interrupt Request Flag</p> <p>RDI is activated when a received frame is moved to a Receive Data Register RDR. Both RDI and RSI will be issued together at correct reception.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.RDI. This bit can be set by bit INTSET_x.RDI. This bit is set independently from INTEN_x.</p>
RBI	2	rh	<p>Receive Buffer Overflow Interrupt Request Flag</p> <p>This bit is set after a frame has been received while the old one was not read from RDR. I.e. the kernel wants to set interrupt RDI and finds this interrupt already set.</p> <p>The old data is overwritten by the new data.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit is NOT cleared by reading RDR. This bit can be cleared by bit INTCLR_x.RBI. This bit can be set by bit INTSET_x.RBI. This bit is set independently from INTEN_x.</p>

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
TEI	3	rh	<p>Timing Error Interrupt Request Flag</p> <p>This bit is set if the watch dog timer expired. Depending from RCRAx.WDMS either the distance between two RDIs is longer than specified in WDL or it expired without reception of CHCI in time. I.e. the time from issuing the sync pulse to reception of the last expected frame configured in NFC.NFx was too long.</p> <p>Note that the root cause might be a non recoverable frame!</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.TEI. This bit can be set by bit INTSET_x.TEI. This bit is set independently from INTEN_x.</p>
CHCI	4	rh	<p>Channel Completed Interrupt Request Flag</p> <p>This bit is set if FCNT.FC_x equals NFC.NFx.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.CHCI. This bit can be set by bit INTSET_x.CHCI. This bit is set independently from INTEN_x.</p>
CRCI	5	rh	<p>CRC Error Request Flag</p> <p>This bit is set if the CRC fails.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.CRCI. This bit can be set by bit INTSET_x.CRCI. This bit is set independently from INTEN_x.</p>

Peripheral Sensor Interface with Serial PHY Connection

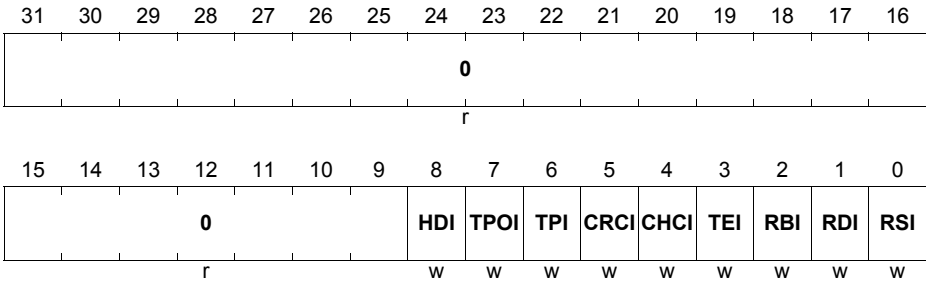
Field	Bits	Type	Description
TPI	6	rh	<p>Transfer Preparation Interrupt Request Flag</p> <p>This bit is set after data to be transferred has been moved completely. Thus a new value can be written to SDRx.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.TPI. This bit can be set by bit INTSET_x.TPI. This bit is set independently from INTEN_x.</p>
TPOI	7	rh	<p>Transmit Preparation Overflow Interrupt Request Flag</p> <p>This bit is set if SDR is written while TPF is set. The old data is NOT overwritten.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLR_x.TPOI. This bit can be set by bit INTSET_x.TPOI. This bit is set independently from INTEN_x.</p>
HDI	8	rh	<p>Header Error Signalled Flag</p> <p>This bit is set if at least one of the error signalling flags in the enveloping Packet Frame Err0 and Err1 is set. Up</p> <p>0_B (Err0 OR Err1) = false (0)</p> <p>1_B (Err0 OR Err1) = true (1)</p>
0	[31:9]	r	<p>Reserved</p> <p>Read as 0.</p>

Peripheral Sensor Interface with Serial PHY Connection
Interrupt Set Registers x

The Interrupt Set Registers INTSETx contain control bits that trigger an interrupt pulse for any interrupt of PSI5-S channel x.

INTSETx (x = 0-7)

Interrupt Set Register x (280_H+x*4) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RSI	0	w	Set Interrupt Request Flag RSI Setting this bit set bit INTSTATx.RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	Set Interrupt Request Flag RDI Setting this bit set bit INTSTATx.RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	Set Interrupt Request Flag RBI Setting this bit set bit INTSTATx.RBI. Clearing this bit has no effect. Reading this bit returns always zero.
TEI	3	w	Set Interrupt Request Flag TEI Setting this bit set bit INTSTATx.TEI. Clearing this bit has no effect. Reading this bit returns always zero.
CHCI	4	w	Set Interrupt Request Flag CHCI Setting this bit set bit INTSTATx.CHCI. Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface with Serial PHY Connection

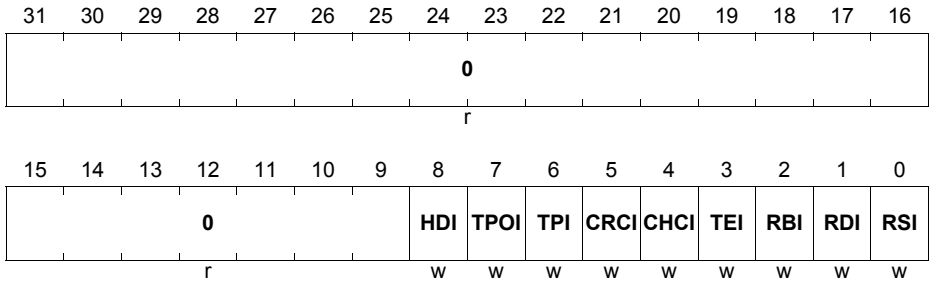
Field	Bits	Type	Description
CRCI	5	w	Set Interrupt Request Flag CRCI Setting this bit set bit INTSTATx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.
TPI	6	w	Set Interrupt Request Flag TPI Setting this bit set bit INTSTATx.TPI. Clearing this bit has no effect. Reading this bit returns always zero.
TPOI	7	w	Set Interrupt Request Flag TPOI Setting this bit set bit INTSTATx.TPOI. Clearing this bit has no effect. Reading this bit returns always zero.
HDI	8	w	Set Interrupt Request Flag HDI Setting this bit set bit INTSTATx.HDI. Clearing this bit has no effect. Reading this bit returns always zero.
0	[31:9]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection
Interrupt Clear Register x

The Interrupt Clear Register INTCLR_x contain control bits that clear the status of any interrupt of PSI5-S channel x.

INTCLR_x (x = 0-7)

Interrupt Clear Register x **(2A0_H+x*4)** **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RSI	0	w	Clear Interrupt Request Flag RSI Setting this bit clears bit INTSTAT _x .RSI. Clearing this bit has no effect. Reading this bit returns always zero.
RDI	1	w	Clear Interrupt Request Flag RDI Setting this bit clears bit INTSTAT _x .RDI. Clearing this bit has no effect. Reading this bit returns always zero.
RBI	2	w	Clear Interrupt Request Flag RBI Setting this bit clears bit INTSTAT _x .RBI. Clearing this bit has no effect. Reading this bit returns always zero.
TEI	3	w	Clear Interrupt Request Flag TEI Setting this bit clears bit INTSTAT _x .TEI. Clearing this bit has no effect. Reading this bit returns always zero.
CHCI	4	w	Clear Interrupt Request Flag CHCI Setting this bit clears bit INTSTAT _x .CHCI. Clearing this bit has no effect. Reading this bit returns always zero.

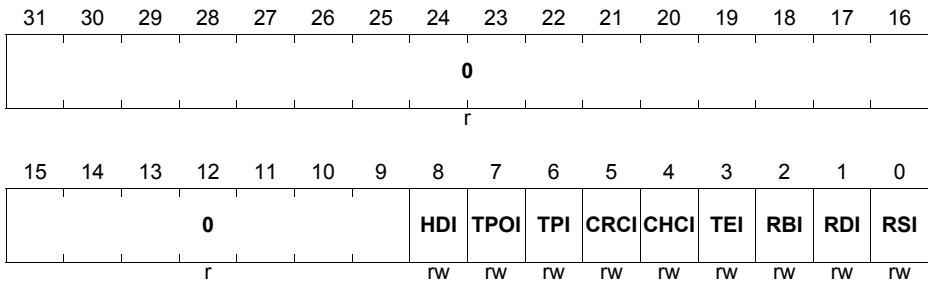
Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
CRCI	5	w	Clear Interrupt Request Flag CRCI Setting this bit clears bit INTSTATx.CRCI. Clearing this bit has no effect. Reading this bit returns always zero.
TPI	6	w	Clear Interrupt Request Flag TPI Setting this bit clears bit INTSTATx.TPI. Clearing this bit has no effect. Reading this bit returns always zero.
TPOI	7	w	Clear Interrupt Request Flag TPOI Setting this bit clears bit INTSTATx.TPOI. Clearing this bit has no effect. Reading this bit returns always zero.
HDI	8	w	Clear Interrupt Request Flag HDI Setting this bit clears bit INTSTATx.HDI. Clearing this bit has no effect. Reading this bit returns always zero.
0	[31:9]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection
Interrupt Enable Registers x

The Interrupt Enable Register INTEN x contain control bits that enable the interrupt source of any interrupt of PSI5-S channel x.

The Interrupt Status bits in register INTSTATx are set independently from the Interrupt Enable in Register INTENx.

INTENx (x = 0-7)
Interrupt Enable Register x ($2C0_H + x \cdot 4$) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
RSI	0	rw	Enable Interrupt Request RSI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
RDI	1	rw	Enable Interrupt Request RDI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
RBI	2	rw	Enable Interrupt Request RBI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
TEI	3	rw	Enable Interrupt Request TEI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
CHCI	4	rw	Enable Interrupt Request CHCI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
CRCI	5	rw	Enable Interrupt Request CRCI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
TPI	6	rw	Enable Interrupt Request TPI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
TPOI	7	rw	Enable Interrupt Request TPOI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
HDI	8	rw	Enable Interrupt Request HDI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
0	[31:9]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection
Interrupt Node Pointer Registers x

The Interrupt Node Pointer Register INPx contains the node pointers of PSI5-S channel x.

INPx (x = 0-7)
Interrupt Node Pointer Register x ($2E0_H + x \cdot 4$) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				HDI			TPOI			TPI			CRCI		
r				rw			rw			rw			rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCI		CHCI		TEI			RBI		RDI			RSI			
rw		rw		rw			rw		rw			rw			

Field	Bits	Type	Description
RSI	[2:0]	rw	Interrupt Node Pointer for Interrupt RSI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.RSI (if enabled by bit INTENx.RSI). 000 _B Trigger Output TRIGO 0 is selected 001 _B Trigger Output TRIGO 1 is selected 111 _B Trigger Output TRIGO 7 is selected
RDI	[5:3]	rw	Interrupt Node Pointer for Interrupt RDI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.RDI (if enabled by bit INTENx.RDI). For bit field definition, see RSI.
RBI	[8:6]	rw	Interrupt Node Pointer for Interrupt RBI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.RBI (if enabled by bit INTENx.RBI). For bit field definition, see RSI.

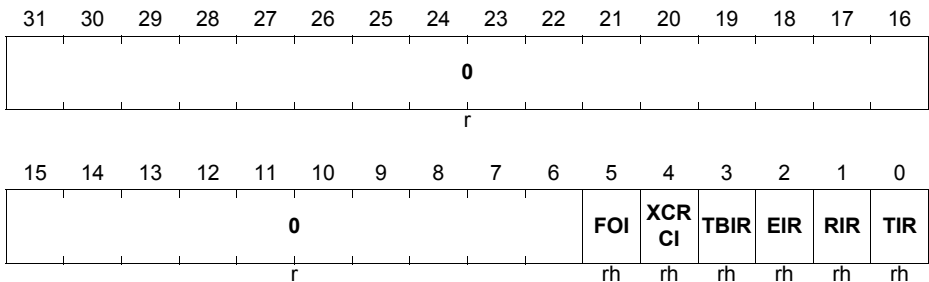
Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
TEI	[11:9]	rw	Interrupt Node Pointer for Interrupt TEI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.TEI (if enabled by bit INTENx.TEI). For bit field definition, see RSI.
CHCI	[14:12]	rw	Interrupt Node Pointer for Interrupt CHCI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.CHCI (if enabled by bit INTENx.CHCI). For bit field definition, see RSI.
CRCI	[17:15]	rw	Interrupt Node Pointer for Interrupt CRCI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.CRCI. For bit field definition, see RSI.
TPI	[20:18]	rw	Interrupt Node Pointer for Interrupt TOI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.TPI (if enabled by bit INTENx.TPI). For bit field definition, see RSI.
TPOI	[23:21]	rw	Interrupt Node Pointer for TPOI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.TPOI. For bit field definition, see RSI.
HDI	[26:24]	rw	Interrupt Node Pointer for HDI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATx.HDI. For bit field definition, see RSI.
0	[31:27]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection
Interrupt Status Registers Global

The Interrupt Status Registers INTSTATG contains status bits that show the status of any global interrupt, i.e. of the ASC inside PSI5-S and XCRCI indicating a non recoverable message is received. On XCRCI, the non recoverable message is stored in ChID '0', FID '1' with original IDs. INTSTATG contains as well FOI indicating a FIFO overrun condition. In a correct setup, this will never be set as the bandwidth of the ASC is assumed to be by far higher than the write bandwidth in order to have short delays.

The bits are set independently from the referring Interrupt Enable in Register INTENG. Thus they can be used as status bits as well e.g. by a SW based on polling.

INTSTATG
Interrupt Status Register G (304_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
TIR	0	rh	Transmit Interrupt Request Flag 0 _B No interrupt was requested since this bit was cleared the last time 1 _B An interrupt was requested since this bit was cleared the last time This bit can be cleared by bit INTCLRG.TIR. This bit can be set by bit INTSETG.TIR. This bit is set independently from INTENG.
RIR	1	rh	Receive Interrupt Request Flag 0 _B No interrupt was requested since this bit was cleared the last time 1 _B An interrupt was requested since this bit was cleared the last time This bit can be cleared by bit INTCLRG.RIR. This bit can be set by bit INTSETG.RIR. This bit is set independently from INTENG.

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
EIR	2	rh	<p>Error Interrupt Request Flag</p> <p>The cause of an error interrupt request EIR (framing, parity, overrun error) can be identified by the error status flags CON.FE, CON.PE, and CON.OE,</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRG.EIR. This bit can be set by bit INTSETG.EIR. This bit is set independently from INTENG.</p>
TBIR	3	rh	<p>Transmit Buffer Interrupt Request Flag</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRG.TBIR. This bit can be set by bit INTSETG.TBIR. This bit is set independently from INTENG.</p>
XCRCI	4	rh	<p>XCRC Error Request Flag</p> <p>This bit is set if the CRC check on the enveloping Packet Frame fails including the case where XCRC check can not be performed (non recoverable frames) see Chapter 34.5.1.1. The received data is not reliable and stored in ChID '0', FID '1' with original IDs.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRG.XCRCI. This bit can be set by bit INTSETG.XCRCI. This bit is set independently from INTENG.</p>

Peripheral Sensor Interface with Serial PHY Connection

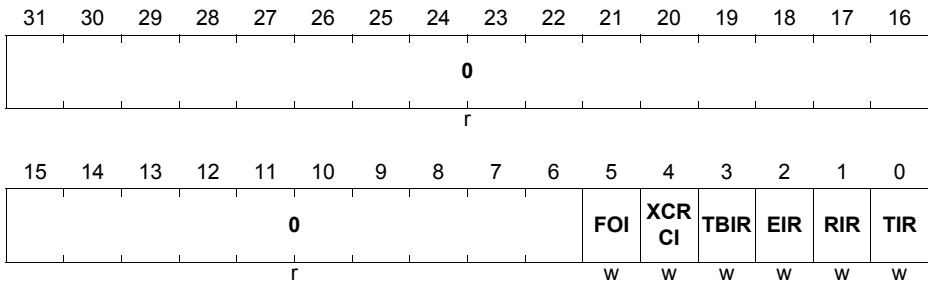
Field	Bits	Type	Description
FOI	5	rh	<p>FIFO Error Request Flag</p> <p>This bit is set if the Transmit FIFO of the message generation is overrun i.e. a transfer to the FIFO was generated by the message generation unit or by CDW (CPU direct write register) while the FIFO was already full.</p> <p>0_B No interrupt was requested since this bit was cleared the last time</p> <p>1_B An interrupt was requested since this bit was cleared the last time</p> <p>This bit can be cleared by bit INTCLRG.FOI. This bit can be set by bit INTSETG.FOI. This bit is set independently from INTENG.</p>
0	[31:6]	r	<p>Reserved</p> <p>Read as 0.</p>

Peripheral Sensor Interface with Serial PHY Connection
Interrupt Set Registers Global

The Interrupt Set Register INTSETG contains control bits that trigger an interrupt pulse for any interrupt of the ASC integrated in PSI5-S and for XCRCI.

INTSETG

Interrupt Set Register G (308_H) Reset Value: 0000 0000_H



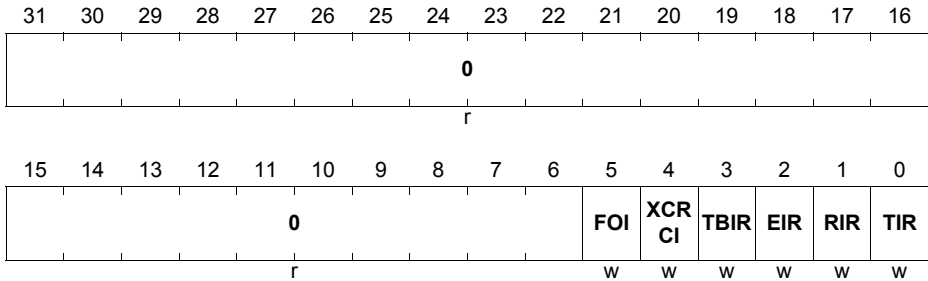
Field	Bits	Type	Description
TIR	0	w	Set Interrupt Request Flag TIR Setting this bit set bit INTSTATG.TIR. Clearing this bit has no effect. Reading this bit returns always zero.
RIR	1	w	Set Interrupt Request Flag RIR Setting this bit set bit INTSTATG.RIR. Clearing this bit has no effect. Reading this bit returns always zero.
EIR	2	w	Set Interrupt Request Flag EIR Setting this bit set bit INTSTATG.EIR. Clearing this bit has no effect. Reading this bit returns always zero.
TBIR	3	w	Set Interrupt Request Flag TBIR Setting this bit set bit INTSTATG.TBIR. Clearing this bit has no effect. Reading this bit returns always zero.
XCRCI	4	w	Set Interrupt Request Flag XCRCI Setting this bit set bit INTSTATG.XCRCI. Clearing this bit has no effect. Reading this bit returns always zero.

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
FOI	5	w	Set Interrupt Request Flag FOI Setting this bit set bit INTSTATG.FOI. Clearing this bit has no effect. Reading this bit returns always zero.
0	[31:6]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection
Interrupt Clear Register Global

The Interrupt Clear Register INTCLRG contain control bits that clear the status of any interrupt of the ASC integrated in PSI5-S and XCRCI.

INTCLRG
Interrupt Clear Register G
(30C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
TIR	0	w	Clear Interrupt Request Flag TIR Setting this bit clears bit INTSTATG.TIR. Clearing this bit has no effect. Reading this bit returns always zero.
RIR	1	w	Clear Interrupt Request Flag RIR Setting this bit clears bit INTSTATG.RIR. Clearing this bit has no effect. Reading this bit returns always zero.
EIR	2	w	Clear Interrupt Request Flag EIR Setting this bit clears bit INTSTATG.EIR. Clearing this bit has no effect. Reading this bit returns always zero.
TBIR	3	w	Clear Interrupt Request Flag TBIR Setting this bit clears bit INTSTATG.TBIR. Clearing this bit has no effect. Reading this bit returns always zero.
XCRCI	4	w	Clear Interrupt Request Flag XCRCI Setting this bit clears bit INTSTATG.XCRCI. Clearing this bit has no effect. Reading this bit returns always zero.

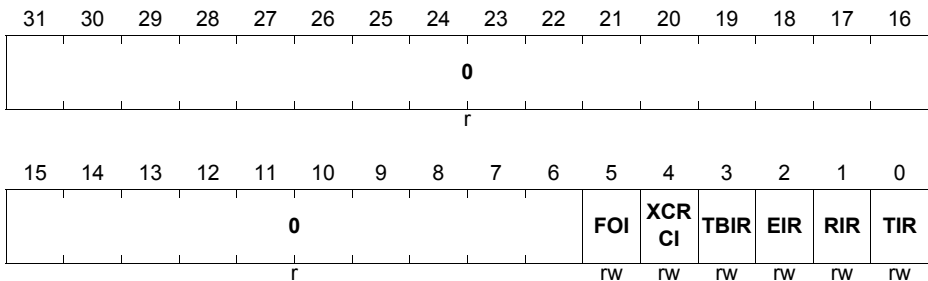
Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
FOI	5	w	Clear Interrupt Request Flag FOI Setting this bit clears bit INTSTATG.FOI. Clearing this bit has no effect. Reading this bit returns always zero.
0	[31:6]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection
Interrupt Enable Registers Global

The Interrupt Enable Register INTENG contain control bits that enable the interrupt source of any interrupt of the ASC integrated in PSI5-S and XCRCI.

The Interrupt Status bits in register INTSTATG are set independently from the Interrupt Enable in Register INTENG.

INTENG
Interrupt Enable Register G
(310_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
TIR	0	rw	Enable Interrupt Request TIR 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
RIR	1	rw	Enable Interrupt Request RIR 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
EIR	2	rw	Enable Interrupt Request EIR 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source

Peripheral Sensor Interface with Serial PHY Connection

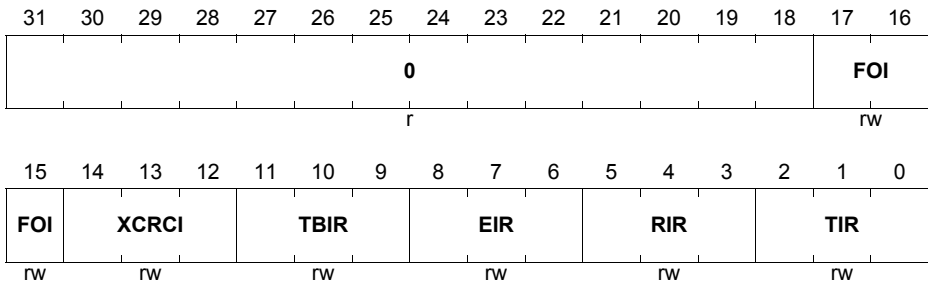
Field	Bits	Type	Description
TBIR	3	rw	Enable Interrupt Request TBIR 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
XCRCI	4	rw	Enable Interrupt Request XCRCI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
FOI	5	rw	Enable Interrupt Request FOI 0 _B No interrupt request can be generated for this source 1 _B An interrupt request can be generated for this source
0	[31:6]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection
Interrupt Node Pointer Register Global

The Interrupt Node Pointer Register INPG contains the node pointers of PSI5-S ASC and XCRCI.

INPG

Interrupt Node Pointer G Register (314_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
TIR	[2:0]	rw	Interrupt Node Pointer for Interrupt TIR This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATG.TIR (if enabled by bit INTENG.TIR). 000 _B Trigger Output TRIGO 0 is selected 001 _B Trigger Output TRIGO 1 is selected 111 _B Trigger Output TRIGO 7 is selected
RIR	[5:3]	rw	Interrupt Node Pointer for Interrupt RIR This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATG.RIR (if enabled by bit INTENG.RIR). For bit field definition, see TIR.
EIR	[8:6]	rw	Interrupt Node Pointer for Interrupt EIR This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATG.EIR (if enabled by bit INTENG.EIR). For bit field definition, see TIR.

Peripheral Sensor Interface with Serial PHY Connection

Field	Bits	Type	Description
TBIR	[11:9]	rw	Interrupt Node Pointer for Interrupt TBIR This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATG.TBIR (if enabled by bit INTENG.TBIR). For bit field definition, see TIR.
XCRCI	[14:12]	rw	Interrupt Node Pointer for Interrupt XCRCI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATG.XCRCI (if enabled by bit INTENG.XCRCI). For bit field definition, see TIR.
FOI	[17:15]	rw	Interrupt Node Pointer for Interrupt FOI This bit field defines the interrupt node, that is requested due to the set condition for bit INTSTATG.FOI (if enabled by bit INTENG.FOI). For bit field definition, see TIR.
0	[31:18]	r	Reserved Read as 0; should be written with 0.

Peripheral Sensor Interface with Serial PHY Connection

34.18 PSI5-S Module Implementation

This section describes the PSI5-S module interface as it is implemented in the TC27x. It especially covers clock control, port and on-chip connections, interrupt control, and address decoding.

34.18.1 Interface Connections of the PSI5-S Module

Figure 34-35 shows the TC26x B-Step-specific implementation details and interconnections of the PSI5-S module.

The PSI5-S module is supplied with a separate clock control, address decoding, and interrupt control logic. The 8 modules' service request outputs are connected with the interrupt router. 6 outputs of the GTM module are connected to the timer inputs.

The serial data inputs PSISRxn of the receive channels of the PSI5-S module as well as the sender output PSISTX are connected to GPIO lines. If the output is used, it is mapped to a different port pin than the referring PSI5-S data input line.

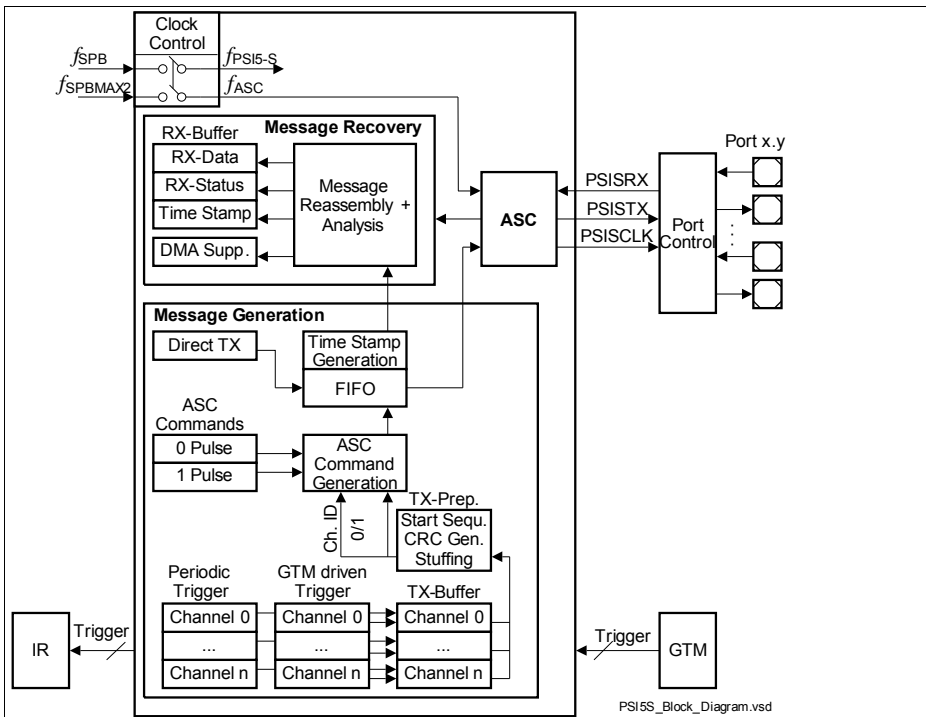


Figure 34-35 PSI5-S Module Implementation and Interconnections

Peripheral Sensor Interface with Serial PHY Connection
34.18.1.1 On-Chip Connections

This section describes the on-chip connections of the PSI5-S module.

Interrupt Router Service Requests

The module has 8 Request Trigger outputs connecting it to the interrupt router (IR). The request lines are connected to the IR controller as shown in [Table 34-7](#). Interrupt node pointers (INPx) select the line for each trigger source. Multiple nodes may point to the same line. In this case the triggers are combined with a simple OR logic.

Table 34-7 Service Request Lines of PSI5-S

INP value	Request Line	Connected to	Description
000b	TRIGO0	PSI5S_SRC0	PSI5-S Service Request Node 0
001b	TRIGO1	PSI5S_SRC1	PSI5-S Service Request Node 1
010b	TRIGO2	PSI5S_SRC2	PSI5-S Service Request Node 2
011b	TRIGO3	PSI5S_SRC3	PSI5-S Service Request Node 3
100b	TRIGO4	PSI5S_SRC4	PSI5-S Service Request Node 4
101b	TRIGO5	PSI5S_SRC5	PSI5-S Service Request Node 5
110b	TRIGO6	PSI5S_SRC6	PSI5-S Service Request Node 6
111b	TRIGO7	PSI5S_SRC7	PSI5-S Service Request Node 7

Trigger Inputs

The module has 8 PSI5-S Channels and 6 trigger inputs which can be randomly chosen by programming PGCx.ETS/ETB, FDRT.ECS and TSCNTA/B.ETB. The trigger inputs (TRIG[7:0]) of the PSI5-S module are connected to the GTM as shown in [Table 34-8](#).

Table 34-8 Trigger Input Lines of PSI5-S

Request Line	Connected to	Description
TRIG0	gtm.psi5_0_trig_0_o	GTM Trigger Line TRIG0
TRIG1	gtm.psi5_1_trig_0_o	GTM Trigger Line TRIG1
TRIG2	gtm.psi5_2_trig_0_o	GTM Trigger Line TRIG2
TRIG3	GND	not connected to GTM
TRIG4	gtm.psi5_4_trig_0_o	GTM Trigger Line TRIG4
TRIG5	gtm.psi5_5_trig_0_o	GTM Trigger Line TRIG5
TRIG6	gtm.psi5_6_trig_0_o	GTM Trigger Line TRIG6
TRIG7	GND	not connected to GTM

Peripheral Sensor Interface with Serial PHY Connection

34.18.2 PSI5-S Module-Related External Registers

The registers listed in [Figure 34-36](#) are not included in the PSI5-S module kernel but must be programmed for proper operation of the PSI5-S module.

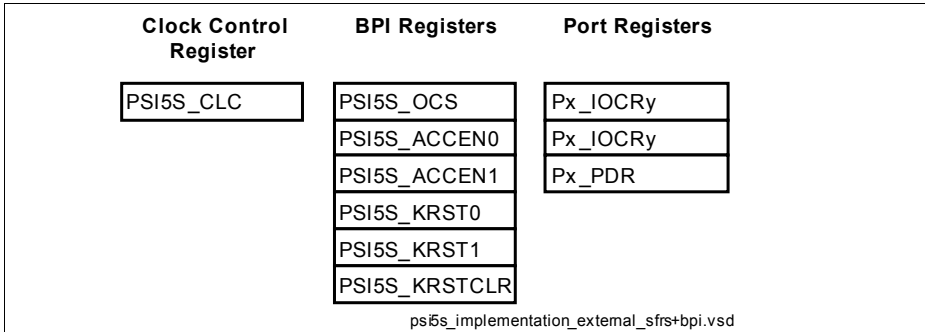


Figure 34-36 PSI5-S Implementation-specific Special Function Registers

34.18.2.1 Port Control

The Port Control Registers contain:

- Input/output function selection (Port IOCR registers)
- Pad driver characteristics selection for the outputs (Port PDR registers)

Input/Output Function Selection

[Table 34-9](#) shows an overview how bits and bit fields must be programmed for the required I/O functionality of the PSI5-S I/O lines.

The PSI5-S interface input shares its input pins with digital ports.

For one PSI5-S module the output and input pins are next to each other. Not connected inputs are tied internally on chip.

The Module can be used as ASC only. PSI5-S message reassembly unit, message generation and pulse generation are not available in this mode.

If in this use case the module is switched to SSC mode, the port wiring is switched automatically in the following way:

- RXD line remains receive data input,
- CLK provides the synchronous transmit clock coming from the internal TXD signal and
- TXD provides the output data coming from the internal RXD signal.

This way the wiring is compatible with the wiring of other UART modules of the product family. See [Figure 34-37](#).

Peripheral Sensor Interface with Serial PHY Connection

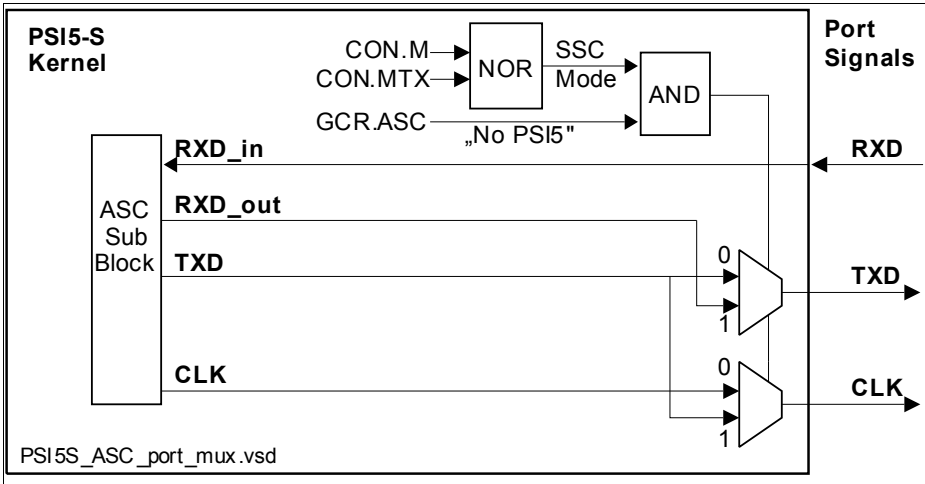


Figure 34-37 Port Multiplexer for ASC in SSC Mode (only if GCR.ASC is set)

Table 34-9 PSI5-S I/O Control Selection and Setup

Port Lines	Input Select Register	Input/Output Control Register Bits	I/O
P00.3/ PSISRXA	PSI5S_IOC.R.ALT1 = 00 _B	P00_IOC.R0.PC3 = 0XXXX _B	I
P00.4/ PSISTX	not applicable	P00_IOC.R4.PC4 = 1X010 _B	O
P02.5/ PSISRXB	PSI5S_IOC.R.ALT1 = 01 _B	P02_IOC.R4.PC5 = 0XXXX _B	I
P02.6/ PSISTX	not applicable	P02_IOC.R4.PC6 = 1X010 _B	O
P33.5/ PSISRXC	PSI5S_IOC.R.ALT1 = 10 _B	P33_IOC.R4.PC5 = 0XXXX _B	I
P33.6/ PSISTX	not applicable	P33_IOC.R4.PC6 = 1X111 _B	O
P02.4/ PSISCLK	not applicable	P02_IOC.R4.PC4 = 1X100 _B	O
P33.10/ PSISCLK	not applicable	P33_IOC.R8.PC10 = 1X101 _B	O

Peripheral Sensor Interface with Serial PHY Connection
34.19 Revision History

This Target Specification is based on: "PSI5 Specification **V2.0 2011-06**".

Table 34-10 Revision History

Version Number	Changes to Previous Version
Rev_0.1D	First Draft (Bullet Points ++)
Rev_0.2	Review Version Addresses, Implementation ASC are the main areas that need further work
Rev_0.3	First Complete Version Addresses, integration of ASC, interrupts completed and details on DMA support
Rev_0.4	Version with lead customer feed back Added separate ASC Parity control for RX and TX Added SW control register SWAP to change buffer base address Added automatic timing of buffer swapping in BCRA.BPRP Added 3 Byte Packet Frames Added bit field UART Frames count per Packet frame (RCRA.UFC) Added description of reassembly process Added idle signal from ASC to reassembly unit for Packet Frame separation. Changed location of PSI5 Frame CRC to RDS Bit EPS changed to 2 bits to allow for Frame 1 -3 in with PWM Method APL enlarged from 4 to 4 Bits Renaming TSRA/B to TSCNTA/B, TSCx to TSCR Consistent naming convention for UART Frames, Packet Frames, PSI5 Frames Removed CLC.RMC (pre divider)
Rev_0.5	Renamed SPL/APL to TXCMD/ATXCMD Rephrased Loop Back description wrt. parity settings Renamed CDW.SP to TSI "Time Stamp Indicator" Deleted CDWx, only one CDW for all channels (typo) Added bit TSTS selecting trigger for time stamp Sync Pulse/Packet Frame Added bit FIDS selecting FID from Packet Frame/revolving number Added Register NFC supporting WD in angle synchronous channels Changed Module base and end address

Peripheral Sensor Interface with Serial PHY Connection
Table 34-10 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.0	<p>Frame Buffer in System Memory: Removed double buffer concept and replaced it by memory mapped concept.</p> <p>Removed registers for SW swapping and data consistency (ABAR, SWAP, BSRA, BSRB)</p> <p>Removed BFI</p> <p>Added CHCI</p> <p>Changed NFC.NFx definitions and added RCRAx.WDMS controlling the Watch Dog Timer behavior.</p> <p>Swapped addresses of RDS and TSM. Having the TS written at the end of a DMA transfer allows a simple check: all data in a buffer is consistent if the TS are strictly monotonic increasing from FID1 .. FID6 - within senseful boundaries. This does not work for extreme use cases.</p> <p>Added CLC.RMC and CLC.EDIS</p> <p>Removed automatic clear of TPI by writing SDRx.</p> <p>Moved XCRC to INTSTATG as it is not available for each channel but for Ch 0 only.</p> <p>Reduced INP bit field to 3 bit</p> <p>Added INTSTATx.HDI and INPx.HDI</p> <p>Removed RCRAx.EN (local channel enable), function is in GCR</p> <p>Added RCRAx.TSEN (Time Stamp enable)</p> <p>Removed TSCNTA/B.ACLR, CLR, TBE</p> <p>Added TSCNTA. CLRA, CLRB, TBEA, TBEB for simultaneous start/stop and clear of both timers.</p> <p>Removed bit SCR.TRQ, redundant with TPF.</p>
Rev_1.1	<p>Updated OCS description.</p> <p>Changed FID from 1 .. 6 to 0 .. 5; non recoverable frames ChID 0, FID 1, transceiver messages ChID 0, FID 0, rolling number FID for asynchronous frames is 0 .. 5</p> <p>Change to TSP: for non recoverable messages, TSP is not relevant but TSCNTA is used always.</p> <p>Changed RDS.CRC to reverse order</p> <p>Changed RDS.HDI bit position to 13, ERR0/1 moved to 11/12</p> <p>Added PFC to RDR, RDS, TSM</p> <p>Detailed definition of XCRC</p> <p>In Tooth Gap Method all commands are transferred to the UART even if it is a command for '0' i.e. for "no pulse"</p> <p>Replaced CLC.RMC by new register FDR.</p> <p>FIFO overrun condition is signalled by INSTATG.FOI</p>

Peripheral Sensor Interface with Serial PHY Connection

Table 34-10 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.2	Added INTOV.FOI corrected typos
Rev_1.3	Added paragraph on wrap around of BAR to register description of BAR. RBI depends on RDI only (no dependency on RSI any more) due to conflicts with GCR.RBI (loop back timing loop) R/TBUF are readable while GCR.ASC is cleared Added clarification: RDS.TEI is set for all frames till CHCI
Rev_1.4	Typo corrected in description of BAR.
Rev_1.5	Added FDO providing PSISCLK, clock out for PHY. Updated Pinning Table
Rev_1.6	Removed FDO.RESULT Removed GTC.ETC typo. Clarified that non recoverable messages are always stored in ChID '0', FID '1' but with original IDs as received. Updated pinning Added clarification to RDS.FID: Frames with FID [6:7] are copied to ChID 0, FID 1 with original IDs Added hint: CON.M and CON.MTX need to be cleared both for sync mode Added mux on port wiring for "ASC only in SSC mode" use case
Rev_1.7	Typo corrections Clarification added in NFC.FCx: FC is cleared with Sync Pulse only in Sync Mode (WDMS is set) Clarification added in RDS.TEI: when TEI is cleared in INTSTATx, it is no longer shown in RDS. Added hint to CDW clarifying the relation between layout of CDW and actual frame (updated Figure 34-6 accordingly) Changed chapter “Packet Frames” received from PHY on Page 34-6 to allow reception of good frames w/o correct idle at the end
Rev_1.8	Correction of feature list entry on asynchronous channels Added RDS.AFC for debugging. Corrected CLC.EDIS description. Added section "Non Recoverable Frame Presentation in RDR and RDS" Moved PSISCLK from P02.2 O2 to P02.4 O4
Rev_1.9	Changed number of GTM inputs depending on product implementation
Rev_1.10	Updated details on DMA programming

35 Ethernet MAC (ETH)

This chapter describes the Ethernet MAC ^{1) 2)} of the TC26x B-Step.

35.1 Overview

Figure 35-1 shows the basic structure of the module:

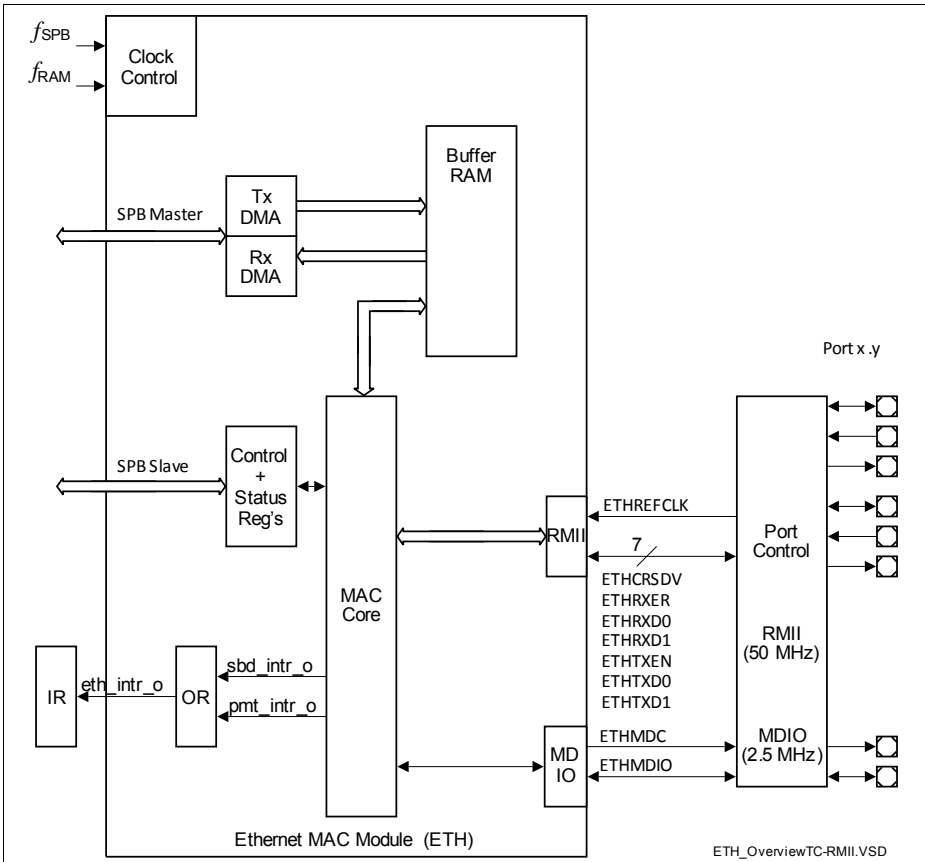


Figure 35-1 Ethernet MAC Module Overview

- 1) Excerpted portions are Synopsys Proprietary. Used with permission.
- 2) Module is not available in some variants. In these variants registers are still accessible but functionality cannot be guaranteed.

35.1.1 General Module Description

The DWC Ether MAC 10/100/1000 Universal, V3.7a, commonly referred to as GMAC-UNIV in this document, enables a host to transmit and receive data over Ethernet in compliance with the IEEE 802.3-2002 standard. In TC26x B-Step it is configured to support 10 and 100 Mbit modes. Note that Gigabit mode is not configured, nevertheless the core is referred to as GMAC to avoid confusion with other cores.

The GMAC-UNIV provides an optimized (with respect to gate count and latency), configurable, flexible product to meet the needs of various applications and customers, and supports a industry standard interface to the PHY: Reduced Media Independent Interface (RMII). The GMAC-AHB is designed to interface to the industry standard AMBA High-Performance Bus (AHB) on the application side. In TC26x B-Step an adaptation logic interfaces to the SPB bus.

The GMAC-UNIV is compliant to the following standards:

- IEEE 802.3-2002 for Ethernet MAC
- IEEE 1588-2008 standard for precision networked clock synchronization
- RMII specification from RMII consortium

Note: Although the naming convention used in this databook and the source code indicate "GMAC," the databook is also applicable to the 10/100 Universal product.

35.1.2 System Overview

35.1.2.1 System-Level Block Diagram

A system-level block diagram is shown in **Figure 35-2**.

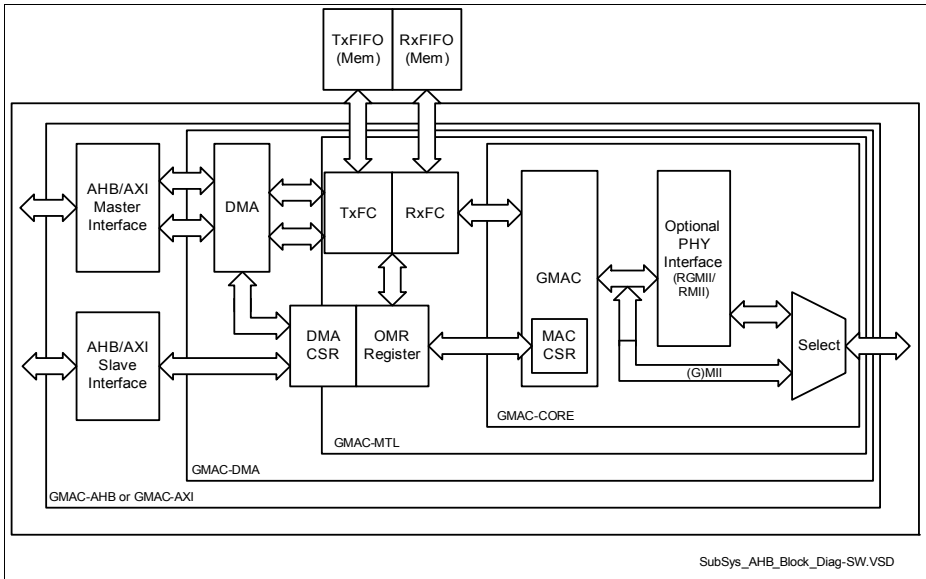


Figure 35-2 GMAC-UNIV Block Diagram

35.1.2.2 Interfaces

The GMAC-AHB transfers data to system memory through the AHB master interface. The host CPU uses the default 32-bit AHB Slave interface to access the GMAC subsystem’s Control and Status registers (CSRs).

The GMAC-UNIV supports the following PHY interface:

- Reduced MII (RMII)

35.1.2.3 Transmit and Receive FIFOs

The Transmit FIFO (Tx FIFO) buffers data read from system memory by the DMA before transmission by the GMAC Core. Similarly, the Receive FIFO (Rx FIFO) stores the Ethernet frames received from the line until they are transferred to system memory by

Ethernet MAC (ETH)

the DMA. These are asynchronous FIFOs, as they also transfer the data between the application clock and the GMAC line clocks.

Tx/Rx FIFOs are 32 bits wide and 4/4 kByte deep.

35.1.3 Features List

The GMAC-UNIV includes the following features, listed by category.

35.1.3.1 GMAC Core Features

- Supports 10/100-Mbit/s data transfer rates with the following PHY interfaces
 - RMIi interface to communicate with an external Fast Ethernet PHY
- Supports both full-duplex and half-duplex operation
 - Supports CSMA/CD Protocol for half-duplex operation
 - Supports IEEE 802.3x flow control for full-duplex operation
 - Optional forwarding of received pause control frames to the user application in full-duplex operation
 - Back-pressure support for half-duplex operation
 - Automatic transmission of zero-quanta pause frame on deassertion of flow control input in full-duplex operation
- Preamble and start-of-frame data (SFD) insertion in Transmit, and deletion in Receive paths
- Automatic CRC and pad generation controllable on a per-frame basis
- Options for Automatic Pad/CRC Stripping on receive frames
- Programmable frame length to support Standard or Jumbo Ethernet frames with sizes up to 16 KB
- Programmable InterFrameGap (40-96 bit times in steps of 8)
- Supports a variety of flexible address filtering modes:
 - Up to 31 additional 48-bit perfect (DA) address filters with masks for each byte
 - Up to 31 48-bit SA address comparison check with masks for each byte
 - 64-bit Hash filter (optional) for multicast and uni-cast (DA) addresses
 - Option to pass all multicast addressed frames
 - Promiscuous mode support to pass all frames without any filtering for network monitoring
 - Passes all incoming packets (as per filter) with a status report
- Separate 32-bit status returned for transmission and reception packets
- Supports IEEE 802.1Q VLAN tag detection for reception frames
- Separate transmission, reception, and control interfaces to the Application
- Supports 32-bit data transfer interface on the system-side
- Complete network statistics (optional) with RMON/MIB Counters (RFC1757/RFC2819/RFC2665). It is completely under control of higher protocol level (SW) to make use of these counters.
- MDIO Master interface for PHY device configuration and management, e.g. for switching the PHY in external loopback mode.
- Optional module for detection of LAN wake-up frames and AMD Magic Packet frames

Ethernet MAC (ETH)

- Optional Receive module for checksum off-load for received IPv4 and TCP packets encapsulated by the Ethernet frame
- Optional Enhanced Receive module for checking IPv4 header checksum and TCP, UDP, or ICMP checksum encapsulated in IPv4 or IPv6 datagrams.
- Optional module to support Ethernet frame time stamping as described in IEEE 1588-2008. Sixty-four-bit time stamps are given in each frame's transmit or receive status.

35.1.3.2 DMA Block Features

The DMA block exchanges data between the MTL block and host memory. A set of registers (DMA CSR) to control DMA operation is accessible by the host.

DMA features include:

- 32-bit data transfers
- Single-channel Transmit and Receive engines
- Fully synchronous design operating on a single system clock (except for CSR module, when a separate CSR clock is configured)
- Optimization for packet-oriented DMA transfers with frame delimiters
- Byte-aligned addressing for data buffer support
- Dual-buffer (ring) or linked-list (chained) descriptor chaining
- Descriptor architecture, allowing large blocks of data transfer with minimum CPU intervention; each descriptor can transfer up to 8 KB of data
- Comprehensive status reporting for normal operation and transfers with errors
- Individual programmable burst size (SINGLE, INCR4/8, not supported by the System: INCR of undefined length) for Transmit and Receive DMA Engines for optimal host bus utilization
- Programmable interrupt options for different operational conditions
- Per-frame Transmit/Receive complete interrupt control
- Round-robin or fixed-priority arbitration between Receive and Transmit engines
- Start/Stop modes
- Separate ports for host CSR access and host data interface

35.1.3.3 Transaction Layer (MTL) Features

The MTL block consists of two sets of FIFOs: a Transmit FIFO with programmable threshold capability, and a Receive FIFO with a configurable threshold (default of 64 bytes).

MTL features include:

- 32-bit Transaction Layer block providing a bridge between the application and the GMAC-CORE
- Single-channel Transmit and Receive engines
- Data transfers executed using simple FIFO-protocol

Ethernet MAC (ETH)

- Synchronization for all clocks in the design (Transmit, Receive and system clocks)
- Optimization for packet-oriented transfers with frame delimiters
- Four Separate ports for system-side and GMAC-CORE-side transmission and reception
- Two RAM-based asynchronous FIFOs with synchronous/asynchronous Read and Write operation with respect to the Read and Write clocks (one for transmission and one for reception)
- Receive Status vectors inserted into the Receive FIFO after the EOF transfer enables multiple-frame storage in the Receive FIFO without requiring another FIFO to store those frames' Receive Status.
- Configurable Receive FIFO threshold (default fixed at 64 bytes) in Cut-Through mode
- Option to filter all error frames on reception and not forward them to the application in Store-and-Forward mode
- Option to forward under-sized good frames
- Supports statistics by generating pulses for frames dropped or corrupted (due to overflow) in the Receive FIFO
- Supports Store and Forward mechanism for transmission to the GMAC core
- Supports threshold control for transmit buffer management
- Supports configurable number of frames to be stored in FIFO at any time. The default is up to 8 frames in GMAC-MTL configuration.
- Automatic generation of PAUSE frame control or backpressure signal to the GMAC core based on Receive FIFO-fill (threshold configurable) level.
- Handles automatic retransmission of Collision frames for transmission
- Discards frames on late collision, excessive collisions, excessive deferral and underrun conditions
- Software control to flush Tx FIFO
- Data FIFO RAM chip-select disabled when inactive, to reduce power consumption
- Optional module to calculate and insert IPv4 header checksum and TCP, UDP, or ICMP checksum in frames transmitted in Store-and-Forward mode.

35.1.3.4 Monitoring, Test, and Debugging Support Features

- Supports internal loopback on the MII for debugging
- external loopback is supported via the integrated MDIO controlling the PHY
- DMA states (Tx and Rx) given as status bits
- Debug status register that gives status of FSMs in Transmit and Receive data-paths and FIFO fill-levels.
- Application Abort status bits
- MMC (RMON) module in the GMAC core
- Current Tx/Rx Buffer pointer as status registers
- Current Tx/Rx Descriptor pointer as status registers

35.2 Architecture

35.2.1 Introduction

The GMAC-UNIV is a highly configurable product with multiple interfaces to the system side or the PHY side. Major modules and functions are only selected when required. A block diagram of the GMAC-UNIV's system configuration is provided in [Figure 35-3](#).

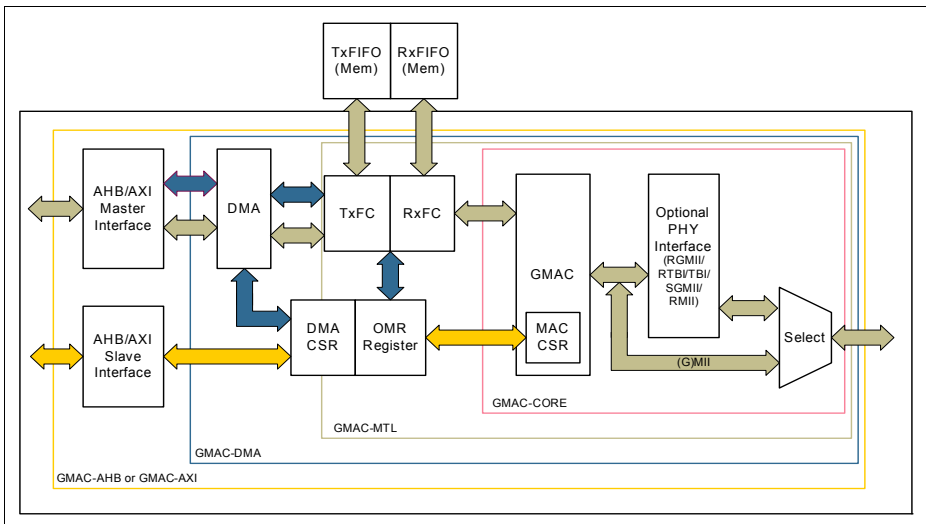


Figure 35-3 GMAC-UNIV Block Diagram

The following sections describe the functionality of the major blocks and timing behavior of the interfaces.

- [Section 35.2.2](#) provides an overview of precision network time synchronization using the Precision Time Protocol established under the IEEE 1588 standard.
- [Section 35.2.3](#) describes the functionality of the AHB interfaces in GMAC-AHB configuration
- [Section 35.2.4](#) describes the functioning of the DMA.
- [Section 35.2.5](#) explains the functioning of the MAC Transaction Layer (MTL) module which controls the Receive and Transmit FIFO memories.
- [Section 35.2.6](#) describes the functionality of the (G)MAC core transmitter and receiver.
- [Section 35.2.7](#), [Section 35.2.8](#), and [Section 35.2.8](#) describe the optional RMON, Power Management, and SMA interface modules.
- [Section 35.2.11](#) describes the Interrupt signal structure in the (G)MAC core.

35.2.2 IEEE 1588-2002 Overview

The IEEE 1588 standard defines a protocol enabling precise synchronization of clocks in measurement and control systems implemented with technologies such as network communication, local computing, and distributed objects. The protocol applies to systems communicating by local area networks supporting multicast messaging, including (but not limited to) Ethernet. This protocol enables heterogeneous systems that include clocks of varying inherent precision, resolution, and stability to synchronize. The protocol supports system-wide synchronization accuracy in the sub-microsecond range with minimal network and local clock computing resources.

The message-based protocol, named Precision Time Protocol (PTP), is transported over UDP/IP. The system or network is classified into Master and Slave nodes for distributing the timing/clock information. The protocol's technique for synchronizing a slave node to a master node by exchanging PTP messages is depicted in [Figure 35-4](#).

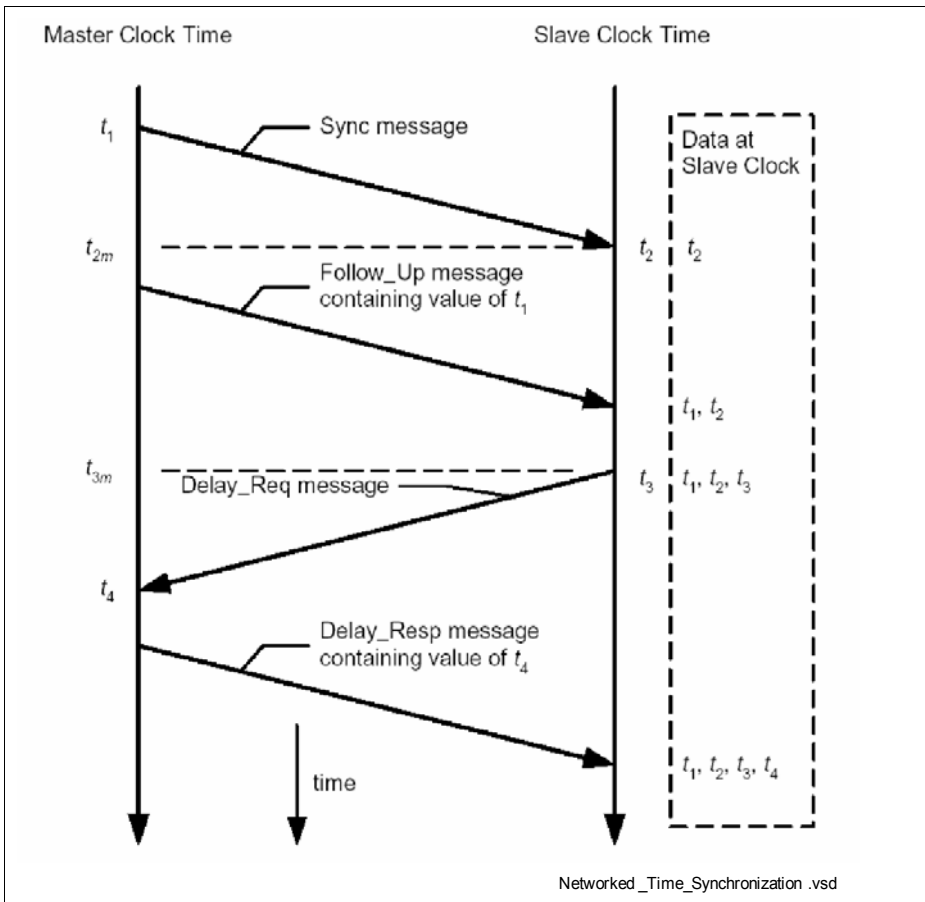


Figure 35-4 Networked Time Synchronization

1. The master broadcasts PTP Sync messages to all its nodes. The Sync message contains the master's reference time information. The time at which this message leaves the master's system is t_1 and must, for Ethernet ports, be captured at the MII.
2. A slave receives the Sync message and also captures the exact time, t_2 , using its timing reference.
3. The master then sends the slave a Follow_up message, which contains t_1 information for later use.
4. The slave sends the master a Delay_Req message, noting the exact time, t_3 , at which this frame leaves the MII.

Ethernet MAC (ETH)

5. The master receives this message, capturing the exact time, t_4 , at which it enters its system.
6. The master sends the t_4 information to the slave in the Delay_Resp message.
7. The slave uses the four values of t_1 , t_2 , t_3 , and t_4 to synchronize its local timing reference to the master's timing reference.

Most of the protocol implementation occurs in the software, above the UDP layer. As described above, however, hardware support is required to capture the exact time when specific PTP packets enter or leave the Ethernet port at the MII. This timing information must be captured and returned to the software for the proper implementation of PTP with high accuracy.

35.2.2.1 Reference Timing Source

To get a snapshot of the time, the core requires a reference time in 64-bit format (split into two 32-bit channels, with the upper 32-bits providing time in seconds, and the lower 32-bits indicating time in nanoseconds) as defined in the IEEE 1588 specification.

Internal Reference Time

This takes only the reference clock input and uses it to generate the Reference time (also called the System Time) internally and use it to capture time stamps. The generation, update, and modification of the System Time are described in [System Time Register Module](#).

35.2.2.2 Transmit Path Functions

When a frame's SFD is output on the MII, a time stamp is captured. Frames for which capturing a time stamp is required are controllable on a per-frame basis. In other words, each transmit frame can be marked to indicate whether or not a time stamp must be captured for that frame.

No snooping or processing of the transmitted frames is performed to identify PTP frames. Frame-wise control is exercised through control bits in the transmit descriptor (as described in [Descriptor Format With IEEE 1588 Time Stamping Enabled](#)).

Captured time stamps are returned to the application in a manner similar to that in which status is provided for frames. The time stamp is returned to software inside the corresponding transmit descriptor, thus connecting the time stamp automatically to the specific PTP frame. The 64-bit time stamp information is written back to the TDES2 and TDES3 fields, with TDES2 holding the time stamp's 32 least significant bits, except as described in [Transmit Time Stamp Field](#).

Note: When the alternate (enhanced) descriptor is selected, the 64-bit time-stamp is written in TDES6 and TDES7, respectively

35.2.2.3 Receive Path Functions

When the IEEE 1588 time-stamping feature is selected and enabled, the Ethernet MAC captures the time stamp of all frames received on the MII. No snooping or processing of the received frames is performed to identify PTP frames in the default mode (Advanced Time Stamp feature is not selected).

The core returns the time-stamp to the software in the corresponding receive descriptor. The 64-bit time stamp information is written back to the RDES2 and RDES3 fields, with RDES2 holding the time stamp's 32 least significant bits, except as mentioned in [Receive Time Stamp](#). The time stamp is only written to the receive descriptor for which the Last Descriptor status field has been set to 1 (the EOF marker). When the time stamp is not available (for example, due to an RxFIFO overflow) an all-ones pattern is written to the descriptors (RDES2 and RDES3), indicating that time stamp is not correct. If the software uses a control register bit to disable time stamping, the DMA does not alter RDES2 or RDES3.

Note: When the alternate (enhanced) descriptor is selected, the 64-bit time-stamp is written in RDES6 and RDES7, respectively. RDES0[7] will indicate whether the time-stamp is updated in RDES6/7 or not.

35.2.2.4 Time Stamp Error Margin

According to the IEEE 1588 specifications, the time stamp must be captured at the SFD of transmitted and received frames at the MII interface. Since the reference timing source (the PTP clock, `clk_ptp_ref_i`) is different from the MII clocks, a small error margin is introduced, due to the transfer of information across asynchronous clock domains.

In the transmit path, the captured and reported time stamp has a maximum error margin of 2 PTP clocks. In other words, the captured time stamp has the value of the reference time source given within 2 clocks after the SFD has been transmitted on the MII.

Similarly, on the receive path, the error margin is 3 MII clocks, plus up to 2 PTP clocks. You can ignore the error margin due to the 3 MII clocks by assuming that this constant delay is present in the system (or link) before the SFD data reaches the GMAC's MII interface.

35.2.2.5 Frequency Range of Reference Timing Clock

Because asynchronous logic is in place for time stamp information transfers across clock domain, a minimum delay is required between two consecutive time stamp captures. This delay is 3 clock cycles of both the MII and PTP clocks. If the gap is shorter, the GMAC does not take a time stamp snapshot for the second frame.

The maximum PTP clock frequency is limited by the maximum resolution of the reference time and the timing constraints achievable for logic operating on the PTP clock. Another factor to consider is that the resolution, or granularity, of the reference time source determines the accuracy of the synchronization. Hence, a higher PTP clock

Ethernet MAC (ETH)

frequency gives better system performance. The minimum PTP clock frequency depends on the time required between two consecutive SFD bytes. Because the MII clock frequency is fixed by IEEE specification, the minimum PTP clock frequency required for proper operation depends on the core's operating mode and operating speed.

For example, in 100 Mbit/s full-duplex operation, the minimum gap between two SFDs is 160 MII clocks (128 clocks for a 64-byte frame + 24 clocks of min IFG + 8 clocks of preamble).

In the example, $(3 \times \text{PTP}) + (3 \times \text{MII}) \leq 160 \times \text{MII}$; thus, the minimum PTP clock frequency is about 0.5 MHz ($((160 - 3) \times 40 \text{ ns} / 3 = 2.093 \text{ ns period})$)

35.2.2.6 Advanced Time Stamp Feature Support

In addition to the basic features for time stamp mentioned in [IEEE 1588-2002 Overview](#), the advanced time stamp option has the following features.

- Support for the IEEE 1588-2008 (Version 2) timestamp format.
- Option to take snapshot for all frames or for PTP type frames.
- Option for taking snapshot for event messages only.
- Option to take the snapshot based on the clock type (ordinary, boundary, end-to-end and peer-to-peer)
- Option to select the node to be a Master or Slave for ordinary and boundary clock.
- Identification of PTP message type, version, and PTP payload sent directly over Ethernet given as status.
- Option to measure time in digital or binary format.

Peer-to-Peer PTP (Pdelay) Transparent Clock (P2P TC) Message Support

The IEEE 1588-2008 version supports Pdelay message in addition to SYNC, Delay Request, Follow-up and Delay Response messages. [Figure 35-5](#) shows the method to calculate the propagation delay in clocks supporting peer-to-peer path correction.

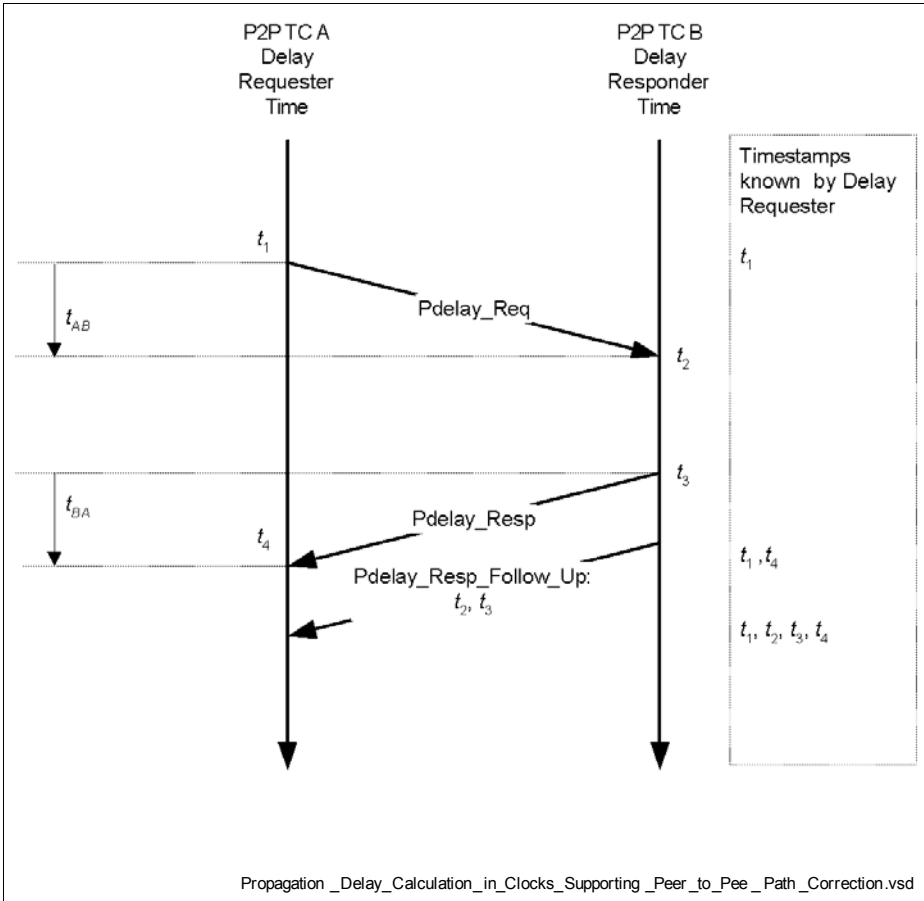


Figure 35-5 Propagation Delay Calculation in Clocks Supporting Peer-to-Peer Path Correction

The link delay measurement starts with port-1 issuing a “Pdelay_Req” message and generating a timestamp, for the Pdelay_Req message. Port-2 receives the “Pdelay_Req” message and generates a timestamp, t_2 , for this message. Port-2 returns a Pdelay_Resp message and generates a timestamp, t_3 , for this message. To minimize errors due to any frequency offset between the two ports, Port-2 returns the Pdelay_Resp message as quickly as possible after the receipt of the Pdelay_Req message.

Port-2 either:

Ethernet MAC (ETH)

- Returns the difference between the timestamps t2 and t3 in the Pdelay_Resp message,
- Returns the difference between the timestamps t2 and t3 in a Pdelay_Resp_Follow_Up message, or
- Returns the timestamps t2 and t3 in the Pdelay_Resp and Pdelay_Resp_Follow_Up messages respectively.

Port-1 generates a timestamp, t4, upon receiving the Pdelay_Resp message. Port-1 then uses these four timestamps to compute the mean link delay.

Clock Types

The type of clock nodes supported in IEEE 1588-2008 is described in this section. The corresponding support provided by the advanced time stamp feature for each of the clock type is also mentioned.

1. Ordinary clock support: In this type the clock can be a grandmaster or a slave clock. This clock has a single PTP state.

Table 35-1 shows the messages for which time-stamp snapshot is taken on the receive side for Master and Slave nodes.

The ordinary clock in the domain supports a single copy of the protocol and has a single PTP state and will typically be a single physical port. In typical industrial automation applications, an ordinary clock is associated with an application device such as sensors and actuators. In telecom applications, the ordinary clock may be associated with a timing demarcation device.

Typically for ordinary clock, you will need to take snapshot for only one type of PTP messages. For e.g. you will require supporting either version 1 or 2 PTP messages, not both.

The following features are supported.

- a) Sends and receives PTP messages. The time stamp snapshot can be controlled as described in **32-bit Register - Timestamp_Control**.
 - b) Maintains the data sets (e.g., time stamp values).
2. Boundary clock support: This type of clock is similar to the ordinary clock except for the following.

Hence the features of ordinary clock holds good for the boundary clock also.

The boundary clock typically has several physical ports communicating with the network. The messages related to synchronization, master-slave hierarchy and signaling terminate in the protocol engine of the boundary clock and are not forwarded. The PTP message type status given by the core (refer to **Receive Path Functions**) will help you to quickly identify the type of message and take appropriate action.

- a) The clock data sets are common to all ports of the boundary clock
 - b) The local clock is common to all ports of the boundary clock.
3. End to end transparent clock support: The end-to-end transparent clock forwards all messages like normal bridge, router or repeater. The resident time needs to be

Ethernet MAC (ETH)

computed to update the correctionField. Hence snapshot needs to be taken for the messages mentioned in [Table 35-2](#).

In the end-to-end transparent clock, the residence times are accumulated in a special field (correctionField) of the PTP event (SYNC) message or the associated Follow-up (FOLLOW_UP) Message. Hence it is important to take a snapshot for these messages alone. This can be quickly done by setting the control bit (TSEVNTENA), which enables snapshot to be taken for event messages and also selecting the type of clock in the “Time Stamp Control Register”.

The residence time is also corrected for Delay_Req messages (but snapshot of the timestamp is not required). The message type statuses provided helps you to quickly identify the message and update the correctionField.

The message type status provided will also help in taking appropriate action depending on the type of PTP message received.

4. Peer to peer transparent clock support: In this type of clock the computation of the link delay is based on an exchange of Pdelay_Req, Pdelay_Resp and Pdelay_Resp_Follow_Up messages with the link peer. Hence support for taking snapshot for the event messages related to Pdelay is added. [Table 35-3](#).

The transparent clock corrects only the SYNC and Follow-up message. As discussed earlier this can be achieved using the message status provided.

The type of clock to be implemented will be configurable through control register, as mentioned in [32-bit Register - Timestamp_Control](#). To ensure that the snapshot is taken only for the messages indicated in the table for the corresponding clock type, the “TSEVNTENA: Enable Time Stamp Snapshot for Event Messages” bit has to be set.

Table 35-1 PTP Messages for which Snapshot is Taken on Receive Side for Ordinary Clock

Master	Slave
Delay_Req	SYNC

Table 35-2 PTP Messages for which Snapshot is Taken for Transparent Clock Implementation

SYNC
FOLLOW_UP

Table 35-3 PTP Messages for which Snapshot is Taken for Peer-to-Peer Transparent Clock Implementation

SYNC

Pdelay_Req

Pdelay_Resp

PTP Processing and Control

The common message header for PTP messages is shown below. This format is taken from IEEE standard 1588-2008 (Revision of IEEE Std. 1588-2002).

Table 35-4 Message Format Defined in IEEE 1588-2008

BITS		OCTETS	OFFSET
transportSpecific	messageType	1	0
Reserved	versionPTP	1	1
messageLength		2	2
domainNumber		1	4
Reserved		1	5
flagField		2	6
correctionField		8	8
Reserved		4	16
sourcePortIdentity		10	20
sequenceId		2	30
controlField ¹⁾		1	32
logMessageInterva		1	33

1) controlField is used in version 1. For version 2, messageType field will be used for detecting different message types.

There are some fields in the PTP frame that are used to detect the type and control the snapshot to be taken. This is different for PTP frames sent directly over Ethernet, PTP frames sent over UDP / IPv4 and PTP frames that are sent over UDP / IPv6. The following sections provide information on the fields that are used to control taking the snapshot.

PTP Frame Over IPv4

Table 35-5 gives the details of the fields that will be matched to control snapshot for PTP packets over UDP over IPv4 for IEEE 1588 version 1 and 2. Note that the octet positions

Ethernet MAC (ETH)

for tagged frames will be offset by 4. This is based on Appendix-D of the IEEE 1588-2008 standard and the message format defined in [Table 35-4](#).

Table 35-5 IPv4-UDP PTP Frame Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
MAC Frame type	12, 13	0x0800	IPv4 datagram
IP Version and Header Length	14	0x45	IP version is IPv4
Layer-4 protocol	23	0x11	UDP
IP Multicast address (IEEE 1588 version 1)	30, 31, 32, 33	0xE0,0x00, 0x01,0x81 (or 0x82 or 0x83 or 0x84)	Multicast IPv4 addresses allowed. 224.0.1.129 224.0.1.130 224.0.1.131 224.0.1.132
IP Multicast address (IEEE 1588 version 2)	30, 31, 32, 33	0xE0, 0x00, 0x01, 0x81 (Hex) 0xE0, 0x00, 0x00, 0x6B (Hex)	PTP-primary multicast address: 224.0.1.129 PTP-Pdelay multicast address: 224.0.0.107
UDP destination port	36, 37	0x013F, 0x0140	0x013F – PTP event message ¹⁾ 0x0140 – PTP general messages
PTP control field (IEEE version 1)	74	0x00/0x01/0x02 /0x03/0x04	0x00 – SYNC, 0x01 – Delay_Req, 0x02 – Follow_Up 0x03 – Delay_Resp 0x04 – Management

Ethernet MAC (ETH)
Table 35-5 IPv4-UDP PTP Frame Fields Required for Control and Status (cont'd)

Field Matched	Octet Position	Matched Value	Description
PTP Message Type Field (IEEE version 2)	42 (nibble)	0x0/0x1/0x2/0x3/0x8/0x9/0xB/0xC/0xD	0x0 – SYNC 0x1 – Delay_Req 0x2 – Pdelay_Req 0x3 – Pdelay_Resp 0x8 – Follow_Up 0x9 – Delay_Resp 0xA – Pdelay_Resp_Follow_Up 0xB – Announce 0xC – Signaling 0xD – Management
PTP version field	43 (nibble)	0x1 or 0x2	0x1 – Supports PTP version 1 0x2 – Supports PTP version 2

1) PTP event messages are SYNC, Delay_Req (IEEE 1588 version 1 and 2) or Pdelay_Req, Pdelay_Resp (IEEE 1588 version 2 only).

PTP Frame Over IPv6

Table 35-6 gives the details of the fields that will be matched to control snapshot for PTP packets over UDP over IPv6 for IEEE 1588 version 1 and 2. Note that the octet positions for tagged frames will be offset by 4. This is based on Appendix-E of the IEEE 1588-2008 standard and the message format defined in **Table 35-4**.

Table 35-6 IPv6-UDP PTP Frame Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
MAC Frame type	12, 13	0x86DD	IP datagram
IP version	14 (bits [7:4])	0x6	IP version is IPv6
Layer-4 protocol	20 ¹⁾	0x11	UDP
PTP Multicast address	38 – 53	FF0x:0:0:0:0:0:0:0: 181 (Hex) FF02:0:0:0:0:0:0:0: 6B (Hex)	PTP – primary multicast address: FF0x:0:0:0:0:0:0:0:181 (Hex) PTP – Pdelay multicast address: FF02:0:0:0:0:0:0:0:6B (Hex)

Ethernet MAC (ETH)
Table 35-6 IPv6-UDP PTP Frame Fields Required for Control and Status (cont'd)

Field Matched	Octet Position	Matched Value	Description
UDP destination port	56, 57 (*)	0x013F, 0x140	0x013F – PTP event message 0x0140 – PTP general messages
PTP control field (IEEE 1588 Version 1)	93 (*)	0x00/0x01/0x02/0x03/0x04	0x00 – SYNC, 0x01 – Delay_Req, 0x02 – Follow_Up 0x03 – Delay_Resp 0x04 – Management (version1)
PTP Message Type Field (IEEE version 2)	74 (*) (nibble)	0x0/0x1/0x2/0x3/0x8/0x9/0xB/0xC/0xD	0x0 – SYNC 0x1 – Delay_Req 0x2 – Pdelay_Req 0x3 – Pdelay_Resp 0x8 – Follow_Up 0x9 – Delay_Resp 0xA – Pdelay_Resp_Follow_Up 0xB – Announce 0xC – Signaling 0xD - Management
PTP version field	75 (nibble)	0x1 or 0x2	0x1 – Supports PTP version 1 0x2 – Supports PTP version 2

1) The Extension Header is not defined for PTP packets.

PTP Frame Over Ethernet

Table 35-7 gives the details of the fields that will be matched to control snapshot for PTP packets over Ethernet for IEEE 1588 version 1 and 2. Note that the octet positions for tagged frames will be offset by 4. This is based on Appendix-E of the IEEE 1588-2008 standard and the message format defined in **Table 35-4**.

Table 35-7 Ethernet PTP Frame Fields Required for Control And Status

Field Matched	Octet Position	Matched Value	Description
MAC Frame type	12, 13	0x88F7	PTP Ethernet frame.
PTP control field (IEEE Version 1)	45	0x00/0x01/0x02/ 0x03/0x04	0x00 – SYNC 0x01 – Delay_Req 0x02 – Follow_Up 0x03 – Delay_Resp 0x04 – Management
PTP Message Type Field (IEEE version 2)	14 (nibble)	0x0/0x1/0x2/0x3/0x8/ 0x9/0xB/ 0xC/0xD	0x0 – SYNC 0x1 – Delay_Req 0x2 – Pdelay_Req 0x3 – Pdelay_Resp 0x8 – Follow_Up 0x9 – Delay_Resp 0xA – Pdelay_Resp_Follow_Up 0xB – Announce 0xC – Signaling 0xD – Management
MAC Destination multicast address ¹⁾	0-5	01-1B-19-00-00-00 01-80-C2-00-00-0E	All except peer delay messages - 01-1B-19-00-00-00 Pdelay messages - 01-80-C2-00-00-0E
PTP version field	15 (nibble)	0x1 or 0x2	0x1 – Supports PTP version 1 0x2 – Supports PTP version 2

1) In addition, the address match of destination addresses (DA) programmed in MAC address 1 to 31 will be used, if the control bit 18 (TSENMACADDR: Enable MAC address for PTP frame filtering) of the Time Stamp Control register is set.

Reference Timing Source (for Advance Timestamp Feature)

The updated functionality for advanced timestamp support is mentioned in the following points.

1. The IEEE 1588-2008 standard defines the seconds field of the time to be 48 bits wide. The fields to time-stamp will be the following.

- a) UInteger48- seconds field
- b) UInteger32-nanoseconds field

The “seconds” field is the integer portion of the timestamp in units of seconds. The

Ethernet MAC (ETH)

“nanoseconds” field is the fractional portion of the timestamp in units of nanoseconds. E.g. 2.000000001 seconds is represented as secondsField = 0x0000_0000_0002 and nanoSeconds = 0x0000_0001. Thus the maximum value in nanoseconds field in this format will be 0x3B9A_C9FF value (i.e (10e9-1) nanoseconds). This is defined as digital rollover mode of operation. It will also support the older mode in which the nano-seconds field will roll-over and increment the seconds field after the value of 0x7FFF_FFFF. (Accuracy is ~0.466 ns per bit). This is defined as the binary rollover mode. The modes can be controlled using the “TSCTRLSSR: Time Stamp Digital or Binary rollover control” bit of the time stamp control register.

2. When the Advanced IEEE 1588 time-stamp feature is selected time maintained in the core will still be 64-bit wide, as the overflow to the upper 16-bits of seconds register happens once in 130 years. The value of the upper 16-bits of the seconds field can only be obtained from the CSR register. This is optional and can be controlled by the coreKit parameter “IEEE1588 Higher Word Register Enable” described in Chapter 6. A CSR status bit which indicates if the 32-bit “seconds” field has overflowed is also provided.
3. There is also a pulse-per-second output given to indicate 1 second interval (default). Option is provided to change the interval. Refer [32-bit Register - Timestamp_Control](#) for more information.
4. Option to take auxiliary time-stamp snapshot with an external event is supported. Refer to section [Auxiliary Snapshot](#) for details.

Transmit Path Functions

There are no changes in the transmit path functions for GMAC-CORE and GMAC-MTL configuration for the Advanced time stamp option.

The structure of the descriptor changes when Advanced IEEE 1588 version support is enabled. The IEEE 1588 timestamp feature is supported using Alternate (Enhanced) descriptors format only. The descriptor is 32-bytes long (8 DWORDS) and the snapshot of the timestamp is written in descriptor 6 and 7.

Receive Path Functions

When the advanced time stamp feature is selected, processing of the received frames to identify valid PTP frames is done. The snapshot of the time to be sent to the application can be controlled.

The following options are provided in the time stamp control register to control the snapshot.

1. Option to enable snapshot for all frames.
2. Enable snapshot for IEEE 1588 version 2 or version 1 time stamp.
3. Enable snapshot for PTP frames transmitted directly over Ethernet or UDP-IP-Ethernet.

Ethernet MAC (ETH)

4. Enable time stamp snapshot for the received frame for IPv4 or IPv6.
5. Enable time stamp snapshot for EVENT messages (SYNC, DELAY_REQ, PDELAY_REQ or PDELAY_RESP) only.
6. Enable the node to be a Master or Slave. This will control the type of messages for which snap-shot will be taken (this depends on the type of clock that is selected and is valid for ordinary or boundary clock only).

Note that PTP messages over VLAN frames are also supported.

The time stamp is returned to the software inside the corresponding Transmit and Receive Descriptor. The Advanced time stamp feature is supported using Alternate (Enhanced) descriptors only which is 32-bytes long. Extended status containing the time stamp message status and the IPC offload status is written back in descriptor 4 and the snapshot of the time stamp is written back in descriptor 6 and 7. The complete details on the descriptors when the advanced time stamping is enabled are provided in [Alternate or Enhanced Descriptors](#).

Auxiliary Snapshot

The auxiliary snap shot feature allows you to store a snapshot of the system time based on an external event. The event is considered to be the rising edge of the sideband signal `ptp_aux_ts_trig_i`. This feature is independent of whether the system time is generated internally or given as input (on `ptp_timestamp_i` bus). Such snap-shots are stored in a 4-deep FIFO. Only 64-bit of the time will be stored in the FIFO. The upper 16-bits of the seconds register can be read from “Time Stamp Higher Word Register” when it is present.

The storing of the snapshot can be indicated to the host with an interrupt. The value of the snapshot is read out through a FIFO register access (see [32-bit Register - Timestamp Control](#)). When the FIFO becomes full and an external trigger to take the snapshot is asserted, then a snapshot trigger-missed status is set in the Time Stamp Status Register. This indicates that the latest auxiliary snapshot of the timestamp was not stored in the FIFO. The latest snapshot will not be written to the FIFO when it is full. When the host reads the 64-bit timestamp from the FIFO, space is available to store the next snapshot. This FIFO can be cleared using the control bit “ATSFC: Auxiliary Snapshot FIFO clear” defined in the Time Stamp Control Register. When multiple snapshots are present in the FIFO, the count is indicated in the “ATSNS: Auxiliary Time Stamp Number of Snapshots” bits of the Time Stamp Status Register.

35.2.3 AHB Application Host Interface

In the GMAC-AHB core, the DMA Controller interfaces with the Host through the AMBA AHB Interface. The AHB Master Interface controls data transfers while the AHB Slave interface accesses CSR space. The DMA can be used in embedded applications where DMA is required to optimize data transfer between the GMAC and system memory.

The AHB Master interface converts the internal DMA request cycles into AHB cycles.

Characteristics of this interface include the following:

- Fully AMBA 2.0-compliant AHB Master: with restrictions
- You can choose fixed burst length only (SINGLE, INCR4, INCR8) by programming the FB in the DMA Bus Mode register (see [Register Description](#)).
 - When fixed burst length is chosen, the AHB master always initiates a burst with SINGLE or INCR4/8type. But when such a burst is responded with SPLIT/RETRY/early burst termination, the AHB master will re-initiate the pending transfers of the burst with INCR or SINGLE burst-length type. It will terminate such INCR bursts when the original requested fixed-burst is transferred. In Fixed Burst-Length mode, if the DMA requests a burst transfer that is not equal to INCR4/8, the AHB Host interface splits the transfer into multiple burst transactions. For example, if the DMA requests a 15-beat burst transfer, the AHB interface splits it into multiple transfers of INCR8 and INCR4 and 3 SINGLE transactions.
- The AHB slaves interfaced to the GMAC-AHB must support SINGLE and INCR transfers, at a minimum.
- Takes care of AHB SPLIT, RETRY, and ERROR conditions. Any ERROR response will halt all further transactions for that DMA, and indicate the error as fatal through the CSR and interrupt. The application must give a hard or soft reset to the module to restart the operation.
- Takes care of AHB 1K boundary breaking
- Handles all data transfers (according to the data bus width configuration), except for Descriptor Status Write accesses (which are always 32-bit). In any burst data transfer, the address bus value is always aligned to the data bus width and need not be aligned to the beat size.

All AHB burst transfers can be aligned to an address value by enabling the DMA Bus Mode register's AAL bit. If both the FB and AAL bits are set to 1, the AHB interface and the DMA together ensure that all initiated beats are aligned to the address, completing the frame transfer in the minimum number of required beats.

For example, in 32-bit data bus mode, if a data buffer transfer's start address is 0xF000_0008 and the DMA is configured for a maximum beat size of 8, the AHB transfers occur in the following sequence:

- 2 SINGLE transfers at addresses 0xF000_0008 and 0xF000_000C
- 1 INCR4 transfer at address 0xF000_0010
- All subsequent beats are INCR8 transfers starting from address 0xF000_0020. For an address-aligned INCR8 transfer, the 5 least-significant bits of the address must be 0.
- The DMA Controller requests an AHB Burst Read transfer only when it can accept the received burst data completely. Data read from the AHB is always pushed into the DMA without any delay or BUSY cycles.
- The DMA requests an AHB Burst Write transfer only when it has the sufficient data to transfer the burst completely. The AHB interface always assumes that it has data available to push into the AHB bus. However, the DMA can prematurely indicate end-

Ethernet MAC (ETH)

of-valid data (due to the transfer of end-of-frame of an Ethernet frame) during the burst. In Fixed Burst Length mode, the AHB Master interface continues the burst with dummy data until the specified length is completed.

The AHB 32-bit Slave interface provides access to the DMA and GMAC CSR space. Characteristics of this interface include the following:

- Fully AMBA 2.0 Compliant AHB Slave—no restrictions
- Supports single and INCR4/8 transfers
- Supports busy and early terminations
- Supports 32-bit, 16-bit, and 8-bit write/read transfers to the CSR; 32-bit access to the CSR are recommended to avoid any SW synchronization problems.
- Generates OKAY only response; does not generate SPLIT, RETRY, or ERROR responses.

35.2.4 DMA Controller

The DMA has independent Transmit and Receive engines, and a CSR space. The Transmit Engine transfers data from system memory to the device port (MTL), while the Receive Engine transfers data from the device port to system memory. The controller utilizes descriptors to efficiently move data from source to destination with minimal Host CPU intervention. The DMA is designed for packet-oriented data transfers such as frames in Ethernet. The controller can be programmed to interrupt the Host CPU for situations such as Frame Transmit and Receive transfer completion, and other normal/error conditions.

The DMA and the Host driver communicate through two data structures:

- Control and Status registers (CSR)
- Descriptor lists and data buffers

Control and Status registers are described in detail in [Chapter 35.3.1.1](#). Descriptors are described in detail in [Chapter 35.4](#).

The DMA transfers data frames received by the core to the Receive Buffer in the Host memory, and Transmit data frames from the Transmit Buffer in the Host memory. Descriptors that reside in the Host memory act as pointers to these buffers.

There are two descriptor lists; one for reception, and one for transmission. The base address of each list is written into DMA Registers 3 and 4, respectively. A descriptor list is forward linked (either implicitly or explicitly). The last descriptor may point back to the first entry to create a ring structure. Explicit chaining of descriptors is accomplished by setting the second address chained in both Receive and Transmit descriptors (RDES1[24] and TDES1[24]). The descriptor lists resides in the Host physical memory address space. Each descriptor can point to a maximum of two buffers. This enables two buffers to be used, physically addressed, rather than contiguous buffers in memory.

A data buffer resides in the Host physical memory space, and consists of an entire frame or part of a frame, but cannot exceed a single frame. Buffers contain only data, buffer

Ethernet MAC (ETH)

status is maintained in the descriptor. Data chaining refers to frames that span multiple data buffers. However, a single descriptor cannot span multiple frames. The DMA will skip to the next frame buffer when end-of-frame is detected. Data chaining can be enabled or disabled.

The descriptor ring and chain structure is shown in **Figure 35-6**.

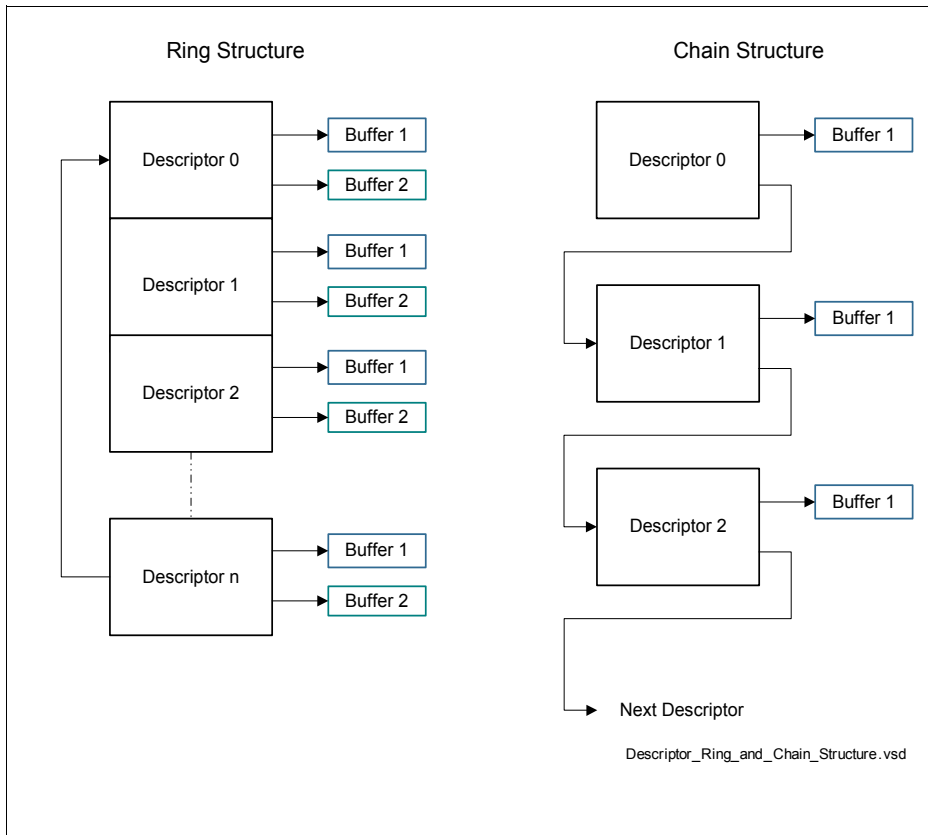


Figure 35-6 Descriptor Ring and Chain Structure

35.2.4.1 Initialization

Initialization for the GMAC is as follows.

1. Write to DMA Register 0 to set Host bus access parameters.
2. Write to DMA Register 7 to mask unnecessary interrupt causes.

Ethernet MAC (ETH)

3. The software driver creates the Transmit and Receive descriptor lists. Then it writes to both DMA Register 3 and DMA Register 4, providing the DMA with the starting address of each list.
4. Write to GMAC Registers 1, 2, and 3 for desired filtering options.
5. Write to GMAC Register 0 to configure and enable the Transmit and Receive operating modes. The PS and DM bits are set based on the auto-negotiation result (read from the PHY).
6. Write to DMA Register 6 to set bits 13 and 1 to start transmission and reception.
7. The Transmit and Receive engines enter the Running state and attempt to acquire descriptors from the respective descriptor lists. The Receive and Transmit engines then begin processing Receive and Transmit operations. The Transmit and Receive processes are independent of each other and can be started or stopped separately.

Host Bus Burst Access

The DMA will attempt to execute fixed-length Burst transfers on the AHBMaster interface if configured to do so (FB bit of DMA Register 0). The maximum Burst length is indicated and limited by the PBL field (DMA Register 0[13:8]). The Receive and Transmit descriptors are always accessed in the maximum possible (limited by PBL or $16 * 8/\text{bus width}$) burst-size for the 16-bytes to be read.

The Transmit DMA will initiate a data transfer only when sufficient space to accommodate the configured burst is available in MTL Transmit FIFO or the number of bytes till the end of frame (when it is less than the configured burst-length). The DMA will indicate the start address and the number of transfers required to the AHB Master Interface. When the AHB Interface is configured for fixed-length burst, then it will transfer data using the best combination of INCR4/8 and SINGLE transactions.

The Receive DMA will initiate a data transfer only when sufficient data to accommodate the configured burst is available in MTL Receive FIFO or when the end of frame (when it is less than the configured burst-length) is detected in the Receive FIFO. The DMA will indicate the start address and the number of transfers required to the AHB Master Interface. When the AHB Interface is configured for fixed-length burst, then it will transfer data using the best combination of INCR4/8 and SINGLE transactions. If the end-of frame is reached before the fixed-burst ends on the AHB interface, then dummy transfers are performed in-order to complete the fixed-burst.

When the AHB interface is configured for address-aligned beats, both DMA engines ensure that the first burst transfer the AHB initiates is less than or equal to the size of the configured PBL. Thus, all subsequent beats start at an address that is aligned to the configured PBL. The DMA can only align the address for beats up to size 8 (for $\text{PBL} > 8$), because the AHB interface does not support more than INCR8.

Host Data Buffer Alignment

The Transmit and Receive data buffers do not have any restrictions on start address alignment. For example, in systems with 32-bit memory, the start address for the buffers can be aligned to any of the four bytes. However, the DMA always initiates transfers with address aligned to the bus width with dummy data for the byte lanes not required. This typically happens during the transfer of the beginning or end of an Ethernet frame.

Example - Buffer Read

If the Transmit buffer address is 32'h0000FF2 (for 32-bit data bus), and 15 bytes need to be transferred, then the DMA will read five full words from address 32'h0000FF0, but when transferring data to the MTL Transmit FIFO, the extra bytes (the first two bytes) will be dropped or ignored. Similarly, the last 3 bytes of the last transfer will also be ignored. The DMA always ensures it transfers a full 32-bit data to the MTL Transmit FIFO, unless it is the end-of-frame.

Buffer Size Calculations

The DMA does not update the size fields in the Transmit and Receive descriptors. The DMA updates only the status fields (RDES and TDES) of the descriptors. The driver has to perform the size calculations.

The transmit DMA transfers the exact number of bytes (indicated by buffer size field of TDES1) towards the GMAC core. If a descriptor is marked as first (FS bit of TDES1 is set), then the DMA marks the first transfer from the buffer as the start of frame. If a descriptor is marked as last (LS bit of TDES1), then the DMA marks the last transfer from that data buffer as the end-of frame to the MTL.

The Receive DMA transfers data to a buffer until the buffer is full or the end-of frame is received from the MTL. If a descriptor is not marked as last (LS bit of RDES0), then the descriptor's corresponding buffer(s) are full and the amount of valid data in a buffer is accurately indicated by its buffer size field minus the data buffer pointer offset when the FS bit of that descriptor is set. The offset is zero when the data buffer pointer is aligned to the data bus width. If a descriptor is marked as last, then the buffer may not be full (as indicated by the buffer size in RDES1). To compute the amount of valid data in this final buffer, the driver must read the frame length (FL bits of RDES0[29:16]) and subtract the sum of the buffer sizes of the preceding buffers in this frame. The Receive DMA always transfers the start of next frame with a new descriptor.

Note: Even when the start address of a receive buffer is not aligned to the system bus's data width, the system should allocate a receive buffer of a size aligned to the system bus width. For example, if the system allocates a 1.024-byte (1 KB) receive buffer starting from address 0x1000, the software can program the buffer start address in the Receive descriptor to have a 0x1002 offset. The Receive DMA writes the frame to this buffer with dummy data in the first two locations (0x1000 and 0x1001). The actual frame is written from location 0x1002. Thus, the actual

Ethernet MAC (ETH)

useful space in this buffer is 1.022 bytes, even though the buffer size is programmed as 1.024 bytes, due to the start address offset.

DMA Arbiter

The arbiter inside the DMA module performs the arbitration between the Transmit and Receive channel accesses to the AHB Master interface. Two types of arbitrations are possible: round-robin, and fixed-priority.

When round-robin arbitration is selected (DA bit of DMA Register 0 is reset), the arbiter allocates the data bus in the ratio set by the PR bits of DMA Register 0, when both Transmit and Receive DMAs are requesting for access simultaneously. When the DA bit is set, the Receive DMA always gets priority over the Transmit DMA for data access.

35.2.4.2 Transmission**TxDMA Operation: Default (Non-OSF) Mode**

The Transmit DMA engine in default mode proceeds as follows:

1. The Host sets up the transmit descriptor (TDES0-TDES3) and sets the Own bit (TDES0[31]) after setting up the corresponding data buffer(s) with Ethernet Frame data.
2. Once the ST bit (DMA Register 6[13]) is set, the DMA enters the Run state.
3. While in the Run state, the DMA polls the Transmit Descriptor list for frames requiring transmission. After polling starts, it continues in either sequential descriptor ring order or chained order. If the DMA detects a descriptor flagged as owned by the Host, or if an error condition occurs, transmission is suspended and both the Transmit Buffer Unavailable (DMA Register 5[2]) and Normal Interrupt Summary (DMA Register 5[16]) bits are set. The Transmit Engine proceeds to Step 8.
4. If the acquired descriptor is flagged as owned by DMA (TDES0[31] = 1'b1), the DMA decodes the Transmit Data Buffer address from the acquired descriptor.
5. The DMA fetches the Transmit data from the Host memory and transfers the data to the MTL for transmission.
6. If an Ethernet frame is stored over data buffers in multiple descriptors, the DMA closes the intermediate descriptor and fetches the next descriptor. Steps Step 2, Step 3, and Step 4 are repeated until the end-of-Ethernet-frame data is transferred to the MTL.
7. When frame transmission is complete, if IEEE 1588 time stamping was enabled for the frame (as indicated in the transmit status) the time-stamp value obtained from MTL is written to the transmit descriptor (TDES2 and TDES3) that contains the end-of-frame buffer. The status information is then written to this transmit descriptor (TDES0). Because the Own bit is cleared during this step, the Host now owns this

Ethernet MAC (ETH)

descriptor. If time stamping was not enabled for this frame, the DMA does not alter the contents of TDES2 and TDES3.

8. Transmit Interrupt (DMA Register 5[0]) is set after completing transmission of a frame that has Interrupt on Completion (TDES1[31]) set in its Last Descriptor. The DMA engine then returns to Step 2.
9. In the Suspend state, the DMA tries to re-acquire the descriptor (and thereby return to Step 2) when it receives a Transmit Poll demand and the Underflow Interrupt Status bit is cleared.

The TxDMA transmission flow in default mode is shown in [Figure 35-7](#).

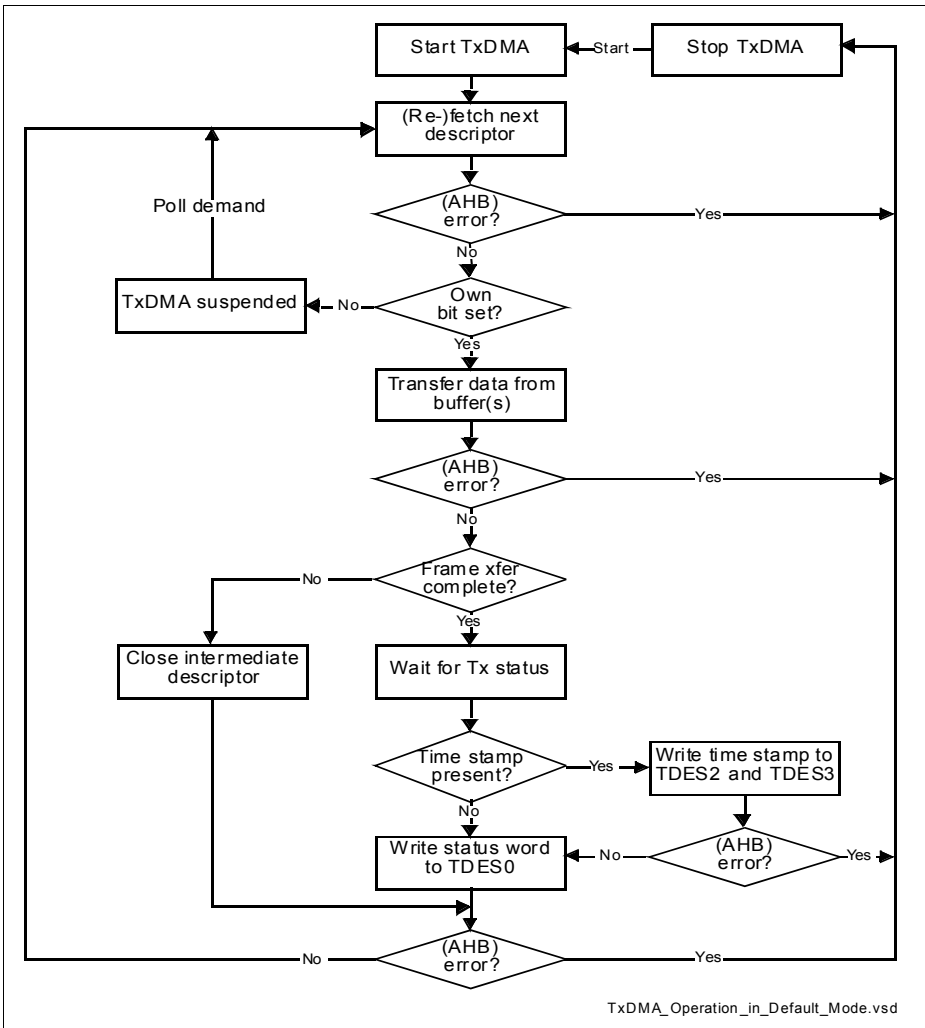


Figure 35-7 TxDMA Operation in Default Mode

TxDMA Operation: OSF Mode

While in the Run state, the transmit process can simultaneously acquire two frames without closing the Status descriptor of the first (if the OSF bit is set in DMA Register 6[2]). As the transmit process finishes transferring the first frame, it immediately polls the

Ethernet MAC (ETH)

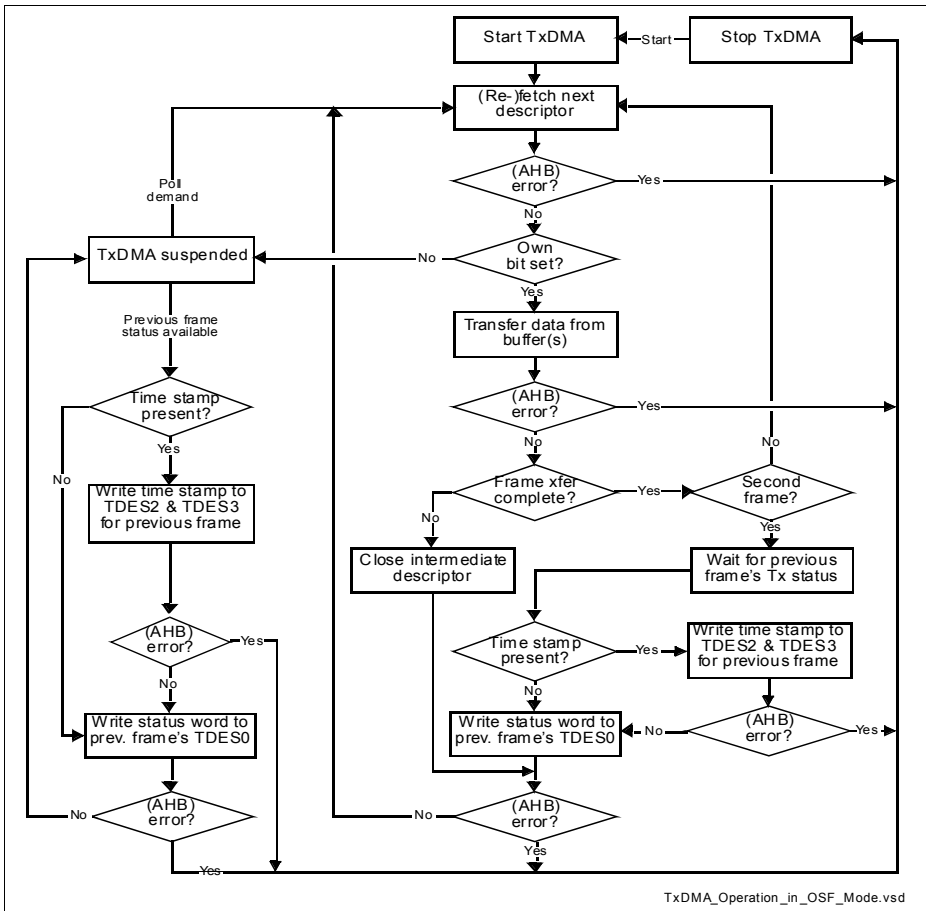
Transmit Descriptor list for the second frame. If the second frame is valid, the transmit process transfers this frame before writing the first frame's status information.

In OSF mode, the Run state Transmit DMA operates in the following sequence:

1. The DMA operates as described in Step 1–Step 5 of the TxDMA (default mode).
2. Without closing the previous frame's last descriptor, the DMA fetches the next descriptor.
3. If the DMA owns the acquired descriptor, the DMA decodes the transmit buffer address in this descriptor. If the DMA does not own the descriptor, the DMA goes into Suspend mode and skips to Step 6.
4. The DMA fetches the Transmit frame from the Host memory and transfers the frame to the MTL until the End-of-Frame data is transferred, closing the intermediate descriptors if this frame is split across multiple descriptors.
5. The DMA waits for the previous frame's frame transmission status and time stamp. Once the status is available, the DMA writes the time stamp to TDES2 and TDES3, if such time stamp was captured (as indicated by a status bit). The DMA then writes the status, with a cleared Own bit, to the corresponding TDES0, thus closing the descriptor. If time stamping was not enabled for the previous frame, the DMA does not alter the contents of TDES2 and TDES3.
6. If enabled, the Transmit interrupt is set, the DMA fetches the next descriptor, then proceeds to Step 2 (when Status is normal). If the previous transmission status shows an underflow error, the DMA goes into Suspend mode (Step 6).
7. In Suspend mode, if a pending status and time stamp are received from the MTL, the DMA writes the time stamp (if enabled for the current frame) to TDES2 and TDES3, then writes the status to the corresponding TDES0. It then sets relevant interrupts and returns to Suspend mode.
8. The DMA can exit Suspend mode and enter the Run state (go to Step 1 or Step 2 depending on pending status) only after receiving a Transmit Poll demand (DMA Register 1).

Note: As the DMA fetches the next descriptor in advance before closing the current descriptor, the descriptor chain should have more than 2 different descriptors for correct and proper operation.

The basic flow is charted in [Figure 35-8](#).


Figure 35-8 TxDMA Operation in OSF Mode

Transmit Frame Processing

The Transmit DMA expects that the data buffers contain complete Ethernet frames, excluding preamble, pad bytes, and FCS fields. The DA, SA, and Type/Len fields contain valid data. If the Transmit Descriptor indicates that the MAC core must disable CRC or PAD insertion, the buffer must have complete Ethernet frames (excluding preamble), including the CRC bytes.

Frames can be data-chained and can span several buffers. Frames must be delimited by the First Descriptor (TDES1[29]) and the Last Descriptor (TDES1[30]), respectively.

Ethernet MAC (ETH)

As transmission starts, the First Descriptor must have (TDES1[29]) set. When this occurs, frame data transfers from the Host buffer to the MTL Transmit FIFO. Concurrently, if the current frame has the Last Descriptor (TDES1[30]) clear, the Transmit Process attempts to acquire the Next Descriptor. The Transmit Process expects this descriptor to have TDES1[29] clear. If TDES1[30] is clear, it indicates an intermediary buffer. If TDES1[30] is set, it indicates the last buffer of the frame.

After the last buffer of the frame has been transmitted, the DMA writes back the final status information to the Transmit Descriptor 0 (TDES0) word of the descriptor that has the last segment set in Transmit Descriptor 1 (TDES1[30]). At this time, if Interrupt on Completion (TDES1[31]) was set, Transmit Interrupt (DMA Register 5[0]) is set, the Next Descriptor is fetched, and the process repeats.

Actual frame transmission begins after the MTL Transmit FIFO has reached either a programmable transmit threshold (DMA Register 6[16:14]), or a full frame is contained in the FIFO. There is also an option for Store and Forward Mode (DMA Register 6[21]). Descriptors are released (Own bit TDES0[31] clears) when the DMA finishes transferring the frame.

Transmit Polling Suspended

Transmit polling can be suspended by either of the following conditions:

- The DMA detects a descriptor owned by the Host (TDES0[31]=0). To resume, the driver must give descriptor ownership to the DMA and then issue a Poll Demand command.
- A frame transmission is aborted when a transmit error due to underflow is detected. The appropriate Transmit Descriptor 0 (TDES0) bit is set.

If the second condition occur, both Abnormal Interrupt Summary (DMA Register 5[15]) and Transmit Underflow bits (DMA Register 5 [5]) are set, and the information is written to Transmit Descriptor 0, causing the suspension. If the DMA goes into SUSPEND state due to the first condition, then both Normal Interrupt Summary (DMA Register 5 [16]) and Transmit Buffer Unavailable (DMA Register 5 [2]) are set.

In both cases, the position in the Transmit List is retained. The retained position is that of the descriptor following the Last Descriptor closed by the DMA.

The driver must explicitly issue a Transmit Poll Demand command after rectifying the suspension cause.

35.2.4.3 Reception

The Receive DMA engine's reception sequence is depicted in [Figure 35-9](#) and proceeds as follows:

1. The host sets up Receive descriptors (RDES0-RDES3) and sets the Own bit (RDES0[31]).

Ethernet MAC (ETH)

2. Once the SR (DMA Register 6[1]) bit is set, the DMA enters the Run state. While in the Run state, the DMA polls the Receive Descriptor list, attempting to acquire free descriptors. If the fetched descriptor is not free (is owned by the host), the DMA enters the Suspend state and jumps to Step 8.
3. The DMA decodes the receive data buffer address from the acquired descriptors.
4. Incoming frames are processed and placed in the acquired descriptor's data buffers.
5. When the buffer is full or the frame transfer is complete, the Receive engine fetches the next descriptor.
6. If the current frame transfer is complete, the DMA proceeds to Step 6. If the DMA does not own the next fetched descriptor and the frame transfer is not complete (EOF is not yet transferred), the DMA sets the Descriptor Error bit in the RDES0 (unless flushing is disabled). The DMA closes the current descriptor (clears the Own bit) and marks it as intermediate by clearing the Last Segment (LS) bit in the RDES0 value (marks it as Last Descriptor if flushing is not disabled), then proceeds to Step 7. If the DMA does own the next descriptor but the current frame transfer is not complete, the DMA closes the current descriptor as intermediate and reverts to Step 3.
7. If IEEE 1588 time stamping is enabled, the DMA writes the time stamp (if available) to the current descriptor's RDES2 and RDES3. It then takes the receive frame's status from the MTL and writes the status word to the current descriptor's RDES0, with the Own bit cleared and the Last Segment bit set.
8. The Receive engine checks the latest descriptor's Own bit. If the host owns the descriptor (Own bit is 1'b0) the Receive Buffer Unavailable bit (Register 5[7]) is set and the DMA Receive engine enters the Suspended state (Step 8). If the DMA owns the descriptor, the engine returns to Step 3 and awaits the next frame.
9. Before the Receive engine enters the Suspend state, partial frames are flushed from the Receive FIFO (You can control flushing using Bit 24 of DMA Register 6).
10. The Receive DMA exits the Suspend state when a Receive Poll demand is given or the start of next frame is available from the MTL's Receive FIFO. The engine proceeds to Step 1 and refetches the next descriptor.

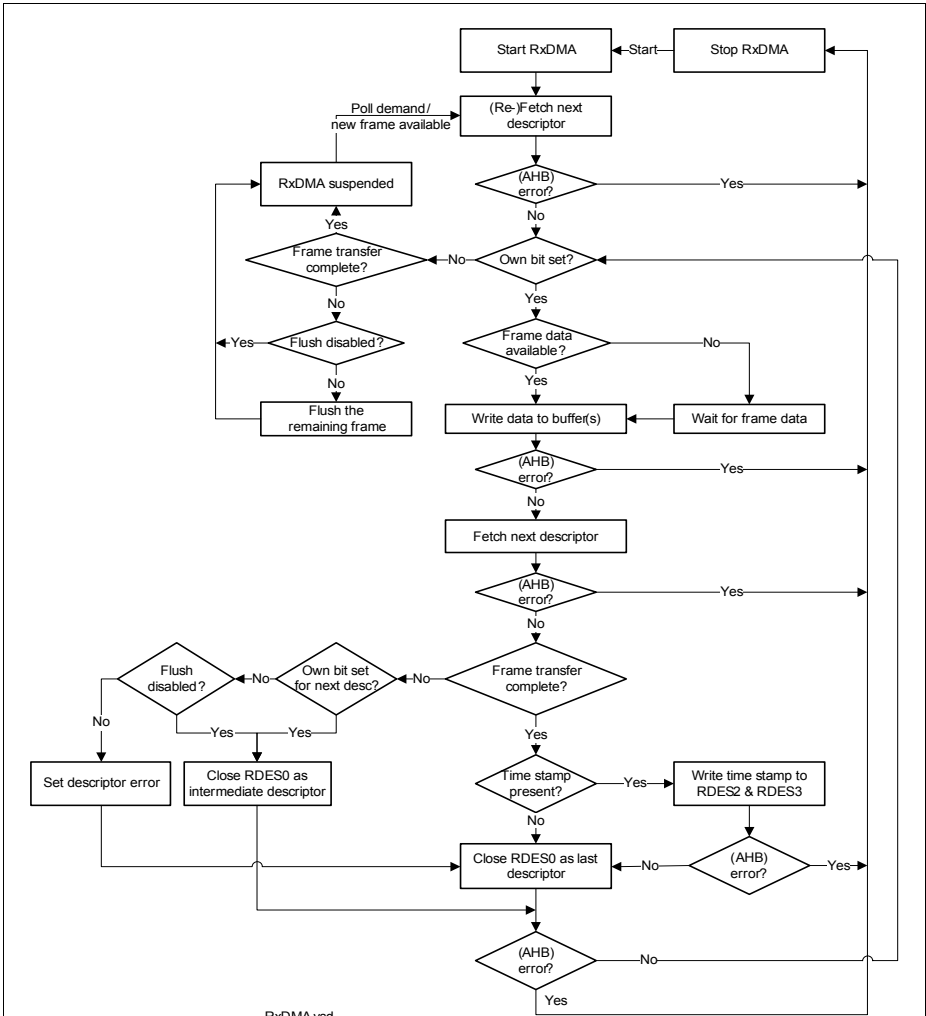


Figure 35-9 Receive DMA Operation

The DMA does not acknowledge accepting the status from the MTL until it has completed the time stamp write-back and is ready to perform status write-back to the descriptor.

If software has enabled time stamping through CSR, when a valid time stamp value is not available for the frame (for example, because the receive FIFO was full before the time stamp could be written to it), the DMA writes all-ones to RDES2 and RDES3.

Ethernet MAC (ETH)

Otherwise (that is, if time stamping is not enabled), the RDES2 and RDES3 remain unchanged.

Receive Descriptor Acquisition

The Receive Engine always attempts to acquire an extra descriptor in anticipation of an incoming frame. Descriptor acquisition is attempted if any of the following conditions is satisfied:

- The receive Start/Stop bit (Register 6[1]) has been set immediately after being placed in the Run state.
- The data buffer of current descriptor is full before the frame ends for the current transfer.
- The controller has completed frame reception, but the current Receive Descriptor is not yet closed.
- The receive process has been suspended because of a host-owned buffer (RDES0[31] = 0) and a new frame is received.
- A Receive poll demand has been issued.

Receive Frame Processing

The GMAC transfers the received frames to the Host memory only when the frame passes the address filter and frame size is greater than or equal to configurable threshold bytes set for the Receive FIFO of MTL, or when the complete frame is written to the FIFO in Store-and-Forward mode.

If the frame fails the address filtering, it is dropped in the GMAC block itself (unless Receive All GMAC Register 1[31] bit is set). Frames that are shorter than 64 bytes, because of collision or premature termination, can be purged from the MTL Receive FIFO.

After 64 (configurable threshold) bytes have been received, the MTL block requests the DMA block to begin transferring the frame data to the Receive Buffer pointed to by the current descriptor. The DMA sets First Descriptor (RDES0[9]) after the DMA Host Interface (AHB/AXI or MDC) becomes ready to receive a data transfer (if DMA is not fetching transmit data from the host), to delimit the frame. The descriptors are released when the Own (RDES[31]) bit is reset to 1'b0, either as the Data buffer fills up or as the last segment of the frame is transferred to the Receive buffer. If the frame is contained in a single descriptor, both Last Descriptor (RDES[8]) and First Descriptor (RDES[9]) are set.

The DMA fetches the next descriptor, sets the Last Descriptor (RDES[8]) bit, and releases the RDES0 status bits in the previous frame descriptor. Then the DMA sets Receive Interrupt (Register 5[6]). The same process repeats unless the DMA encounters a descriptor flagged as being owned by the host. If this occurs, the Receive Process sets Receive Buffer Unavailable (Register 5[7]) and then enters the Suspend state. The position in the receive list is retained.

Receive Process Suspended

If a new Receive frame arrives while the Receive Process is in Suspend state, the DMA refetches the current descriptor in the Host memory. If the descriptor is now owned by the DMA, the Receive Process re-enters the Run state and starts frame reception. If the descriptor is still owned by the host, by default, the DMA discards the current frame at the top of the MTL Rx FIFO and increments the missed frame counter. If more than one frame is stored in the MTL Rx FIFO, the process repeats.

The discarding or flushing of the frame at the top of the MTL Rx FIFO can be avoided by setting Operation Mode register bit 24 (DFF). In such conditions, the receive process sets the Receive Buffer Unavailable status and returns to the Suspend state.

35.2.4.4 Interrupts

Interrupts can be generated as a result of various events. DMA Register 5 contains all the bits that might cause an interrupt. DMA Register 7 contains an enable bit for each of the events that can cause an interrupt.

There are two groups of interrupts, Normal and Abnormal, as described in DMA Register 5. Interrupts are cleared by writing a 1'b1 to the corresponding bit position. When all the enabled interrupts within a group are cleared, the corresponding summary bit is cleared. When both the summary bits are cleared, the interrupt signal `sbd_intr_o` is deasserted. If the GMAC core is the cause for assertion of the interrupt, then any of the GLI, GMI, or GPI bits of DMA Register 5 will be set high.

Note: DMA Register 5 is the (interrupt) status register. The interrupt pin (`sbd_intr_o`) will be asserted due to any event in this status register only if the corresponding interrupt enable bit is set in DMA Register 7.

Interrupts are not queued and if the interrupt event occurs before the driver has responded to it, no additional interrupts are generated. For example, Receive Interrupt (DMA Register 5[6]) indicates that one or more frames was transferred to the Host buffer. The driver must scan all descriptors, from the last recorded position to the first one owned by the DMA.

An interrupt is generated only once for simultaneous, multiple events. The driver must scan DMA Register 5 for the cause of the interrupt. The interrupt is not generated again unless a new interrupting event occurs, after the driver has cleared the appropriate bit in DMA Register 5. For example, the controller generates a Receive interrupt (DMA Register 5[6]) and the driver begins reading DMA Register 5. Next, Receive Buffer Unavailable (DMA Register 5[7]) occurs. The driver clears the Receive interrupt. Even then, the `sbd_intr_o` signal is not deasserted, due to the active or pending Receive Buffer Unavailable interrupt.

An interrupt timer is given for flexible control of Receive Interrupt (DMA register 5[6]). When this Interrupt timer is programmed with a non-zero value, it will get activated as soon as the RxDMA completes a transfer of a received frame to system memory without

asserting the Receive Interrupt because it is not enabled in the corresponding Receive Descriptor (RDES1[31] in Table 7-3). When this timer runs out as per the programmed value, RI bit is set and the interrupt is asserted if the corresponding RI is enabled in DMA Register 7. This timer gets disabled before it runs out, when a frame is transferred to memory and the RI is set because it is enabled for that descriptor.

35.2.5 MAC Transaction Layer (MTL)

The MAC Transaction Layer provides FIFO memory to buffer and regulate the frames between the application system memory and the GMAC core. It also enables the data to be transferred between the application clock domain and the GMAC clock domains. The MTL layer has 2 data paths, namely the Transmit path and the Receive Path. The data path for both directions is 32-bit wide and operates with a simple FIFO protocol.

The GMAC-MTL communicates with the application side with the Application Transmit Interface (ATI), Application Receive Interface (ARI), and the MAC Control Interface (MCI).

35.2.5.1 Transmit Path

The DMA controls all transactions for the transmit path through the ATI. Ethernet frames read from the system memory is pushed into the FIFO by the DMA. The frame is then popped out and transferred to the GMAC core when triggered. When the end-of-frame is transferred, the status of the transmission is taken from the GMAC core and transferred back to the DMA.

The Transmit FIFO has a depth of 4K bytes. 4FIFO-fill level is indicated to the DMA so that it can initiate a data fetch in required bursts from the system memory, using the AHB interface. The data from the AHB Master interface is pushed into the FIFO with the appropriate byte lanes qualified by the DMA. The DMA also indicates the start-of-frame (SOF) and end-of-frame (EOF) transfers along with a few sideband signals controlling the pad-insertion/CRC generation for that frame in the GMAC core.

Per-frame control bits, such as Automatic Pad/CRC Stripping disable, time stamp capture, and so forth are taken as sideband control inputs on the ATI, stored in a separate register FIFO, and passed on to the core transmitter when the corresponding frame data is read from the Transmit FIFO.

There are two modes of operation for popping data towards the GMAC core. In Threshold mode, as soon as the number of bytes in the FIFO crosses the configured threshold level (or when the end-of-frame is written before the threshold is crossed), the data is ready to be popped out and forwarded to the GMAC core. The threshold level is configured using the TTC bits of DMA Register 0. In store-and-forward mode, the MTL pops the frame towards the GMAC core only when one or more of the following conditions are true:

- When a complete frame is stored in the FIFO

Ethernet MAC (ETH)

- When the TX FIFO becomes almost full
- When the ATI watermark becomes low. The watermark becomes low when the requested FIFO does not have space to accommodate the requested burst-length on the ATI.

Therefore, the MTL never stops in the store-and-forward mode even if the Ethernet frame length is bigger than the Tx FIFO depth.

The application can flush the Transmit FIFO of all contents by setting the FTF (DMA Register 6[20]) bit. This bit is self-clearing and initializes the FIFO pointers to the default state. If the FTF bit is set during a frame transfer from the MTL to the GMAC core, then the MTL stops further transfer as the FIFO is considered to be empty. Hence an underflow event occurs at the GMAC transmitter and the corresponding Status word is forwarded to the DMA.

Initialization through **Transmit Status Word** detail initialization and transmit operations for the MTL Layer.

Initialization

Upon reset, the MTL is ready to manage the flow of data to and from the DMA and the GMAC.

There are no requirements for enabling the MTL. However, the GMAC block and the DMA controller must be enabled individually through their respective CSRs.

Single-Packet Transmit Operation

During a transmit operation, the MTL block is slaved to the DMA controller. The general sequence of events for a transmit operation is as follows.

1. If the system has data to be transferred, the DMA controller, if enabled, fetches data from the Host through the AHB/AXI Master interface and starts forwarding it to the MTL. The MTL pushes the data received from the DMA into the FIFO. It continues to receive the data until the end-of frame of the frame is transferred.
2. The data is taken out of the FIFO and sent to the MAC by the FIFO controller engine. When the threshold level is crossed or a full packet of data is received into the FIFO, the MTL pops out the frame data and drives them to the GMAC core. The engine continues to transfer data from the FIFO until a complete packet has been transferred to the MAC. Upon completion of the frame, the MTL receives the Status from the GMAC and then notifies the DMA controller

Transmit Operation—Two Packets in the Buffer

1. Because the DMA must update the descriptor status before releasing it to the Host, there can be at the most two frames inside a transmit FIFO. The second frame will be fetched by the DMA and put into the FIFO only if the OSF (Operate on Second

Ethernet MAC (ETH)

Frame bit is set). If this bit is not set, the next frame will be fetched from the memory only after the MAC has completely processed the frame and the DMA has released the descriptors.

2. If the OSF bit is set, the DMA starts fetching the second frame immediately after completing the transfer of the first frame to the FIFO. It does not wait for the status to be updated. The MTL, in the meantime, receives the second frame into the FIFO while transmitting the first frame. As soon as the first frame has been transferred and the status is received from the MAC, the MTL pushes it to the DMA. If the DMA has already completed sending the second packet to the MTL, it must wait for the status of the first packet before proceeding to the next frame.

Transmit Operation—Multiple Packets in Buffer

In GMAC-MTL configuration, the transmit FIFO can be configured to accept more than 2 packets at a time. This option limits the number of status words that can be stored in the MTL before it is transferred to the DMA/host. By default, this number is limited to 2 but can be configured for 4 or 8 as well. Once the MTL FIFO accepts the number of frames equal to the status FIFO depth, it will stop accepting further frames unless the transmit Status that is given out and accepted by the host/DMA thus freeing up the space in this small FIFO.

Retransmission During Collision

While a frame is being transferred from the MTL to the GMAC, a collision event occurs on the GMAC line interface in Half-Duplex mode. The GMAC then indicates a retry attempt to the MTL by giving the status even before the end-of-frame is transferred from MTL. Then the MTL will enable the retransmission by popping out the frame again from the FIFO.

After more than 96 bytes are popped towards the GMAC core, the FIFO controller frees up that space and makes it available to the DMA to push in more data. This means that the retransmission is not possible after this threshold is crossed or when the GMAC core indicates a late-collision event.

Transmit FIFO Flush Operation

The GMAC provides a control to the software to flush the Transmit FIFO in the MTL layer through the use of Bit 20 of the Operation Mode register. The Flush operation is immediate and the MTL clears the Tx FIFO and the corresponding pointers to the initial state even if it is in the middle of transferring a frame to the GMAC Core. The data which is already accepted by the MAC transmitter will not be flushed. It will be scheduled for transmission and will result in underflow as Tx FIFO does not complete the transfer of rest of the frame. As in all underflow conditions, a runt frame will be transmitted and observed on the line. The status of such a frame will be marked with both Underflow and Frame Flush events (TDES0 bits 13 and 1).

Ethernet MAC (ETH)

The MTL layer also stops accepting any data from the application (DMA) during the Flush operation. It will generate and transfer Transmit Status Words to the application for the number of frames that is flushed inside the MTL (including partial frames). Frames that are completely flushed in the MTL will have the Frame Flush Status bit (TDES0 13) set. The MTL completes the Flush operation when the application (DMA) accepts all of the Status Words for the frames that were flushed, and then clears the Transmit FIFO Flush control register bit. At this point, the MTL starts accepting new frames from the application (DMA).

Transmit Status Word

At the end of transfer of the Ethernet frame to the GMAC core and after the core completes the transmission of the frame, the MTL outputs the transmit status to the application. The detailed description of the Transmit Status is the same as for bits [23:0] of TDES0, given in [Table 35-28](#).

If IEEE 1588 time stamping is enabled, the MTL returns specific frame's 64-bit time stamp, along with the ATI's transmit status.

Transmit Checksum Offload Engine

Communication protocols such as TCP and UDP implement checksum fields, which help determine the integrity of data transmitted over a network. Because the most widespread use of Ethernet is to encapsulate TCP and UDP over IP datagrams, the GMAC-UNIV has an Checksum Offload Engine (COE) to support checksum calculation and insertion in the transmit path, and error detection in the receive path. This section explains the operation of the Checksum Offload Engine for transmitted frames.

Note: The checksum for TCP, UDP, or ICMP is calculated over a complete frame, then inserted into its corresponding header field. Due to this requirement, this function is enabled only when the Transmit FIFO is configured for Store-and-Forward mode (that is, when the TSF bit is set in DMA Register 6). If the core is configured for Threshold (cut-through) mode, the Transmit COE is bypassed.

Note: You must make sure that the Transmit FIFO is deep enough to store a complete frame before that frame is transferred to the GMAC Core transmitter. The reason being that when space is not available to accept the programmed burst length of the data, then the MTL TxFIFO starts reading to avoid dead-lock. Once reading starts, then checksum insertion engine fails and consequently all succeeding frames may get corrupted due to improper recovery. Therefore, you must enable the checksum insertion only in the frames that are less than the following number of bytes in size (even in the store-and-forward mode):

FIFO Depth – PBL – 3 FIFO Locations

The PBL is the programmed burst-length.

Ethernet MAC (ETH)

This module supports two types of checksum calculation and insertion. This checksum engine can be controlled for each frame by setting the CIC bits (Bits 28:27 of TDES1, described in **Transmit Descriptor 1 (TDES1)**) in GMAC-AHB or GMAC-DMA configurations.

Note: See IETF specifications RFC 791, RFC 793, RFC 768, RFC 792, RFC 2460, and RFC 4443 for IPv4, TCP, UDP, ICMP, IPv6, and ICMPv6 packet header specifications, respectively.

IP Header Checksum Engine

In IPv4 datagrams, the integrity of the header fields is indicated by the 16-bit Header Checksum field (the eleventh and twelfth bytes of the IPv4 datagram). The COE detects an IPv4 datagram when the Ethernet frame's Type field has the value 0x0800 and the IP datagram's Version field has the value 0x4. The input frame's checksum field is ignored during calculation and replaced with the calculated value.

IPv6 headers do not have a checksum field; thus, the COE does not modify IPv6 header fields.

The result of this IP header checksum calculation is indicated by the IP Header Error status bit in the Transmit status (Bit 16 in **Table 35-28**). This status bit is set whenever the values of the Ethernet Type field and the IP header's Version field are not consistent, or when the Ethernet frame does not have enough data, as indicated by the IP header Length field.

In other words, this bit is set when an IP header error is asserted under the following circumstances:

For IPv4 datagrams

- The received Ethernet type is 0x0800, but the IP header's Version field does not equal 0x4
- The IPv4 Header Length field indicates a value less than 0x5 (20 bytes)
- The total frame length is less than the value given in the IPv4 Header Length field

For IPv6 datagrams

- The Ethernet type is 0x86dd but the IP header Version field does not equal 0x6
- The frame ends before the IPv6 header (40 bytes) or extension header (as given in the corresponding Header Length field in an extension header) is completely received.

Even when the COE detects such an IP header error, it inserts an IPv4 header checksum if the Ethernet Type field indicates an IPv4 payload.

TCP/UDP/ICMP Checksum Engine

Ethernet MAC (ETH)

The TCP/UDP/ICMP Checksum Engine processes the IPv4 or IPv6 header (including extension headers) and determines whether the encapsulated payload is TCP, UDP, or ICMP.

Note: For non-TCP, -UDP, or -ICMP/ICMPv6 payloads, this checksum engine is bypassed and nothing further is modified in the frame.

Note: Fragmented IP frames (IPv4 or IPv6), IP frames with security features (such as an authentication header or encapsulated security payload), and IPv6 frames with routing headers are not processed by this engine, and therefore must be bypassed. In other words, payload checksum insertion must not be enabled for such frames.

The checksum is calculated for the TCP, UDP, or ICMP payload and inserted into its corresponding field in the header. This engine can work in the following two modes:

- In the first mode, the TCP, UDP, or ICMPv6 pseudo-header is not included in the checksum calculation and is assumed to be present in the input frame's Checksum field. This engine includes the Checksum field in the checksum calculation, then replaces the Checksum field with the final calculated checksum.
- In the second mode, the engine ignores the Checksum field, includes the TCP, UDP, or ICMPv6 pseudo-header data into the checksum calculation, and overwrites the checksum field with the final calculated value.

Note: For ICMP-over-IPv4 packets, the Checksum field in the ICMP packet must always be 16'h0000 in both modes, because pseudo-headers are not defined for such packets. If it does not equal 16'h0000, an incorrect checksum may be inserted into the packet.

The result of this operation is indicated by the Payload Checksum Error status bit in the Transmit Status vector (Bit 12 in [Table 35-28](#)). This engine sets the Payload Checksum Error status bit when it detects that the frame has been forwarded to the MAC Transmitter engine in Store-and-Forward mode without the end-of-frame being written to the FIFO, or when the packet ends before the number of bytes indicated by the Payload Length field in the IP Header is received. When the packet is longer than the indicated payload length, the COE ignores them as stuff bytes, and no error is reported. When this engine detects the first type of error, it does not modify the TCP, UDP, or ICMP header. For the second error type, it still inserts the calculated checksum into the corresponding header field.

35.2.5.2 Receive Path

This module receives the frames given out by the GMAC core and pushes them into the Rx FIFO. The status (fill level) of this FIFO is indicated to the DMA once it crosses the configured Receive threshold (RTC of DMA Register 6). The MTL also indicates the FIFO fill level so that the DMA can initiate pre-configured burst transfers towards the AHB interface.

Receive Operation through **Receive Status Word** detail receive operations for the MTL Layer.

Receive Operation

During an Rx operation, the MTL is slaved to the GMAC. The general sequence of Receive operation events is as follows:

1. When the GMAC receives a frame, it pushes in data along with byte enables. The GMAC also indicates the SOF and EOF. The MTL accepts the data and pushes it into the Rx FIFO. After the EOF is transferred, the GMAC drives the status word, which is also pushed into the same Rx FIFO by the MTL.
2. When IEEE 1588 time stamping is enabled and the 64-bit time stamp is available along with the receive status, it is appended to the frame received from the GMAC and is pushed into the Rx FIFO before the corresponding receive status word is written. Thus, two additional locations per frame are taken for storing the time stamp in the Rx FIFO.
3. The MTL_RX engine takes the data out of the FIFO and sends it to the DMA. In the default Cut-Through mode, when 64 bytes (configured with RTC bits of DMA Register 6) or a full packet of data are received into the FIFO, the MTL_RX engine pops out the data and indicates its availability to the DMA. Once the DMA initiates the transfer to the AHBinterface, the MTL_RX engine continues to transfer data from the FIFO until a complete packet has been transferred. Upon completion of the EOF frame transfer, the MTL pops out the status word and sends it to the DMA controller.
4. In Rx FIFO Store-and-Forward mode (configured by the RSF bit of DMA Register 6), a frame is read out only after being written completely into the Receive FIFO. In this mode, all error frames are dropped (if the core is configured to do so) such that only valid frames are read out and forwarded to the application. In Cut-Through mode, some error frames are not dropped, because the error status is received at the end-of-frame, by which time the start of that frame has already been read out of the FIFO.

Note: In 32-bit data bus mode, the time-stamp transfer takes two clock cycles and the lower 32-bit of the time-stamp is given out first. The status also may be extended to two cycles when Advanced Time-stamp feature is enabled.

Receive Operation Multiframe Handling

Since the status is available immediately following the data, the MTL is capable of storing any number of frames into the FIFO, as long as it is not full.

GMAC Flow Control

The flow control operation of the GMAC can also be enabled (EFC bit of DMA Register 6) by the Rx FIFO control logic. The flow control signal to the GMAC is asserted whenever the Rx FIFO fill level crosses the configured threshold (RFA bits of DMA Register 6). This flow control signal is deasserted once the FIFO fill-level falls below the

Ethernet MAC (ETH)

configured threshold (RFD bits). This operation is independent of whether the GMAC is configured for full-duplex or half-duplex.

The above hardware flow control operation is applicable only when the Rx FIFO is 4.096 or more bytes deep. A separate sideband flow control signal (sbd_flowctrl_i) is optionally provided at the top-level I/O for all GMAC-AHB, GMAC-AXI, GMAC-DMA and GMAC-MTL configurations. For 4.096-byte or larger Rx FIFOs, this sideband signal is logically ORed with the hardware flow control signal. Thus, flow control is enabled when either of these signals is asserted and disabled when both these signals are deasserted.

Error Handling

If the MTL Rx FIFO is full before it receives the EOF data from the GMAC, an overflow is declared, the whole frame (including the status word) is dropped, and the overflow counter in the DMA (Register 8) is incremented. This is true even if the Forward Error Frame (FEF bit of DMA Register 6) is set. If the start address of such a frame has already been transferred to the Read Controller, the rest of the frame is dropped and a dummy EOF is written to the FIFO along with the status word. The status will indicate a partial frame due to overflow. In such frames, the Frame Length field is invalid.

The MTL Rx Control logic can filter error and undersized frames, if enabled (using the DMA Register 6 FEF and FUF bits). If the start address of such a frame has already been transferred to the Rx FIFO Read Controller, that frame is not filtered. The start address of the frame is transferred to the Read Controller after the frame crosses the receive threshold (set by the DMA Register 6 RTC bits).

If the MTL Receive FIFO is configured to operate in Store-and-Forward mode, all error frames can be filtered and dropped.

Receive Status Word

At the end of the transfer of the Ethernet frame to the host, the MTL outputs the receive status to the Application. The detailed description of the receive status is the same as for Bits[31:0] of RDES0, given in [Table 35-23](#), except that Bits 31, 14, 9, and 8 are reserved and have a reset of 1'b0 by default. When the status of a partial frame due to overflow is given out, the Frame Length field in the status word is not valid.

Note: When Advanced Time Stamp feature is enabled, the status is composed of two parts - normal (default [31:0]), and extended. The extended status[63:32] gives the information about the received ethernet payload when it is carrying PTP packets or TCP/UDP/ICMP over IP packets. In 32-bit data-bus, these are transferred over two clock cycles. The detailed description of the receive status is the same as described in RDES0 and RDES4 in [Receive Descriptor](#), except that bits 31, 14, 9, and 8 of normal status is reserved and have a reset value of 1'b0. When the status of a partial frame due to overflow is given out, the Frame Length field in the status word is not valid.

35.2.6 GMAC Core

The GMAC core supports many interfaces towards the PHY chip. In TC26x B-Step RMII is implemented. The PHY interface can be selected only once after reset. The GMAC core communicates with the application side with the MAC Transmit Interface (MTI), MAC Receive Interface (MRI) and the MAC Control Interface (MCI).

35.2.6.1 Transmission

Transmission is initiated when the MTL Application pushes in data with the SOF . When the SOF signal is detected, the GMAC accepts the data and begins transmitting to the MII. The time required to transmit the frame data to the RMII/MII after the Application initiates transmission is variable, depending on delay factors like IFG delay, time to transmit preamble/SFD, and any back-off delays for Half-Duplex mode. Until then, the GMAC does not accept the data received from MTL.

After the EOF is transferred to the GMAC Core, the core complete normal transmission and then gives the Status of Transmission back to the MTL. If a normal collision (in Half-duplex mode) occurs during transmission, the GMAC core makes valid the Transmit Status to the MTL. It then accepts and drops all further data until the next SOF is received. The MTL block should retransmit the same frame from SOF on observing a Retry request (in the Status) from the GMAC.

The GMAC issues an underflow status if the MTL is not able to provide the data continuously during the transmission. During the normal transfer of a frame from MTL, if the GMAC receives a SOF without getting an EOF for the previous frame, then it (the SOF) is ignored and the new frame is considered as continuation of the previous frame.

The following six modules constitute the transmission function of the GMAC:

- Transmit Bus Interface Module (TBU)
- Transmit Frame Controller Module (TFC)
- Transmit Protocol Engine Module (TPE)
- Transmit Scheduler Module (STX)
- Transmit CRC Generator Module (CTX)
- Transmit Flow Control Module (FTX)

Transmit Bus Interface Module

This module interfaces the transmit path of the GMAC core with the external frame with a FIFO interface.

This module also outputs the (32-bit) Transmit Status to the application at the end of normal transmission or collision.

Additionally, this module outputs the Transmit Snapshot register value.

Transmit Frame Controller Module

The Transmit Frame Controller (TFC) consists of two registers to hold data, byte enables, and the last data control received from the TBU. The register provides a buffer between the Application and the TPE to regulate data flow as well as converts the input data into an 8-bit bus towards the TPE.

When the number of bytes received from the Application falls below 60 (DA+SA+LT+DATA), the state machine that interfaces with the TBU automatically appends zeros to the transmitting frame to make the data length exactly 46 bytes to meet the minimum data field requirement of IEEE 802.3. The GMAC can be programmed not to append any padding.

The cyclic redundancy check (CRC) for the Frame Check Sequence (FCS) field is calculated before transmission to the TPE module. This value is computed by CTX module. The TFC module receives the computed CRC and appends it to the data being transmitted to the TPE module. When the GMAC is programmed to not append the CRC value to the end of Ethernet frames, the TFC module ignores the computed CRC and transmits only the data received from the TBU module to the TPE module. An exception to this rule is that when the GMAC is programmed to append pads for frames (DA+SA+LT+DATA) less than 60 bytes sent by the TBU module, the TFC module will append the CRC at the end of padded frame.

The TFC converts the data received on the 32/64/128-bit interface from the TBU into 8-bit data for the TPE module.

Transmit Protocol Engine Module

The Transmit Protocol Engine (TPE) module consists of a transmit state machine that controls the operation of Ethernet frame transmission. The module's transmit state machine performs the following functions to meet the IEEE 802.3 specifications.

- Generates preamble and SFD
- Generates jam pattern in Half-Duplex mode
- Jabber timeout
- Flow control for Half-Duplex mode (back pressure)
- Generates transmit frame status
- Contains time stamp snapshot logic for IEEE 1588 support

When a new frame transmission from the TFC is requested, the transmit state machine sends out the preamble and SFD, followed by the data received. The preamble is defined as 7 bytes of 8'b10101010 pattern, and the SFD is defined as 1 byte of 8b'10101011 pattern.

The collision window is defined as 1 slot time (512 bit times for 10/100 Mbit/s Ethernet). The jam pattern generation is applicable only to Half-Duplex mode, not to Full-Duplex mode. In Full-Duplex mode, the transmit state machine ignores the phy_col_i signal from the PHY.

Ethernet MAC (ETH)

In MII mode, if a collision occurs any time from the beginning of the frame to the end of the CRC field, the transmit state machine sends a 32-bit jam pattern of 32'h55555555 on the MII to inform all other stations that a collision has occurred. If the collision is seen during the preamble transmission phase, the transmit state machine completes the transmission of preamble and SFD and then sends the jam pattern.

If the collision occurs after the collision window and before the end of the FCS field (or the end of Burst if the Frame Burst mode is enabled), the transmit state machine sends a 32-bit jam pattern and sets the late collision bit in the transmit frame status.

The TPE module maintains a jabber timer (only in 10/100-Mbit/s mode) to cut off the transmission of Ethernet frames if the TFC module transfers more than 2.048 (default) bytes. The time-out is changed to 10.240 bytes when the Jumbo frame is enabled.

The Transmit state machine uses the deferral mechanism for the flow control (Back Pressure) in Half-Duplex mode. When the Application requests to stop receiving frames, the Transmit state machine sends a JAM pattern of 32 bytes whenever it senses a reception of a frame, provided the transmit flow control is enabled. This will result in a collision and the remote station will back off. The Application requests the flow control by setting BPA bit of Register6. If the application requests a frame to be transmitted, then it will be scheduled and transmitted even when the backpressure is activated. Note that if the backpressure is kept activated for a long time (and more than 16 consecutive collision events occur) then the remote stations will abort their transmissions due to excessive collisions.

If IEEE 1588 time stamping is enabled for the transmit frame, this block takes a snapshot of the system time when the SFD is put onto the transmit MII bus. The system time source is either an external input or internally generated, according to the configuration selected.

Transmit Scheduler Module

The Transmit Scheduler (STX) module is responsible for scheduling the frame transmission on the MII. The two major functions of this module are to maintain the inter-frame gap between two transmitted frames and to follow the Truncated Binary Exponential Back-off algorithm for Half-Duplex mode. This module provides an enable signal to the TPE module after satisfying the IFG and Back-off delays.

The STX module maintains an idle period of the configured inter-frame gap (IFG bits of Register 0 between any two transmitted frames. If frames from the TFC arrive at the TPE module sooner than the configured IFG time, the TPE module waits for the enable signal from the STX module before starting the transmission on the MII. The STX module starts its IFG counter as soon as the carrier signal of the MII goes inactive. At the end of programmed IFG value, the module issues an enable signal to the TPE module in Full-Duplex mode. In Half-Duplex mode and when IFG is configured for 96 bit times, the STX module follows the rule of deference specified in Section 4.2.3.2.1 of the IEEE 802.3

Ethernet MAC (ETH)

specification. The module resets its IFG counter if a carrier is detected during the first two-thirds (64-bit times for all IFG values) of the IFG interval. If the carrier is detected during the final one third of the IFG interval, the STX module continues the IFG count and enables the transmitter after the IFG interval.

The STX module implements the Truncated Binary Exponential Back-off algorithm when it operates in Half-Duplex mode.

Transmit CRC Generator Module

The Transmit CRC Generator (CTX) module interfaces with the TFC module to generate CRC for the FCS field of the Ethernet frame. The TFC module sends the frame data and any necessary padding to the CTX module through an 8-bit interface.

This module calculates the 32-bit CRC for the FCS field of the Ethernet frame. The encoding is defined by the following generating polynomial.

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

The module gets the Ethernet frame's byte data from the TFC module (DA + SA + LT + DATA + PAD) qualified with a Data Valid signal. The TFC also indicates to the CTX when to reset the previously calculated CRC and to start the new CRC calculation for the coming frame. The TFC module issues the start command before sending the new frame data for calculation. The calculated CRC is valid on the next clock after the data is received.

Transmit Flow Control Module

The Transmit Flow Control (FTX) module generates Pause frames and transmit them to the TFC module as necessary, in Full-Duplex mode. The TFC module receives the Pause frame from the FTX module, appends the calculated CRC, and sends the frame to the TPE module. Pause frame generation can be initiated in two ways. The Application can request the FTX module to send a Pause frame either by setting the FCB bit in the Flow Control register Register 6 or by asserting the `mti_flowctrl_i` signal in response to the receive FIFO full conditions (packet buffer).

If the Application has requested the flow control by setting the FCB bit of Register 6, the FTX module will generate and transmit a single Pause frame to the TFC module. The value of the Pause Time in the generated frame contains the programmed Pause Time value in Register 6. To extend the pause or end the pause prior to the time specified in the previously transmitted Pause frame, the application must request another Pause frame transmission after programming the Pause Time register with appropriate value.

If the Application has requested the flow control by asserting the `mti_flowctrl_i` signal, the FTX module will generate and transmit a Pause frame to the TFC module. The value of the Pause Time in the generated frame contains the programmed Pause Time value in the Register 6. The FTX module monitors the `mti_flowctrl_i` signal. If it remains asserted at a configurable number of slot-times (PLT bits of GMAC Register 6) before this Pause-

Ethernet MAC (ETH)

time runs-out, a second Pause frame will be transmitted to the TFC module. The process will be repeated as long as the `mti_flowctrl` signal remains asserted.

If the `mti_flowctrl_i` signal goes inactive prior to the sampling time, the FTX module will transmit a Pause frame with zero Pause Time to indicate to the remote end that the receive buffer is ready to receive new data frames.

35.2.6.2 MAC Transmit Interface Protocol

The MAC Transmit Interface (MTI) connects the application (MTL module in the GMAC) with the GMAC to provide the Ethernet data for transmission.

The application initiates the Ethernet frame transmission by writing the first data of the frame to the GMAC, provided the GMAC is ready to accept data. The Application can push-in data as long as the GMAC core is ready to accept it.

If the frame transmission is not successful (due to underflow, collision, jabber timeout, excessive deferral events), the GMAC core will assert the transmit status even before the EOF is received. The Application will have to take the appropriate action as per the status. The GMAC will drop all further data input to it until the next SOF.

35.2.6.3 Reception

A receive operation is initiated when the GMAC detects an SFD on the MII. The core strips the preamble and SFD before proceeding to process the frame. The header fields are checked for the filtering and the FCS field used to verify the CRC for the frame. The received frame is stored in a shallow buffer until the address filtering is performed. The frame is dropped in the core if it fails the address filter.

The following are the functional blocks in the Receive path of the GMAC core.

- Receive Protocol Engine Module (RPE)
- Receive CRC Module (CRX)
- Receive Frame Controller Module (RFC)
- Receive Flow Control Module (FRX)
- Receive IP Checksum checker (IPC)
- Receive Bus Interface Unit Module (RBU)
- Address Filtering Module (AFM)

Receive Protocol Engine Module

The RPE consists of the receive state machine which strips the preamble and SFD. Once the `phy_rxdv_i` signal of the MII becomes active, the RPE's receive state machine begins hunting for the SFD field from the receive modifier logic. Until then, the state machine drops the receiving preambles. Once the SFD is detected, the state machine begins sending the data of the Ethernet frame to the RFC module, beginning with the first byte following the SFD (destination address).

Ethernet MAC (ETH)

If IEEE 1588 time stamping is enabled, the RPE takes a snapshot of the system time when any frame's SFD is detected on the MII. Unless the MAC filters out and drops the frame, this time stamp is passed on to the application.

In MII mode, the RPE converts the received nibble data into bytes, then forwards the valid frame data to the RFC module

The receive state machine of the RPE module decodes the Length/Type field of the receiving Ethernet frame. If the Length/Type field is less than 600 (hex) and if the MAC is programmed for the auto crc/pad stripping option, the state machine sends the data of the frame up to the count specified in the Length/Type field, then starts dropping bytes (including the FCS field). The state machine of the RPE module decodes the Length/Type field and checks for the Length interpretation.

If the Length/Type field is greater than or equal to 600 (hex), the RPE module will send all received Ethernet frame data to the RFC module, irrespective of the value on the programmed auto-CRC strip option.

As a default, the GMAC is programmed for watchdog timer to be enabled, that is, frames above 2.048 (10.240 if Jumbo Frame is enabled) bytes (DA + SA + LT + DATA + PAD + FCS) are cut off at the RPE module. This feature can be disabled by programming the GMAC Configuration register, Watchdog Disable. However even if the watchdog timer is disabled, frames greater than 16 KB in size are cut off and a watchdog time-out status is given.

The GMAC supports loopback of transmitted frames onto its receiver. As a default, the GMAC loopback function is disabled, but this feature can be enabled by programming the GMAC Configuration register, Loopback bit. The transmit and receive clocks can have an asynchronous timing relationship, so an asynchronous FIFO is used to make the loopback path of the phy_txd_o data onto the receive path. The asynchronous FIFO is 10 bits (6 bits in 10/100 Mbit/s mode) wide to accommodate phy_txd_o, phy_txen_o, and phy_txer_o. The FIFO is nine deep in 10/100 Mbit/s mode and free-running to write on the write clock (clk_tx_i) and read on every read clock (clk_rx_i).

The write and read pointers get re-initialized to have an offset of 2 (4 in 10/100 Mbit/s mode) at the start of each frame read out of the FIFO. This helps to avoid overflow/underflow during the transfer of a frame, and ensures that the overflow/underflow occurs only during the IFG period between the frames. Please note that the FIFO depth of nine is sufficient to prevent data corruption for frame sizes up to 9.022 bytes with a difference of 200 ppm between the MII Transmit and Receive clock frequencies. Hence, bigger frames should not be looped back, as they may get corrupted in this loopback FIFO.

At the end of every received frame, the RPE module generates received frame status and sends it to the RFC module. Control, missed frame, and filter fail status are added to the receive status in the RFC module.

Receive CRC Module

The Receive CRC (CRX) interfaces to the RPE module to check for any CRC error in the receiving frame.

This module calculates the 32-bit CRC for the received frame that includes the Destination address field through the FCS field. The encoding is defined by the following generating polynomial.

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

The module gets the data from the RPE module (DA+SA+LT+DATA+PAD+FCS). The RPE module also sends a control signal that indicates the validity of the data. Irrespective of the auto pad/CRC strip, the CRX module receives the entire frame to compute the CRC check for received frame. As a note on the auto pad/CRC strip settings, the entire frame is not transferred between the RPE and RFC 8-bit interface.

Receive Checksum Offload Engine

The Checksum Offload engine is optional (not available in the default configuration) and selectable during configuration. Two types of Receive Checksum Offload engine are available for configuration. Full Checksum Offload, the Type 2 engine is instantiated.

Type 2

In this mode, both IPv4 and IPv6 frames in the received Ethernet frames are detected and processed for data integrity. You can enable this module by setting the IPC bit in the GMAC Configuration register. The GMAC receiver identifies IPv4 or IPv6 frames by checking for value 0x0800 or 0x86DD, respectively, in the received Ethernet frames' Type field. This identification applies to VLAN-tagged frames as well.

The Receive Checksum Offload engine calculates IPv4 header checksums and checks that they match the received IPv4 header checksums. The result of this operation (pass or fail) is given to the RFC module for insertion into the receive status word. The IP Header Error bit is set for any mismatch between the indicated payload type (Ethernet Type field) and the IP header version, or when the received frame does not have enough bytes, as indicated by the IPv4 header's Length field (or when fewer than 20 bytes are available in an IPv4 or IPv6 header).

This engine also identifies a TCP, UDP or ICMP payload in the received IP datagrams (IPv4 or IPv6) and calculates the checksum of such payloads properly, as defined in the TCP, UDP, or ICMP specifications. This engine includes the TCP/UDP/ICMPv6 pseudo-header bytes for checksum calculation and checks whether the received checksum field matches the calculated value. The result of this operation is given as a Payload Checksum Error bit in the receive status word. This status bit is also set if the length of the TCP, UDP, or ICMP payload does not tally to the expected payload length given in the IP header.

Ethernet MAC (ETH)

As mentioned in [TCP/UDP/ICMP Checksum Engine](#), this engine bypasses the payload of fragmented IP datagrams, IP datagrams with security features, IPv6 routing headers, and payloads other than TCP, UDP or ICMP.

In this configuration, the core does not append any payload checksum bytes to the received Ethernet frames.

Receive Frame Controller Module

The Receive Frame Controller (RFC) receives the Ethernet frame data and status from the RPE module. The RFC module consists of a FIFO of parameterized depth (default set to 4 deep and 37 bits wide) and two state machines for writing and reading the FIFO. The FIFO holds the received Ethernet frame data and byte enables, along with a control bit to indicate the last data. The state machines manage the FIFO and provide a frame buffering for the receiving Ethernet frame from the RPE module. The main functions of the RFC module are:

- Data path conversion, which converts the 8-bit data to 32-bit data to the RBU module.
- Frame filtering
- Attaching the calculated IP Checksum input from IPC.
- Update the Receive Status and forward to RBU.

If the RA bit of the GMAC CSR Frame Filter register is set, the RFC module initiates the data transfer to the RBU module as soon as 4 bytes of Ethernet data are received from the RPE module. At the end of the data transfer, the RFC module sends out the received frame status that includes the frame filter bits (SA Filterfail and DAFilterfail) and status from the RFC module. These bits are generated based on the filter-fail signals from the AFM module. This status bit indicates to the Application whether the received frame has passed the filter controls (both address filter and Frame Filter controls from CSR). The RFC module will not drop any frame on its own in this mode.

If the RA bit is reset, the RFC module performs frame filtering based on the destination/source address (the Application still needs to perform another level of filtering if it decides not to receive any bad frames like runt, CRC error frames, etc. The RFC module waits to receive the first 14 bytes of received data (type field) from the RPE module. Until then, the module will not initiate any transfers to the RBU module. After receiving the destination/source address bytes, the RFC checks the filter-fail signal from the AFM module for an address match. On detecting a filter-fail from AFB, the frame is dropped at the RFC module and not transferred to the Application.

On a delayed filter response from the AFM (this can only occur if you change the AFM logic), the RFC module waits until the FIFO is full, and then proceeds with the frame transfer to the RBU module. However, it will still take the delayed response from the AFM module and if it is a (DA/SA) filter failure, then it will drop the rest of the frame and send the Rx Status Word (with zero frame-length, CRC Error and Runt Error bits set) immediately indicating the filter-fail. If there is no response from the AFM until the end of frame is transmitted, the filter fail status in the Rx Status Word is updated accordingly.

Ethernet MAC (ETH)

When the PMT module is configured for power-down mode, all received frames are dropped by this block, and are not forwarded to the application.

Receive Flow Control Module

The Receive Flow Controller (FRX) detects the receiving Pause frame and pauses the frame transmission for the delay specified within the received Pause frame. The FRX module is enabled only in Full-Duplex mode. The Pause frame detection function can be enabled or disabled with the RFE bit of GMAC CSR Register 6.

Once the receive flow control is enabled, the FRX module begins monitoring the received frame destination address for any match with the multicast address of the control frame (48'h0180C200001). If a match is detected, the FRX module indicates to the RFC module, that the destination address of the received frame matches the reserved control frame destination address. The RFC module then decides whether or not to transfer the received control frame to the Application, based on the (PCF) bit setting of GMAC CSR Register 1 (Filter register).

The FRX module also decodes the Type, Op-code, and Pause Timer field of the receiving control frame. At the end of received frame, the FRX module gets the received frame status from RPE. If the byte count of the status indicates 64 bytes, and if there is no CRC error, the FRX module requests the MAC transmitter to pause the transmission of any data frame for the duration of the decoded Pause Time value, multiplied by the slot time (64 byte times). Meanwhile, if another Pause frame is detected with a zero Pause Time value, the FRX module resets the Pause Time and gives another pause request to the Transmitter. If the received control frame matches neither the Type field (16'h8808), Opcode (16'h00001), nor byte length (64 bytes), or if there is a CRC error, the FRX module does not generate a Pause request to Transmitter.

In the case of a pause frame with a multicast destination address, the RFC filters the frame based on the address match from the FRX module. For a pause frame with a unicast destination address, the filtering in the FRX module depends on whether the DA matched the contents of the MAC Address Register 0 and the UP Bit of GMAC Core Register 6 is set (detecting a pause frame even with a unicast destination address). The PCF register bits (Bit [7:6] of GMAC Register 1) controls the filtering for control frames in addition to the Address filter module.

Receive Bus Interface Unit Module

The Receive Bus Interface Unit (RBU) converts the 32-bit data received from the RFC module into a 32-bit FIFO protocol on the Application side. The RBU module interfaces with the Application through the MAC receive interface (MRI).

If IEEE 1588 time stamping is enabled, the RBU also outputs the time stamp captured from the received frame.

Address Filtering Module

The Address Filtering (AFM) module performs the destination and source address checking function on all received frames and reports the address filtering status to the RFC module. The address checking is based on different parameters (Frame Filter register) chosen by the Application. These parameters are inputs to the AFM module as control signals, and the AFM module reports the status of the address filtering based on the combination of these inputs. The AFM module does not filter the receive frames by itself, but reports the status of the address filtering (whether to drop the frame or not) to the RFC module. The AFM module also reports whether the receiving frame is a multicast frame or a broadcast frame, as well as the address filter status.

The AFM module probes the 8-bit receive data path between the RPE module and the RFC module and checks the destination and source address field of each incoming packet. In Mill mode the module takes 14/26 clocks (from the start of frame) to compare the destination/ source address of the receiving frame. The AFM module gets the station's physical (MAC) address and the Multicast Hash table from CSR module for address checking. The CSR module provides the Frame Filter register parameters to AFM.

Unicast Destination Address Filter

The AFM supports up to 32 MAC addresses for unicast perfect filtering. If perfect filtering is selected (HUC bit of Frame Filter register is reset), the AFM compares all 48 bits of the received unicast address with the programmed MAC address for any match. Default MacAddr0 is always enabled, other addresses MacAddr1–MacAddr31 are selected with an individual enable bit. Each byte of these other addresses (MacAddr1–MacAddr31) can be masked during comparison with the corresponding received DA byte by setting the corresponding Mask Byte Control bit in the register. This helps group address filtering for the DA.

In Hash filtering mode (When HUC bit is set), the AFM performs imperfect filtering for unicast addresses using a 64-bit Hash table. For hash filtering, the AFM uses the upper 6 bits CRC of the received destination address to index the content of the Hash table. A value of 000000 selects Bit 0 of the selected register, and a value of 111111 selects Bit 63 of the Hash Table register. If the corresponding bit (indicated by the 6-bit CRC) is set to 1, the unicast frame is said to have passed the Hash filter; otherwise, the frame has failed the Hash filter.

Multicast Destination Address Filter

The GMAC can be programmed to pass all multicast frames by setting the PM bit in the Frame Filter register. If the PM bit is reset, the AFM performs the filtering for multicast addresses based on the HMC bit of Frame Filter register. In Perfect Filtering mode, the multicast address is compared with the programmed MAC Destination Address registers (1–31). Group address filtering is also supported.

Ethernet MAC (ETH)

In Hash filtering mode, the AFM performs imperfect filtering using a 64-bit Hash table. For hash filtering, the AFM uses the upper 6 bits CRC of the received multicast address to index the content of the Hash table. A value of 000000 selects Bit 0 of the selected register and a value of 111111 selects Bit 63 of the Hash Table register.

If the corresponding bit is set to 1, then the multicast frame is said to have passed the Hash filter; otherwise, the frame has failed the Hash filter.

Hash or Perfect Address Filter

The DA filter can be configured to pass a frame when its DA matches either the Hash filter or the Perfect filter by setting the HPF bit of the Frame Filter register and setting the corresponding HUC or HMC bits. This configuration applies to both unicast and multicast frames. If the HPF bit is reset, only one of the filters (Hash or Perfect) is applied to the received frame.

Broadcast Address Filter

The AFM doesn't filter any broadcast frames in the default mode. However, if the GMAC is programmed to reject all broadcast frames by setting the DBF bit in the Frame Filter register, the DAF module asserts the Filter fail signal to RFC, whenever a broadcast frame is received. This will tell the RFC module to drop the frame.

Unicast Source Address Filter

The GMAC can also perform a perfect filtering based on the source address field of the received frames. By default, the AFM compares the SA field with the values programmed in the SA registers. The MAC Address registers [1:31] can be configured to contain SA instead of DA for comparison, by setting Bit 30 of the corresponding Register. Group filtering with SA is also supported. The frames that fail the SA Filter are dropped by the GMAC if the SAF bit of Frame Filter register is set.

When SAF bit is set, the result of SA Filter and DA filter is AND'ed to decide whether the frame needs to be forwarded. This means that either of the filter fail result will drop the frame and both filters have to pass in-order to forward the frame to the application.

Inverse Filtering Operation

For both Destination and Source address filtering, there is an option to invert the filter-match result at the final output. These are controlled by the DAIF and SAIF bits of the Frame Filter register respectively. The DAIF bit is applicable for both Unicast and Multicast DA frames. The result of the unicast/multicast destination address filter is inverted in this mode. Similarly, when the SAIF bit is set, the result of unicast SA filter is reversed.

Table 35-8 and **Table 35-9** summarize the Destination and Source Address filtering based on the type of frames received.

Table 35-8 Destination Address Filtering Table

Frame Type	PR	HPF	HUC	DAIF	HMC	PM	DB	DA Filter Operation
Broadcast	1	X	X	X	X	X	X	Pass
	0	X	X	X	X	X	0	Pass
	0	X	X	X	X	X	1	Fail
Unicast	1	X	X	X	X	X	X	Pass all frames.
	0	X	0	0	X	X	X	Pass on Perfect/Group filter match.
	0	X	0	1	X	X	X	Fail on Perfect/Group filter match.
	0	0	1	0	X	X	X	Pass on Hash filter match.
	0	0	1	1	X	X	X	Fail on Hash filter match.
	0	1	1	0	X	X	X	Pass on Hash or Perfect/Group filter match.
	0	1	1	1	X	X	X	Fail on Hash or Perfect/Group filter match.
Multicast	1	X	X	X	X	X	X	Pass all frames.
	X	X	X	X	X	1	X	Pass all frames.
	0	X	X	0	0	0	X	Pass on Perfect/Group filter match and drop PAUSE control frames if PCF = 0x.
	0	0	X	0	1	0	X	Pass on Hash filter match and drop PAUSE control frames if PCF = 0x.
	0	1	X	0	1	0	X	Pass on Hash or Perfect/Group filter match and drop PAUSE control frames if PCF = 0x.
	0	X	X	1	0	0	X	Fail on Perfect/Group filter match and drop PAUSE control frames if PCF = 0x.
	0	X	X	1	0	0	X	Fail on Perfect/Group filter match and drop PAUSE control frames if PCF = 0x.

Table 35-8 Destination Address Filtering Table (cont'd)

Frame Type	PR	HPF	HUC	DAIF	HMC	PM	DB	DA Filter Operation
	0	0	X	1	1	0	X	Fail on Hash filter match and drop PAUSE control frames if PCF = 0x.
	0	1	X	1	1	0	X	Fail on Hash or Perfect/Group filter match and drop PAUSE control frames if PCF = 0x.

Table 35-9 Source Address Filtering Table

Frame Type	PR	SAIF	SAF	SA Filter Operation
Unicast	1	X	X	Pass all frames.
	0	0	0	Pass status on Perfect/Group filter match but do not drop frames that fail.
	0	1	0	Fail status on Perfect/Group filter match but do not drop frame.
	0	0	1	Pass on Perfect/Group filter match and drop frames that fail.
	0	1	1	Fail on Perfect/Group filter match and drop frames that fail.

35.2.6.4 System Time Register Module

64-bit time is maintained in this module and updated using the input reference clock. This time is the source for taking snapshots (time stamps) of Ethernet frames being transmitted or received at the MII.

The System Time counter can be initialized or corrected using the coarse correction method. In this method, the initial value or the offset value is written to the Time Stamp Update register. For initialization, the System Time counter is written with the value in the Time Stamp Update registers, while for system time correction, the offset value is added to or subtracted from the system time.

In the fine correction method, a slave clock's frequency drift with respect to the master clock (as defined in IEEE 1588) is corrected over a period of time instead of in one clock, as in coarse correction. This helps maintain linear time and does not introduce drastic changes (or a large jitter) in the reference time between PTP Sync message intervals. In this method, an accumulator sums up the contents of the Addend register, as shown in

Figure 35-10. The arithmetic carry that the accumulator generates is used as a pulse to increment the system time counter. The accumulator and the addend are 32-bit registers. Here, the accumulator acts as a high-precision frequency multiplier or divider. This algorithm is depicted in **Figure 35-10**:

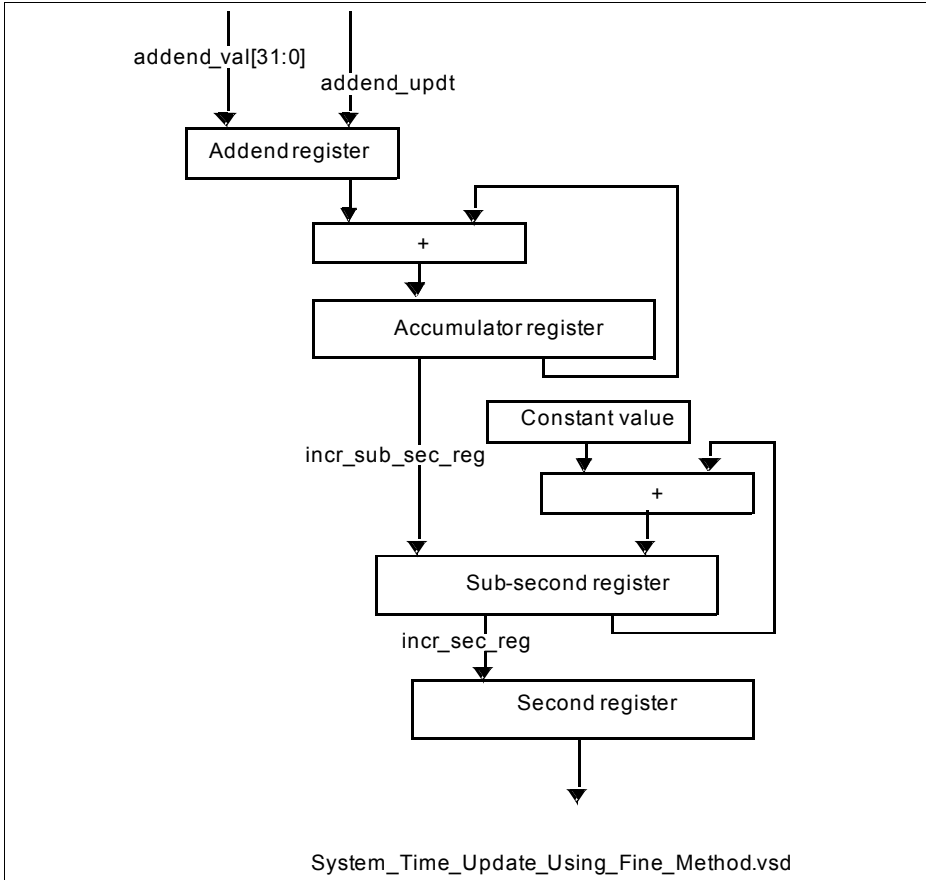


Figure 35-10 System Time Update Using Fine Method

The System Time Update logic requires a 50-MHz clock frequency to achieve 20-ns accuracy. The frequency division is the ratio of the reference clock frequency to the required clock frequency. Hence, if the reference clock (`clk_ptp_ref_i`) is, for example, 66 MHz, this ratio is calculated as $66 \text{ MHz} / 50 \text{ MHz} = 1.32$. Hence, the default addend value to be set in the register is $2^{32} / 1.32$, `0xC1F07C1F`.

Ethernet MAC (ETH)

If the reference clock drifts lower, to 65 MHz for example, the ratio is $65 / 50$, or 1.3 and the value to set in the addend register is $2^{32} / 1.30$, or 0xC4EC4EC4. If the clock drifts higher, to 67 MHz for example, the addend register must be set to 0xBF0B7672. When the clock drift is nil, the default addend value of 0xC1F07C1F ($2^{32} / 1.32$) must be programmed.

In **Figure 35-10**, the constant value used to accumulate the sub-second register is decimal 43, which achieves an accuracy of 20 ns in the system time (in other words, it is incremented in 20-ns steps). Two different methods are used to update the System Time register, depending on which configuration you choose (See **Architecture**).

The software must calculate the drift in frequency based on the Sync messages and update the Addend register accordingly.

Initially, the slave clock is set with FreqCompensationValue0 in the Addend register. This value is as follows:

$$\text{FreqCompensationValue}_0 = 2^{32} / \text{FreqDivisionRatio}$$

If MasterToSlaveDelay is initially assumed to be the same for consecutive Sync messages, the algorithm described below must be applied. After a few Sync cycles, frequency lock occurs. The slave clock can then determine a precise MasterToSlaveDelay value and re-synchronize with the master using the new value.

The algorithm is as follows:

- At time MasterSyncTime_n the master sends the slave clock a Sync message. The slave receives this message when its local clock is SlaveClockTime_n and computes MasterClockTime_n as:

$$\text{MasterClockTime}_n = \text{MasterSyncTime}_n + \text{MasterToSlaveDelay}_n$$
- The master clock count for current Sync cycle, MasterClockCount_n is given by:

$$\text{MasterClockCount}_n = \text{MasterClockTime}_n - \text{MasterClockTime}_{n-1}$$
 (assuming that MasterToSlaveDelay is the same for Sync cycles n and n – 1)
- The slave clock count for current Sync cycle, SlaveClockCount_n is given by:

$$\text{SlaveClockCount}_n = \text{SlaveClockTime}_n - \text{SlaveClockTime}_{n-1}$$
- The difference between master and slave clock counts for current Sync cycle, ClockDiffCount_n is given by:

$$\text{ClockDiffCount}_n = \text{MasterClockCount}_n - \text{SlaveClockCount}_n$$
- The frequency-scaling factor for slave clock, FreqScaleFactor_n is given by:

$$\text{FreqScaleFactor}_n = (\text{MasterClockCount}_n + \text{ClockDiffCount}_n) / \text{SlaveClockCount}_n$$
- The frequency compensation value for Addend register, FreqCompensationValue_n is given by:

$$\text{FreqCompensationValue}_n = \text{FreqScaleFactor}_n * \text{FreqCompensationValue}_{n-1}$$

In theory, this algorithm achieves lock in one Sync cycle; however, it may take several cycles, due to changing network propagation delays and operating conditions.

This algorithm is self-correcting: if for any reason the slave clock is initially set to a value from the master that is incorrect, the algorithm will correct it at the cost of more Sync cycles.

35.2.7 MAC Management Counters

The counters in the MAC Management Counters (MMC) module can be viewed as an extension of the register address space of the CSR module. The MMC module maintains a set of registers for gathering statistics on the received and transmitted frames. These include a control register for controlling the behavior of the registers, two 32-bit registers containing interrupts generated (receive and transmit), and two 32-bit registers containing masks for the Interrupt register (receive and transmit). These registers are accessible from the Application through the MAC Control Interface (MCI). Each register is 32 bits wide. The write data is qualified with the corresponding `mci_be_i` signals. Thus, non-32-bit accesses are allowed as long as the address is word-aligned.

The organization of these registers is shown in [Table 1-17](#). The MMCs are accessed using transactions, in the same way the CSR address space is accessed. The following sections in the chapter describe the various counters and list the address for each of the statistics counters. This address will be used for Read/Write accesses to the desired transmit/receive counter.

The Receive MMC counters are updated for frames that are passed by the Address Filter (AFM) block. Statistics of frames that are dropped by the AFM module are not updated unless they are runt frames of less than 6 bytes (DA bytes are not received fully).

The MMC module gathers statistics on encapsulated IPv4, IPv6, TCP, UDP, or ICMP payloads in received Ethernet frames. This gathering is only enabled when Full Checksum Offload Engine is selected during RTL configuration. The address map of the corresponding registers, 0x0200–0x02FC, is given in [Table 1-17](#).

35.2.7.1 Address Assignments

The MMC register naming convention is as follows.

- “tx” as a prefix or suffix indicates counters associated with transmission
- “rx” as a prefix or suffix indicates counters associated with reception
- “_g” as a suffix indicates registers that count good frames only
- “_gb” as a suffix indicates registers that count frames regardless of whether they are good or bad

The following explanations pertain to terminology used in [Table 1-17](#) through [Table 1-24](#).

Transmitted frames are considered “good” if transmitted successfully. In other words, a transmitted frame is good if the frame transmission is not aborted due to any of the following errors:

- Jabber Timeout

- No Carrier/Loss of Carrier
- Late Collision
- Frame Underflow
- Excessive Deferral
- Excessive Collision

Received frames are considered “good” if none of the following errors exists:

- CRC error
- Runt Frame (shorter than 64 bytes)
- Alignment error (in 10/100 Mbit/s only)
- Length error (non-Type frames only)
- Out of Range (non-Type frames only, longer than maximum size)

The maximum frame size depends on the frame type, as follows:

- Untagged frame maxsize = 1518
- VLAN Frame maxsize = 1522
- Jumbo Frame maxsize = 9018
- JumboVLAN Frame maxsize = 9022

Table 35-10 MMC Register Map

GMAC CSR Register No.	Address Offset	Register Name	Register Description
64	0x0100	mmc_cntrl	MMC Control establishes the operating mode of MMC. For more details, refer to Table 1-18 .
65	0x0104	mmc_intr_rx	MMC Receive Interrupt maintains the interrupt generated from all of the receive statistic counters. For more details, refer to Table 1-19 .
66	0x0108	mmc_intr_tx	MMC Transmit Interrupt maintains the interrupt generated from all of the transmit statistic counters. For more details, refer to Table 1-20 .
67	0x010C	mmc_intr_mask_rx	MMC Receive Interrupt mask maintains the mask for the interrupt generated from all of the receive statistic counters. For more details, refer to Table 1-21 .

Table 35-10 MMC Register Map (cont'd)

GMAC CSR Register No.	Address Offset	Register Name	Register Description
68	0x0110	mmc_intr_mask_tx	MMC Transmit Interrupt Mask maintains the mask for the interrupt generated from all of the transmit statistic counters. For more details, refer to Table 1-22 .
69	0x0114	txoctetcount_gb	Number of bytes transmitted, exclusive of preamble and retried bytes, in good and bad frames.
70	0x0118	txframecount_gb	Number of good and bad frames transmitted, exclusive of retried frames.
71	0x011C	txbroadcastframes_g	Number of good broadcast frames transmitted.
72	0x0120	txmulticastframes_g	Number of good multicast frames transmitted.
73	0x0124	tx64octets_gb	Number of good and bad frames transmitted with length 64 bytes, exclusive of preamble and retried frames.
74	0x0128	tx65to127octets_gb	Number of good and bad frames transmitted with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried frames.
75	0x012C	tx128to255octets_gb	Number of good and bad frames transmitted with length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried frames.
76	0x0130	tx256to511octets_gb	Number of good and bad frames transmitted with length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried frames.
77	0x0134	tx512to1023octets_g	Number of good and bad frames transmitted with length between 512 and 1.023 (inclusive) bytes, exclusive of preamble and retried frames.

Table 35-10 MMC Register Map (cont'd)

GMAC CSR Register No.	Address Offset	Register Name	Register Description
78	0x0138	tx1024tomaxoctets_gb	Number of good and bad frames transmitted with length between 1.024 and maxsize (inclusive) bytes, exclusive of preamble and retried frames.
79	0x013C	txunicastframes_gb	Number of good and bad unicast frames transmitted.
80	0x0140	txmulticastframes_gb	Number of good and bad multicast frames transmitted.
81	0x0144	txbroadcastframes_gb	Number of good and bad broadcast frames transmitted.
82	0x0148	txunderflowerror	Number of frames aborted due to frame underflow error.
83	0x014C	txsinglecol_g	Number of successfully transmitted frames after a single collision in Half-duplex mode.
84	0x0150	txmulticol_g	Number of successfully transmitted frames after more than a single collision in Half-duplex mode.
85	0x0154	txdeferred	Number of successfully transmitted frames after a deferral in Half-duplex mode.
86	0x0158	txlatecol	Number of frames aborted due to late collision error.
87	0x015C	txexesscol	Number of frames aborted due to excessive (16) collision errors.
88	0x0160	txcarriererror	Number of frames aborted due to carrier sense error (no carrier or loss of carrier).
89	0x0164	txoctetcount_g	Number of bytes transmitted, exclusive of preamble, in good frames only.
90	0x0168	txframecount_g	Number of good frames transmitted.
91	0x016C	txexcessdef	Number of frames aborted due to excessive deferral error (deferred for more than two max-sized frame times).

Table 35-10 MMC Register Map (cont'd)

GMAC CSR Register No.	Address Offset	Register Name	Register Description
92	0x0170	txpauseframes	Number of good PAUSE frames transmitted.
93	0x0174	txvlanframes_g	Number of good VLAN frames transmitted, exclusive of retried frames.
94	0x0178	Reserved	
95	0x017C	Reserved	
96	0x0180	rxframecount_gb	Number of good and bad frames received.
97	0x0184	rxoctetcount_gb	Number of bytes received, exclusive of preamble, in good and bad frames.
98	0x0188	rxoctetcount_g	Number of bytes received, exclusive of preamble, only in good frames.
99	0x018C	rxbroadcastframes_g	Number of good broadcast frames received.
100	0x0190	rxmulticastframes_g	Number of good multicast frames received.
101	0x0194	rxrcrcerror	Number of frames received with CRC error.
102	0x0198	rxalignmenterror	Number of frames received with alignment (dribble) error.
103	0x019C	rxrunterror	Number of frames received with runt (<64 bytes and CRC error) error.
104	0x01A0	rxjabbererror	Number of giant frames received with length (including CRC) greater than 1.518 bytes (1.522 bytes for VLAN tagged) and with CRC error. If Jumbo Frame mode is enabled, then frames of length greater than 9,018 bytes (9,022 for VLAN tagged) are considered as giant frames.
105	0x01A4	rxundersize_g	Number of frames received with length less than 64 bytes, without any errors.
106	0x01A8	rxoversize_g	Number of frames received with length greater than the maxsize (1.518 or 1.522 for VLAN tagged frames), without errors.

Table 35-10 MMC Register Map (cont'd)

GMAC CSR Register No.	Address Offset	Register Name	Register Description
107	0x01AC	rx64octets_gb	Number of good and bad frames received with length 64 bytes, exclusive of preamble.
108	0x01B0	rx65to127octets_gb	Number of good and bad frames received with length between 65 and 127 (inclusive) bytes, exclusive of preamble.
109	0x01B4	rx128to255octets_gb	Number of good and bad frames received with length between 128 and 255 (inclusive) bytes, exclusive of preamble.
110	0x01B8	rx256to511octets_gb	Number of good and bad frames received with length between 256 and 511 (inclusive) bytes, exclusive of preamble.
111	0x01BC	rx512to1023octets_gb	Number of good and bad frames received with length between 512 and 1.023 (inclusive) bytes, exclusive of preamble.
112	0x01C0	rx1024tomaxoctets_gb	Number of good and bad frames received with length between 1.024 and maxsize (inclusive) bytes, exclusive of preamble and retried frames.
113	0x01C4	rxunicastframes_g	Number of good unicast frames received.
114	0x01C8	rxlengtherror	Number of frames received with length error (Length type field $\frac{1}{4}$ frame size), for all frames with valid length field.
115	0x01CC	rxoutofrangetype	Number of frames received with length field not equal to the valid frame size (greater than 1.500 but less than 1.536).
116	0x01D0	rxpauseframes	Number of good and valid PAUSE frames received.
117	0x01D4	rxfifooverflow	Number of missed received frames due to FIFO overflow.
118	0x01D8	rxvlanframes_gb	Number of good and bad VLAN frames received.

Table 35-10 MMC Register Map (cont'd)

GMAC CSR Register No.	Address Offset	Register Name	Register Description
119	0x01DC	rxwatchdogerror	Number of frames received with error due to watchdog timeout error (frames with a data load larger than 2.048 bytes).
120:127	0x01E0–0x01FC	Reserved	
128	0x0200	mmc_ipc_intr_mask_rx	MMC IPC Receive Checksum Offload Interrupt Mask maintains the mask for the interrupt generated from the receive IPC statistic counters. See Table 1-23 for further detail.
129	0x0204	Reserved	
130	0x0208	mmc_ipc_intr_rx	MMC Receive Checksum Offload Interrupt maintains the interrupt that the receive IPC statistic counters generate. See Table 1-24 for further detail.
131	0x020C	Reserved	
132	0x0210	rxipv4_gd_frms	Number of good IPv4 datagrams received with the TCP, UDP, or ICMP payload
133	0x0214	rxipv4_hdrerr_frms	Number of IPv4 datagrams received with header (checksum, length, or version mismatch) errors
134	0x0218	rxipv4_nopay_frms	Number of IPv4 datagram frames received that did not have a TCP, UDP, or ICMP payload processed by the Checksum engine
135	0x021C	rxipv4_frag_frms	Number of good IPv4 datagrams with fragmentation
136	0x0220	rxipv4_udtbl_frms	Number of good IPv4 datagrams received that had a UDP payload with checksum disabled
137	0x0224	rxipv6_gd_frms	Number of good IPv6 datagrams received with TCP, UDP, or ICMP payloads

Table 35-10 MMC Register Map (cont'd)

GMAC CSR Register No.	Address Offset	Register Name	Register Description
138	0x0228	rxipv6_hdrerr_frms	Number of IPv6 datagrams received with header errors (length or version mismatch)
139	0x022C	rxipv6_nopay_frms	Number of IPv6 datagram frames received that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers
140	0x0230	rxudp_gd_frms	Number of good IP datagrams with a good UDP payload. This counter is not updated when the rxipv4_udtbl_frms counter is incremented.
141	0x0234	rxudp_err_frms	Number of good IP datagrams whose UDP payload has a checksum error
142	0x0238	rttcp_gd_frms	Number of good IP datagrams with a good TCP payload
143	0x023C	rttcp_err_frms	Number of good IP datagrams whose TCP payload has a checksum error
144	0x0240	rxicmp_gd_frms	Number of good IP datagrams with a good ICMP payload
145	0x0244	rxicmp_err_frms	Number of good IP datagrams whose ICMP payload has a checksum error
146:147	0x0248–0x024C	Reserved	
148	0x0250	rxipv4_gd_octets	Number of bytes received in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter or in the octet counters listed below).
149	0x0254	rxipv4_hdrerr_octets	Number of bytes received in IPv4 datagrams with header errors (checksum, length, version mismatch). The value in the Length field of IPv4 header is used to update this counter.

Table 35-10 MMC Register Map (cont'd)

GMAC CSR Register No.	Address Offset	Register Name	Register Description
150	0x0258	rxipv4_nopay_octets	Number of bytes received in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the IPv4 header's Length field is used to update this counter.
151	0x025C	rxipv4_frag_octets	Number of bytes received in fragmented IPv4 datagrams. The value in the IPv4 header's Length field is used to update this counter.
152	0x0260	rxipv4_udtbl_octets	Number of bytes received in a UDP segment that had the UDP checksum disabled. This counter does not count IP Header bytes.
153	0x0264	rxipv6_gd_octets	Number of bytes received in good IPv6 datagrams encapsulating TCP, UDP or ICMPv6 data
154	0x0268	rxipv6_hdrerr_octets	Number of bytes received in IPv6 datagrams with header errors (length, version mismatch). The value in the IPv6 header's Length field is used to update this counter.
155	0x026C	rxipv6_nopay_octets	Number of bytes received in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the IPv6 header's Length field is used to update this counter.
156	0x0270	rxudp_gd_octets	Number of bytes received in a good UDP segment. This counter (and the counters below) does not count IP header bytes.
157	0x0274	rxudp_err_octets	Number of bytes received in a UDP segment that had checksum errors
158	0x0278	rtcp_gd_octets	Number of bytes received in a good TCP segment

Table 35-10 MMC Register Map (cont'd)

GMAC CSR Register No.	Address Offset	Register Name	Register Description
159	0x027C	rxtcp_err_octets	Number of bytes received in a TCP segment with checksum errors
160	0x0280	rxicmp_gd_octets	Number of bytes received in a good ICMP segment
161	0x0284	rxicmp_err_octets	Number of bytes received in an ICMP segment with checksum errors
162:191	0x0288–0x02FC	Reserved	

35.2.7.2 MMC Register Description

MMC Control Register

The MMC Control register establishes the operating mode of the management counters.

Table 35-11 MMC Control Register

Bit	Access Type	Reset Value	Description
31:6	R	0000_000H	Reserved
5	rw	0	Full-Half preset When low and bit4 is set, all MMC counters get preset to almost-half value. All octet counters get preset to 0x7FFF_F800 (half - 2KBytes) and all frame-counters gets preset to 0x7FFF_FFF0 (half - 16) When high and bit4 is set, all MMC counters get preset to almost-full value. All octet counters get preset to 0xFFFF_F800 (full - 2KBytes) and all frame-counters gets preset to 0xFFFF_FFF0 (full - 16)

Table 35-11 MMC Control Register (cont'd)

Bit	Access Type	Reset Value	Description
4	rwh	0	Counters Preset When set, all counters will be initialized or preset to almost full or almost half as per Bit5 above. This bit will be cleared automatically after 1 clock cycle. This bit along with bit5 is useful for debugging and testing the assertion of interrupts due to MMC counter becoming half-full or full.
3	rw	0	MMC Counter Freeze When set, this bit freezes all the MMC counters to their current value. (None of the MMC counters are updated due to any transmitted or received frame until this bit is reset to 0. If any MMC counter is read with the Reset on Read bit set, then that counter is also cleared in this mode.)
2	rw	0	Reset on Read When set, the MMC counters will be reset to zero after Read (self-clearing after reset). The counters are cleared when the least significant byte lane (bits[7:0]) is read.
1	rw	0	Counter Stop Rollover When set, counter after reaching maximum value will not roll over to zero.
0	rwh	0	Counters Reset When set, all counters will be reset. This bit will be cleared automatically after 1 clock cycle

Note: When Bit 0 and Bit 4 are set in the same cycle, Counters get preset immediately after getting cleared.

MMC Receive Interrupt Register

The MMC Receive Interrupt register maintains the interrupts generated when the receive statistic counters reach half their maximum values (0x8000_0000), and when they cross their maximum values (0xFFFF_FFFF). When Counter Stop Rollover is set, then interrupts are set but the counter remains at all-ones. The MMC Receive Interrupt register is a 32-bit wide register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (bits[7:0]) of the respective counter must be read in order to clear the interrupt bit.

Note: Access restriction applies: Any read access to the register clears this register!

Table 35-12 MMC Receive Interrupt Register

Bit	Access Type	Reset Value	Description
31:24	r	00H	Reserved
23	rh	0	The bit is set when the rxwatchdog error counter reaches half the maximum value, and also when it reaches the maximum value.
22	rh	0	The bit is set when the rxvlanframes_gb counter reaches half the maximum value, and also when it reaches the maximum value.
21	rh	0	The bit is set when the rxfifooverflow counter reaches half the maximum value, and also when it reaches the maximum value.
20	rh	0	The bit is set when the rxpauseframes counter reaches half the maximum value, and also when it reaches the maximum value.
19	rh	0	The bit is set when the rxoutofrangetype counter reaches half the maximum value, and also when it reaches the maximum value.
18	rh	0	The bit is set when the rxlengtherror counter reaches half the maximum value, and also when it reaches the maximum value.
17	rh	0	The bit is set when the rxunicastframes_g counter reaches half the maximum value, and also when it reaches the maximum value.
16	rh	0	The bit is set when the rx1024tomaxoctets_gb counter reaches half the maximum value, and also when it reaches the maximum value.
15	rh	0	The bit is set when the rx512to1023octets_gb counter reaches half the maximum value, and also when it reaches the maximum value.
14	rh	0	The bit is set when the rx256to511octets_gb counter reaches half the maximum value, and also when it reaches the maximum value.
13	rh	0	The bit is set when the rx128to255octets_gb counter reaches half the maximum value, and also when it reaches the maximum value.

Table 35-12 MMC Receive Interrupt Register (cont'd)

Bit	Access Type	Reset Value	Description
12	rh	0	The bit is set when the rx65to127octets_gb counter reaches half the maximum value, and also when it reaches the maximum value.
11	rh	0	The bit is set when the rx64octets_gb counter reaches half the maximum value, and also when it reaches the maximum value.
10	rh	0	The bit is set when the rxoversize_g counter reaches half the maximum value, and also when it reaches the maximum value.
9	rh	0	The bit is set when the rxundersize_g counter reaches half the maximum value, and also when it reaches the maximum value.
8	rh	0	The bit is set when the rxjabbererror counter reaches half the maximum value, and also when it reaches the maximum value.
7	rh	0	The bit is set when the rxrunterror counter reaches half the maximum value, and also when it reaches the maximum value.
6	rh	0	The bit is set when the rxalignmenterror counter reaches half the maximum value, and also when it reaches the maximum value.
5	rh	0	The bit is set when the rxrcerror counter reaches half the maximum value, and also when it reaches the maximum value.
4	rh	0	The bit is set when the rxmulticastframes_g counter reaches half the maximum value, and also when it reaches the maximum value.
3	rh	0	The bit is set when the rxbroadcastframes_g counter reaches half the maximum value, and also when it reaches the maximum value.
2	rh	0	The bit is set when the rxoctetcount_g counter reaches half the maximum value, and also when it reaches the maximum value.

Table 35-12 MMC Receive Interrupt Register (cont'd)

Bit	Access Type	Reset Value	Description
1	rh	0	The bit is set when the rxoctetcount_gb counter reaches half the maximum value, and also when it reaches the maximum value.
0	rh	0	The bit is set when the rxframecount_gb counter reaches half the maximum value, and also when it reaches the maximum value.

Note: rh means that this register bit is set internally and is cleared when the Counter register is read.

MMC Transmit Interrupt Register

The MMC Transmit Interrupt register maintains the interrupts generated when transmit statistic counters reach half their maximum values (0x8000_0000), and when they cross their maximum values (0xFFFF_FFFF). When Counter Stop Rollover is set, then interrupts are set but the counter remains at all-ones. The MMC Transmit Interrupt register is a 32-bit wide register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (bits[7:0]) of the respective counter must be read in order to clear the interrupt bit.

Note: Access restriction applies: Any read access to the register clears this register!

Table 35-13 MMC Transmit Interrupt Register

Bit	Access Type	Reset Value	Description
31:25	r	00 _H	Reserved
24	rh	0	The bit is set when the txvlanframes_g counter reaches half the maximum value, and also when it reaches the maximum value.
23	rh	0	The bit is set when the txpauseframes error counter reaches half the maximum value, and also when it reaches the maximum value.
22	rh	0	The bit is set when the txoexcessdef counter reaches half the maximum value, and also when it reaches the maximum value.
21	rh	0	The bit is set when the txframecount_g counter reaches half the maximum value, and also when it reaches the maximum value.

Table 35-13 MMC Transmit Interrupt Register (cont'd)

Bit	Access Type	Reset Value	Description
20	rh	0	The bit is set when the txoctetcount_g counter reaches half the maximum value, and also when it reaches the maximum value.
19	rh	0	The bit is set when the txcarriererror counter reaches half the maximum value, and also when it reaches the maximum value.
18	rh	0	The bit is set when the txexesscol counter reaches half the maximum value, and also when it reaches the maximum value.
17	rh	0	The bit is set when the txlatecol counter reaches half the maximum value, and also when it reaches the maximum value.
16	rh	0	The bit is set when the txdeferred counter reaches half the maximum value, and also when it reaches the maximum value.
15	rh	0	The bit is set when the txmulticol_g counter reaches half the maximum value, and also when it reaches the maximum value.
14	rh	0	The bit is set when the txsinglecol_g counter reaches half the maximum value, and also when it reaches the maximum value.
13	rh	0	The bit is set when the txunderflowerror counter reaches half the maximum value, and also when it reaches the maximum value.
12	rh	0	The bit is set when the txbroadcastframes_gb counter reaches half the maximum value, and also when it reaches the maximum value.
11	rh	0	The bit is set when the txmulticastframes_gb counter reaches half the maximum value, and also when it reaches the maximum value.
10	rh	0	The bit is set when the txunicastframes_gb counter reaches half the maximum value, and also when it reaches the maximum value.

Table 35-13 MMC Transmit Interrupt Register (cont'd)

Bit	Access Type	Reset Value	Description
9	rh	0	The bit is set when the tx1024tomaxoctets_gb counter reaches half the maximum value, and also when it reaches the maximum value.
8	rh	0	The bit is set when the tx512to1023octets_gb counter reaches half the maximum value, and also when it reaches the maximum value.
7	rh	0	The bit is set when the tx256to511octets_gb counter reaches half the maximum value, and also when it reaches the maximum value.
6	rh	0	The bit is set when the tx128to255octets_gb counter reaches half the maximum value, and also when it reaches the maximum value.
5	rh	0	The bit is set when the tx65to127octets_gb counter reaches half the maximum value, and also when it reaches the maximum value.
4	rh	0	The bit is set when the tx64octets_gb counter reaches half the maximum value, and also when it reaches the maximum value.
3	rh	0	The bit is set when the txmulticastframes_g counter reaches half the maximum value, and also when it reaches the maximum value.
2	rh	0	The bit is set when the txbroadcastframes_g counter reaches half the maximum value, and also when it reaches the maximum value.
1	rh	0	The bit is set when the txframecount_gb counter reaches half the maximum value, and also when it reaches the maximum value.
0	rh	0	The bit is set when the txoctetcount_gb counter reaches half the maximum value, and also when it reaches the maximum value.

MMC Receive Interrupt Mask Register

The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when receive statistic counters reach half their maximum value, and when they reach their maximum values. This register is 32 bits wide.

Table 35-14 MMC Receive Interrupt Mask Interrupt Register

Bit	Access Type	Reset Value	Description
31:24	r	00 _H	Reserved
23	rw	0	Setting this bit masks the interrupt when the rxwatchdog counter reaches half the maximum value, and also when it reaches the maximum value.
22	rw	0	Setting this bit masks the interrupt when the rxvlanframes_gb counter reaches half the maximum value, and also when it reaches the maximum value.
21	rw	0	Setting this bit masks the interrupt when the rxfifooverflow counter reaches half the maximum value, and also when it reaches the maximum value.
20	rw	0	Setting this bit masks the interrupt when the rxpauseframes counter reaches half the maximum value, and also when it reaches the maximum value.
...	rw Same as above for corresponding counters in MMC Receive Interrupt Register
1	rw	0	Setting this bit masks the interrupt when the rxoctetcount_gb counter reaches half the maximum value, and also when it reaches the maximum value.
0	rw	0	Setting this bit masks the interrupt when the rxframecount_gb counter reaches half the maximum value, and also when it reaches the maximum value.

MMC Transmit Interrupt Mask Register

The MMC Transmit Interrupt Mask register maintains the masks for the interrupts generated when transmit statistic counters reach half their maximum value, and when they reach their maximum values. This register is 32 bits wide.

Table 35-15 MMC Transmit Interrupt Mask Register

Bit	Access Type	Reset Value	Description
31:25	r	00 _H	Reserved
24	rw	0	Setting this bit masks the interrupt when the txvlanframes_g counter reaches half the maximum value, and also when it reaches the maximum value.

Table 35-15 MMC Transmit Interrupt Mask Register (cont'd)

Bit	Access Type	Reset Value	Description
23	rw	0	Setting this bit masks the interrupt when the txpauseframes counter reaches half the maximum value, and also when it reaches the maximum value.
22	rw	0	Setting this bit masks the interrupt when the txoexcessdef counter reaches half the maximum value, and also when it reaches the maximum value.
21	rw	0	Setting this bit masks the interrupt when the txframecount_g counter reaches half the maximum value, and also when it reaches the maximum value.
....	rw Same as above for corresponding counters in MMC Transmit Interrupt Register
1	rw	0	Setting this bit masks the interrupt when the txframecount_gb counter reaches half the maximum value, and also when it reaches the maximum value.
0	rw	0	Setting this bit masks the interrupt when the txoctetcount_gb counter reaches half the maximum value, and also when it reaches the maximum value.

MMC Receive Checksum Offload Interrupt Mask Register

The MMC Receive Checksum Offload Interrupt Mask register maintains the masks for the interrupts generated when the receive IPC (Checksum Offload) statistic counters reach half their maximum value , and when they reach their maximum values. This register is 32-bits wide.

Table 35-16 MMC Receive Checksum Offload Interrupt Mask Register

Bit	Access Type	Reset Value	Description
31:30	r	00	Reserved
29	rw	0	Setting this bit masks the interrupt when the rxicmp_err_octets counter reaches half the maximum value, and also when it reaches the maximum value.
28	rw	0	Setting this bit masks the interrupt when the rxicmp_gd_octets counter reaches half the maximum value, and also when it reaches the maximum value.

Table 35-16 MMC Receive Checksum Offload Interrupt Mask Register (cont'd)

Bit	Access Type	Reset Value	Description
27	rw	0	Setting this bit masks the interrupt when the <code>rxtcp_err_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
26	rw	0	Setting this bit masks the interrupt when the <code>rxtcp_gd_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
25	rw	0	Setting this bit masks the interrupt when the <code>rxudp_err_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
24	rw	0	Setting this bit masks the interrupt when the <code>rxudp_gd_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
23	rw	0	Setting this bit masks the interrupt when the <code>rxipv6_nopay_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
22	rw	0	Setting this bit masks the interrupt when the <code>rxipv6_hdrerr_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
21	rw	0	Setting this bit masks the interrupt when the <code>rxipv6_gd_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
20	rw	0	Setting this bit masks the interrupt when the <code>rxipv4_udtbl_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
19	rw	0	Setting this bit masks the interrupt when the <code>rxipv4_frag_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
18	rw	0	Setting this bit masks the interrupt when the <code>rxipv4_nopay_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
17	rw	0	Setting this bit masks the interrupt when the <code>rxipv4_hdrerr_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.

Table 35-16 MMC Receive Checksum Offload Interrupt Mask Register (cont'd)

Bit	Access Type	Reset Value	Description
16	rw	0	Setting this bit masks the interrupt when the rxipv4_gd_octets counter reaches half the maximum value, and also when it reaches the maximum value.
15:14	r	0	Reserved
13	rw	0	Setting this bit masks the interrupt when the rxicmp_err_frms counter reaches half the maximum value, and also when it reaches the maximum value.
12	rw	0	Setting this bit masks the interrupt when the rxicmp_gd_frms counter reaches half the maximum value, and also when it reaches the maximum value.
11	rw	0	Setting this bit masks the interrupt when the rxtcp_err_frms counter reaches half the maximum value, and also when it reaches the maximum value.
10	rw	0	Setting this bit masks the interrupt when the rxtcp_gd_frms counter reaches half the maximum value, and also when it reaches the maximum value.
9	rw	0	Setting this bit masks the interrupt when the rxudp_err_frms counter reaches half the maximum value, and also when it reaches the maximum value.
8	rw	0	Setting this bit masks the interrupt when the rxudp_gd_frms counter reaches half the maximum value, and also when it reaches the maximum value.
7	rw	0	Setting this bit masks the interrupt when the rxipv6_nopay_frms counter reaches half the maximum value, and also when it reaches the maximum value.
6	rw	0	Setting this bit masks the interrupt when the rxipv6_hdrerr_frms counter reaches half the maximum value, and also when it reaches the maximum value.
5	rw	0	Setting this bit masks the interrupt when the rxipv6_gd_frms counter reaches half the maximum value, and also when it reaches the maximum value.
4	rw	0	Setting this bit masks the interrupt when the rxipv4_udtbl_frms counter reaches half the maximum value, and also when it reaches the maximum value.

Table 35-16 MMC Receive Checksum Offload Interrupt Mask Register (cont'd)

Bit	Access Type	Reset Value	Description
3	rw	0	Setting this bit masks the interrupt when the rxipv4_frag_frms counter reaches half the maximum value, and also when it reaches the maximum value.
2	rw	0	Setting this bit masks the interrupt when the rxipv4_nopay_frms counter reaches half the maximum value, and also when it reaches the maximum value.
1	rw	0	Setting this bit masks the interrupt when the rxipv4_hdrerr_frms counter reaches half the maximum value, and also when it reaches the maximum value.
0	rw	0	Setting this bit masks the interrupt when the rxipv4_gd_frms counter reaches half the maximum value, and also when it reaches the maximum value.

MMC Receive Checksum Offload Interrupt Register

The MMC Receive Checksum Offload Interrupt register maintains the interrupts generated when receive IPC statistic counters reach half their maximum values (0x8000_0000), and when they cross their maximum values (0xFFFF_FFFF). When Counter Stop Rollover is set, then interrupts are set but the counter remains at all-ones. The MMC Receive Checksum Offload Interrupt register is 32 bits wide. When the MMC IPC counter that caused the interrupt is read, its corresponding interrupt bit is cleared. The counter's least-significant byte lane (bits[7:0]) must be read to clear the interrupt bit.

Note: Access restriction applies: Any read access to the register clears this register!

Table 35-17 MMC Receive Checksum Offload Interrupt Register

Bit	Access Type	Reset Value	Description
31:30	r	00	Reserved
29	rh	0	The bit is set when the rxicmp_err_octets counter reaches half the maximum value, and also when it reaches the maximum value.
28	rh	0	The bit is set when the rxicmp_gd_octets counter reaches half the maximum value, and also when it reaches the maximum value.

Table 35-17 MMC Receive Checksum Offload Interrupt Register (cont'd)

Bit	Access Type	Reset Value	Description
27	rh	0	The bit is set when the <code>rxtcp_err_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
26	rh	0	The bit is set when the <code>rxtcp_gd_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
25	rh	0	The bit is set when the <code>rxudp_err_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
24	rh	0	The bit is set when the <code>rxudp_gd_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
23	rh	0	The bit is set when the <code>rxipv6_nopay_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
22	rh	0	The bit is set when the <code>rxipv6_hdrerr_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
21	rh	0	The bit is set when the <code>rxipv6_gd_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
20	rh	0	The bit is set when the <code>rxipv4_udtbl_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
19	rh	0	The bit is set when the <code>rxipv4_frag_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
18	rh	0	The bit is set when the <code>rxipv4_nopay_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.
17	rh	0	The bit is set when the <code>rxipv4_hdrerr_octets</code> counter reaches half the maximum value, and also when it reaches the maximum value.

Ethernet MAC (ETH)
Table 35-17 MMC Receive Checksum Offload Interrupt Register (cont'd)

Bit	Access Type	Reset Value	Description
16	rh	0	The bit is set when the rxiipv4_gd_octets counter reaches half the maximum value, and also when it reaches the maximum value.
15:14	r	00	Reserved
13	rh	0	The bit is set when the rxiicmp_err_frms counter reaches half the maximum value, and also when it reaches the maximum value.
12	rh	0	The bit is set when the rxiicmp_gd_frms counter reaches half the maximum value, and also when it reaches the maximum value.
11	rh	0	The bit is set when the rxtcp_err_frms counter reaches half the maximum value, and also when it reaches the maximum value.
10	rh	0	The bit is set when the rxtcp_gd_frms counter reaches half the maximum value, and also when it reaches the maximum value.
9	rh	0	The bit is set when the rxudp_err_frms counter reaches half the maximum value, and also when it reaches the maximum value.
8	rh	0	The bit is set when the rxudp_gd_frms counter reaches half the maximum value, and also when it reaches the maximum value.
7	rh	0	The bit is set when the rxiipv6_nopay_frms counter reaches half the maximum value, and also when it reaches the maximum value.
6	rh	0	The bit is set when the rxiipv6_hdrerr_frms counter reaches half the maximum value, and also when it reaches the maximum value.
5	rh	0	The bit is set when the rxiipv6_gd_frms counter reaches half the maximum value, and also when it reaches the maximum value.
4	rh	0	The bit is set when the rxiipv4_udtbl_frms counter reaches half the maximum value, and also when it reaches the maximum value.

Table 35-17 MMC Receive Checksum Offload Interrupt Register (cont'd)

Bit	Access Type	Reset Value	Description
3	rh	0	The bit is set when the rxipv4_frag_frms counter reaches half the maximum value, and also when it reaches the maximum value.
2	rh	0	The bit is set when the rxipv4_nopay_frms counter reaches half the maximum value, and also when it reaches the maximum value.
1	rh	0	The bit is set when the rxipv4_hdrerr_frms counter reaches half the maximum value, and also when it reaches the maximum value.
0	rh	0	The bit is set when the rxipv4_gd_frms counter reaches half the maximum value, and also when it reaches the maximum value.

35.2.8 Power Management Block

This section describes the power management (PMT) mechanisms supported by the GMAC. PMT supports the reception of network (remote) wake-up frames and Magic Packet frames. PMT does not perform the clock gate function, but generates interrupts for wake-up frames and Magic Packets received by the GMAC. The PMT block sits on the receiver path of the GMAC and is enabled with remote wake-up frame enable and Magic Packet enable. These enables are in the PMT Control and Status register and are programmed by the Application. You can include the optional PMT module by selecting it in the coreKit during configuration. You have the option to select either or both types of power-management frames (remote wake-up and Magic Packet)

PMT registers are accessed in the same manner as with GMAC-CSR registers. Refer to registers 10 and 11, for mapping information.

When the power down mode is enabled in the PMT, then all received frames are dropped by the core and they are not forwarded to the application. The core comes out of the power down mode only when either a Magic Packet or a Remote Wake-up frame is received and the corresponding detection is enabled.

35.2.8.1 PMT Block Description

PMT Control and Status Register

The PMT CSR program the request wake-up events and monitor the wake-up events.

Note: Access restriction applies: Any read access to the register clears this register!

Table 35-18 PMT Control and Status Register

Bit	Access Type	Reset Value	Description
31	rwh	0	Wake-Up Frame Filter Register Pointer Reset When set, resets the Remote Wake-up Frame Filter register pointer to 3'b000. It is automatically cleared after 1 clock cycle. Access restriction applies: setting sets + clearing has no effect.
30:10	R	0000_00 _H	Reserved
9	rw	00	Global Unicast When set, enables any unicast packet filtered by the GMAC (DAF) address recognition to be a wake-up frame.
8:7	R	0	Reserved
6	rh	0	Wake-Up Frame Received When set, this bit indicates the power management event was generated due to reception of a wake-up frame. This bit is cleared by a Read into this register.
5	rh	0	Magic Packet Received When set, this bit indicates the power management event was generated by the reception of a Magic Packet. This bit is cleared by a Read into this register.
4:3	R	00	Reserved
2	rw	0	Wake-Up Frame Enable When set, enables generation of a power management event due to wake-up frame reception.

Table 35-18 PMT Control and Status Register (cont'd)

Bit	Access Type	Reset Value	Description
1	rw	0	Magic Packet Enable When set, enables generation of a power management event due to Magic Packet reception.
0	rwh	0	Power Down When set, all received frames will be dropped. This bit is cleared automatically when a magic packet or Wake-Up frame is received, and Power-Down mode is disabled. Frames received after this bit is cleared are forwarded to the application. This bit must only be set when either the Magic Packet Enable, Global Unicast, or Wake-Up Frame Enable bit is set high. Access restriction applies: setting sets + clearing has no effect.

Remote Wake-Up Frame Filter Register

The register `wkupmfilter_reg`, address (028_H), loads the Wake-up Frame Filter register. To load values in a Wake-up Frame Filter register, the entire register (`wkupmfilter_reg`) must be written. The `wkupmfilter_reg` register is loaded by sequentially loading the eight register values in address (028) for `wkupmfilter_reg0`, `wkupmfilter_reg1`,... `wkupmfilter_reg7`, respectively. `wkupmfilter_reg` is read in the same way.

Note: The internal counter to access the appropriate `wkupmfilter_reg` is incremented when `lane3` is accessed by the CPU. This should be kept in mind if you are accessing these registers in byte or half-word mode.

wkupfilter_reg0	Filter 0 Byte Mask							
wkupfilter_reg1	Filter 1 Byte Mask							
wkupfilter_reg2	Filter 2 Byte Mask							
wkupfilter_reg3	Filter 3 Byte Mask							
wkupfilter_reg4	RSVD	Filter 3 Command	RSVD	Filter 2 Command	RSVD	Filter 1 Command	RSVD	Filter 0 Command
wkupfilter_reg5	Filter 3 Offset		Filter 2 Offset		Filter 1 Offset		Filter 0 Offset	
wkupfilter_reg6	Filter 1 CRC - 16				Filter 0 CRC - 16			
wkupfilter_reg7	Filter 3 CRC - 16				Filter 2 CRC - 16			

Wake_Up_Frame_Filter_Register.vsd

Figure 35-11 Wake-Up Frame Filter Register

Filter i Byte Mask

This register defines which bytes of the frame are examined by filter i (0, 1, 2, and 3) in order to determine whether or not the frame is a wake-up frame. The MSB (thirty-first bit) must be zero. Bit j [30:0] is the Byte Mask. If bit j (byte number) of the Byte Mask is set, then Filter i Offset + j of the incoming frame is processed by the CRC block; otherwise Filter i Offset + j is ignored.

Filter i Command

This 4-bit command controls the filter i operation. Bit 3 specifies the address type, defining the pattern's destination address type. When the bit is set, the pattern applies to only multicast frames; when the bit is reset, the pattern applies only to unicast frame. Bit 2 and Bit 1 are reserved. Bit 0 is the enable for filter i; if Bit 0 is not set, filter i is disabled.

Filter i Offset

This register defines the offset (within the frame) from which the frames are examined by filter i. This 8-bit pattern-offset is the offset for the filter i first byte to be examined. The minimum allowed is 12, which refers to the 13th byte of the frame (offset value 0 refers to the first byte of the frame).

Filter i CRC-16

This register contains the CRC_16 value calculated from the pattern, as well as the byte mask programmed to the wake-up filter register block.

35.2.8.2 Remote Wake-Up Frame Detection

When the GMAC is in sleep mode and the remote wake-up bit is enabled in PMT Control and Status register (002C), normal operation is resumed after receiving a remote wake-up frame. The Application writes all eight wake-up filter registers, by performing a sequential Write to address (0028). The Application enables remote wake-up by writing a 1 to Bit 2 of the PMT Control and Status register.

PMT supports four programmable filters that allow support of different receive frame patterns. If the incoming frame passes the address filtering of Filter Command, and if Filter CRC-16 matches the incoming examined pattern, then the wake-up frame is received.

Filter_offset (minimum value 12, which refers to the 13th byte of the frame) determines the offset from which the frame is to be examined. Filter Byte Mask determines which bytes of the frame must be examined. The thirty-first bit of Byte Mask must be set to zero.

The remote wake-up CRC block determines the CRC value that is compared with Filter CRC-16. The wake-up frame is checked only for length error, FCS error, dribble bit error, GMII error, collision, and to ensure that it is not a runt frame. Even if the wake-up frame is more than 512 bytes long, if the frame has a valid CRC value, it is considered valid. Wake-up frame detection is updated in the PMT Control and Status register for every remote Wake-up frame received. A PMT interrupt to the Application triggers a Read to the PMT Control and Status register to determine reception of a wake-up frame.

35.2.8.3 Magic Packet Detection

The Magic Packet frame is based on a method that uses Advanced Micro Device's Magic Packet technology to power up the sleeping device on the network. The GMAC receives a specific packet of information, called a Magic Packet, addressed to the node on the network.

Only Magic Packets that are addressed to the device or a broadcast address will be checked to determine whether they meet the wake-up requirements. Magic Packets that pass the address filtering (unicast or broadcast) will be checked to determine whether they meet the remote Wake-on-LAN data format of 6 bytes of all ones followed by a GMAC Address appearing 16 times.

The application enables Magic Packet wake-up by writing a 1 to Bit 1 of the PMT Control and Status register. The PMT block constantly monitors each frame addressed to the node for a specific Magic Packet pattern. Each frame received is checked for a 48'hFF_FF_FF_FF_FF_FF pattern following the destination and source address field. The PMT block then checks the frame for 16 repetitions of the GMAC address without

Ethernet MAC (ETH)

any breaks or interruptions. In case of a break in the 16 repetitions of the address, the 48'hFF_FF_FF_FF_FF_FF pattern is scanned for again in the incoming frame. The 16 repetitions can be anywhere in the frame, but must be preceded by the synchronization stream (48'hFF_FF_FF_FF_FF_FF). The device will also accept a multicast frame, as long as the 16 duplications of the GMAC address are detected.

If the MAC address of a node is 48'h00_11_22_33_44_55, then the GMAC scans for the data sequence:

```
Destination Address Source Address ..... FF FF FF FF FF FF
00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33
44 55
00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33
44 55
00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33
44 55
00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33
44 55
...CRC
```

Magic Packet detection is updated in the PMT Control and Status register for Magic Packet received. A PMT interrupt to the Application triggers a read to the PMT CSR to determine whether a Magic Packet frame has been received.

35.2.8.4 System Considerations During Power-Down

GMAC-UNIV neither gates nor stops clocks when Power-Down mode is enabled. Power saving by clock gating must be done outside the core by the application. The receive data path must be clocked with clk_rx_i during Power-Down mode, because it is involved in magic packet/wake-on-LAN frame detection. However, the transmit path and the application path clocks can be gated off during Power-Down mode.

The pmt_intr_o signal is asserted when a valid wake-up frame is received. This signal is generated in the clk_rx_i domain.

The recommended power-down and wake-up sequence is as follows.

1. Disable the Transmit DMA (if applicable) and wait for any previous frame transmissions to complete. These transmissions can be detected when Transmit Interrupt (TI-DMA Register 5[0]) is received.
2. Disable the MAC transmitter and MAC receiver by clearing the appropriate bits in the MAC Configuration register.
3. Wait until the Receive DMA empties all the frames from the Rx FIFO (a software timer may be required).
4. Enable Power-Down mode by appropriately configuring the PMT registers.
5. Enable the MAC Receiver and enter Power-Down mode.

Ethernet MAC (ETH)

6. Gate the application and transmit clock inputs to the core (and other relevant clocks in the system) to reduce power and enter Sleep mode.
7. On receiving a valid wake-up frame, the GMAC-UNIV asserts the pmt_intr_o signal and exits Power-Down mode.
8. On receiving the interrupt, the system must enable the application and transmit clock inputs to the core.
9. Read the PMT Status register to clear the interrupt, then enable the other modules in the system and resume normal operation.

35.2.9 Station Management Agent

The Station Management Agent (SMA) module allows the Application to access any PHY registers through a 2-wire Station Management interface (MIM). The interface supports accessing up to 32 PHYs.

The application can select one of the 32 PHYs and one of the 32 registers within any PHY and send control data or receive status information. Only one register in one PHY can be addressed at any given time. For more details on the communication from the Application to the PHYs, refer to the Reconciliation Sublayer and Media Independent Interface Specifications section of the IEEE 802.3z specification, 1000BASE Ethernet. The application sends the control data to the PHY and receives status information from the PHY through the SMA module, as shown in [Figure 35-12](#).

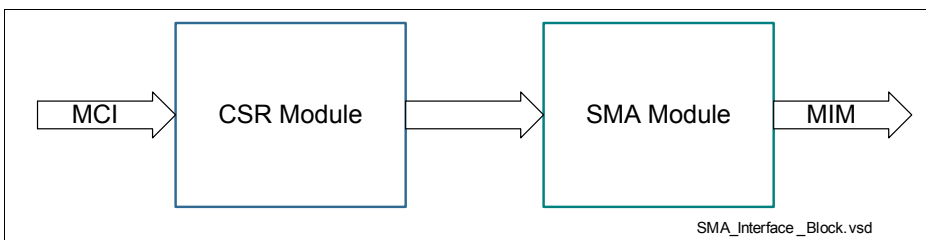


Figure 35-12 SMA Interface Block

35.2.9.1 Functions

The GMAC initiates the Management Write/Read operation. The clock gmii_mdc_o is a divided clock from the Application clock clk_csr_i. The divide factor depends on the clock range setting in the GMII Address register. Clock range is set as follows:

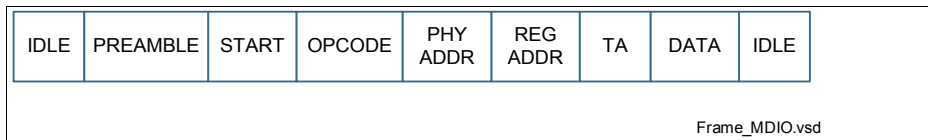
Selection	clk_csr_i	MDC Clock
0000	60-100 MHz	clk_csr_i/42
0001	100-150 MHz	clk_csr_i/62
0010	20-35 MHz	clk_csr_i/16

Ethernet MAC (ETH)

Selection	clk_csr_i	MDC Clock
0011	35-60 MHz	clk_csr_i/26
0100	150-250 MHz	clk_csr_i/102
0101	250-300 MHz	clk_csr_i/124
0110, 0111	Reserved	

The gmii_mdc_o is the derivative of the Application clock clk_csr_i. The Management operation is performed through the gmii_md_i, gmii_mdo_o and gmii_mdo_o_e signals. A three-state buffer is implemented outside the GMAC to interface with an external PHY.

The frame structure on the MDIO line is shown below.



IDLE The mdio line is three-state; there is no clock on gmii_mdc_o

PREAMBLE 32 continuous bits of value 1

START Start-of-frame is 2'01

OPCODE 2'b10 for Read and 2'b01 for Write

PHY ADDR 5-bit address select for one of 32 PHYs

REG ADDR Register address in the selected PHY

TA Turnaround is 2'bZ0 for Read and 2'b10 for Write

DATA Any 16-bit value. In a Write operation, the GMAC drives mdio; in a Read operation, PHY drives it.

35.2.9.2 MII Management Write Operation

When the user sets the GMII Write and Busy bits (see GMII Address Register, **“32-bit Register - GMII_Address” on Page 1-309**), the GMAC CSR module transfers the PHY address, the register address in PHY, and the write data (GMII Data Register) to the SMA to initiate a Write operation into the PHY registers. At this point, the SMA module starts a Write operation on the GMII Management Interface using the Management Frame Format specified in the GMII specifications (Section 22.2.4.5 of IEEE Standard). The application should not change the GMII Address register contents or the GMII Data register while the transaction is ongoing. Write operations to the GMII Address register or the GMII Data Register during this period are ignored (the Busy bit is high), and the transaction is completed without any error on the MCI interface.

Ethernet MAC (ETH)

After the Write operation has completed, the SMA indicates this to the CSR which then resets the Busy bit. The SMA module divides the CSR (Application) clock with the clock divider programmed (CR bits of GMII Address Register) to generate the MDC clock for this interface. The GMAC drives the MDIO line for the complete duration of the frame. The frame format for the Write operation is as follows:

IDLE	PREAMBLE	START	OPCODE	PHY ADDR	REG ADDR	TA	DATA	IDLE
Z	1111...11	01	01	AAAAA	RRRRR	10	DDDDDD	Z

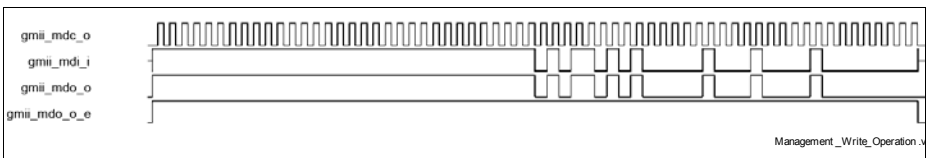


Figure 35-13 Management Write Operation

Figure 35-13 is a reference for the Write operation.

35.2.9.3 MII Management Read Operation

When the user sets the GMII Busy bit (see GMII Address Register, **“32-bit Register - GMII Address” on Page 1-309**) with the GMII Write bit as 0, the GMAC CSR module transfers the PHY address and the register address in PHY to the SMA to initiate a Read operation in the PHY registers. At this point, the SMA module starts a Read operation on the GMII Management Interface using the Management Frame Format specified in the GMII specifications (Section 22.2.4.5 of IEEE Standard). The application should not change the GMII Address register contents or the GMII Data register while the transaction is ongoing. Write operations to the GMII Address register or GMII Data Register during this period are ignored (the Busy bit is high) and the transaction completed without any error on the MCI interface.

After the Read operation has completed, the SMA indicates this to the CSR, which then resets the Busy bit and updates the GMII Data register with the data read from the PHY. The SMA module divides the CSR (Application) clock with the clock divider programmed (CR bits of GMII Address Register) to generate the MDC clock for this interface. The GMAC drives the MDIO line for the complete duration of the frame except during the Data fields when the PHY is driving the MDIO line. The frame format for the Read operation is as follows:

IDLE	PREAMBLE	START	OPCODE	PHY ADDR	REG ADDR	TA	DATA	IDLE
Z	1111...11	01	10	AAAAA	RRRRR	Z0	DDD... .DDD	Z

Figure 35-14 is a reference for the read operation.

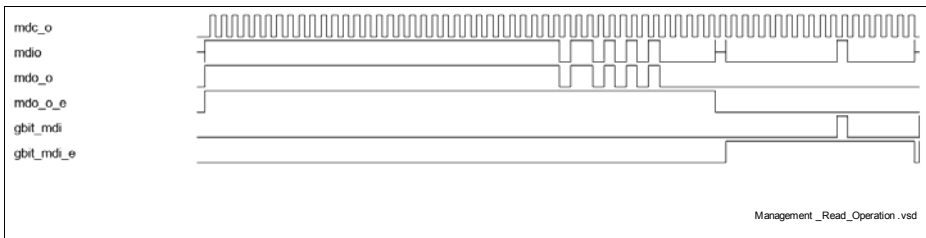


Figure 35-14 Management Read Operation

35.2.10 Reduced Media Independent Interface

The Reduced Media Independent Interface (RMII) specification reduces the pin count between Ethernet PHYs and Switch ASICs (only in 10/100 mode). According to the IEEE 802.3u standard, an MII contains 16 pins for data and control. In devices incorporating multiple MAC or PHY interfaces (such as switches), the number of pins adds significant cost with increase in port count. The RMII specification addresses this problem by reducing the pin count to 7 for each port — a 62.5% decrease in pin count.

- The RMII module is instantiated between the GMAC and the PHY. This helps translation of the MAC’s MII into the RMII. The RMII block has the following characteristics:
- Supports 10 Mbit/s and 100 Mbit/s operating rates.
- Two clock references are sourced externally, providing independent, 2-bit wide transmit and receive paths.

35.2.10.1 Block Diagram

Figure 35-15 shows the position of the RMII block relative to the Ethernet MAC and RMII PHY. The RMII block is placed in front of the Ethernet MAC to translate the MII signals to RMII signals.

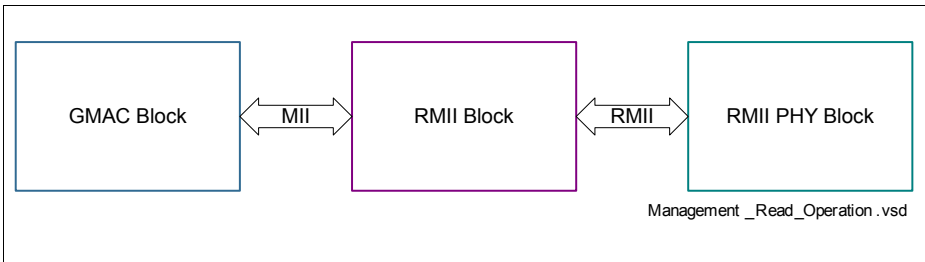


Figure 35-15 RMII Block Diagram

35.2.10.2 Block Overview

The following list describes the RMII’s hardware components, which are shown in [Figure 35-16](#). Each of these blocks is briefly described in the following sections.

MII-RMII Transmit (MRT) Block: This block translates all MII transmit signals to RMII transmit signals. All RMII signals are synchronous to `clk_rmii_i`.

MII-RMII Receive (MRR) Block: This block translates all RMII receive signals to MII receive signals. All MII signals are synchronous to `clk_rx_i`. You must ensure that the same clock is connected to both `clk_tx_i` and `clk_rx_i` clock input ports in RMII mode. This is required because clock-MUXing logic is avoided inside the GMAC core.

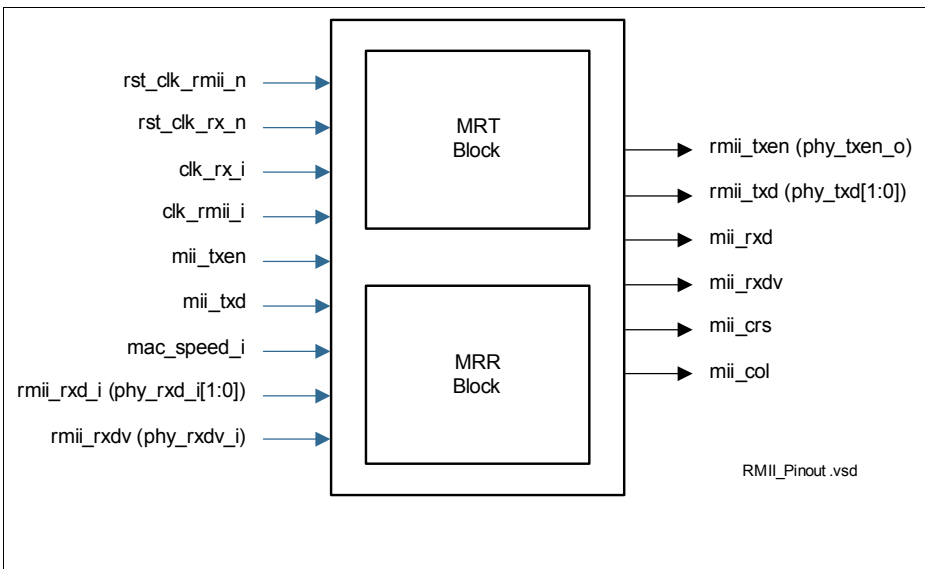


Figure 35-16 RMII Pinout

Ethernet MAC (ETH)

Note: The `mac_speed_i` signal configures the RMII to operate at 10 Mbit/s or 100 Mbit/s. This signal is driven from the MAC Configuration register's FES bit.

35.2.10.3 Transmit Bit Ordering

Each nibble from the MII must be transmitted on the RMII a di-bit at a time with the order of di-bit transmission shown in [Figure 35-17](#). The lower order bits (D1 and D0) are transmitted first followed by higher order bits (D2 and D3).

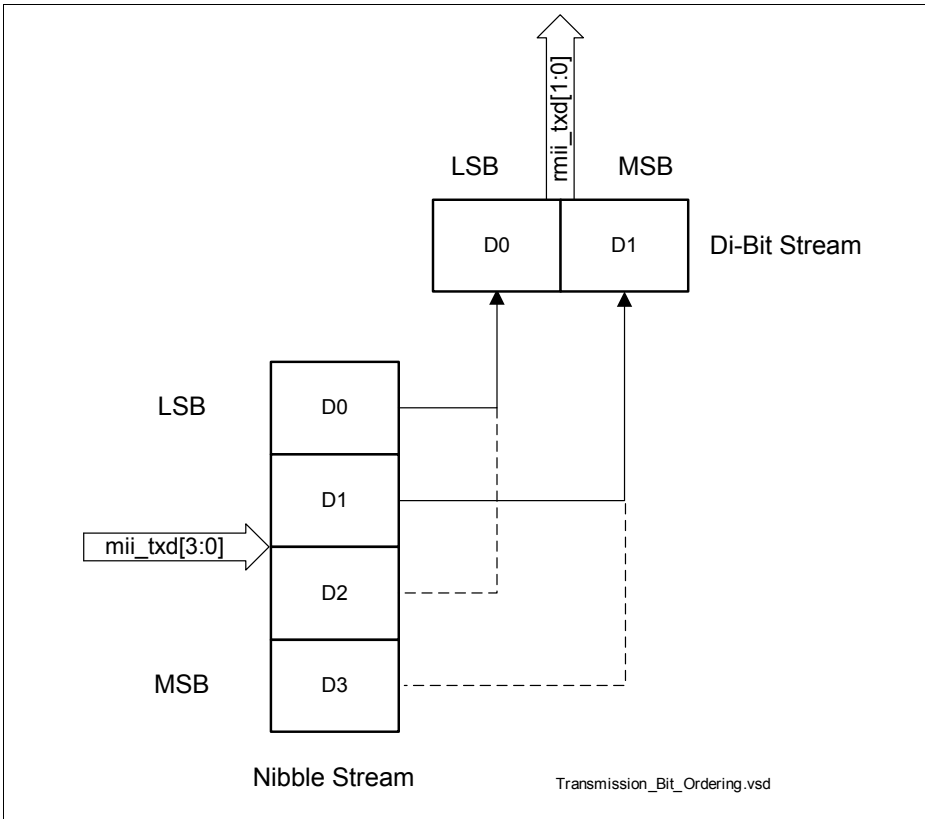


Figure 35-17 Transmission Bit Ordering

35.2.10.4 RMII Transmit Timing Diagrams

[Figure 35-18](#) through [Figure 35-21](#) show MII-to-RMII transaction timing.

Ethernet MAC (ETH)

Figure 35-18 shows the start of MII transmission and the following RMII transmission in 100 Mbit/s mode.

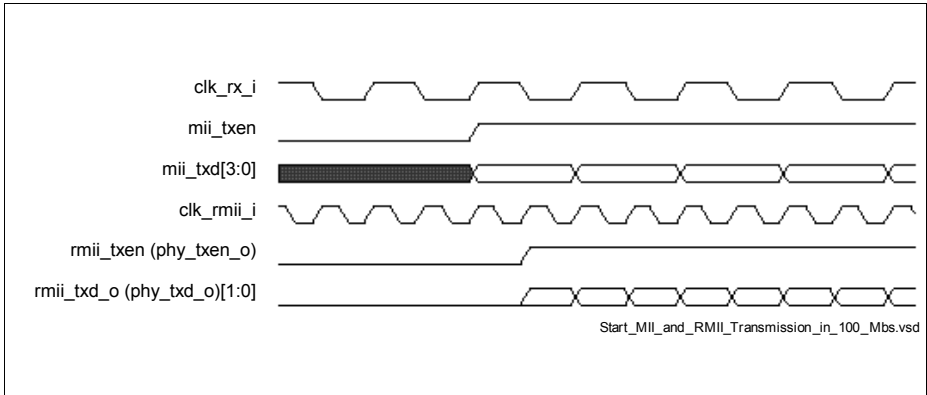


Figure 35-18 Start of MII and RMII Transmission in 100 Mbit/s Mode

Figure 35-19 shows the end of frame transmission for MII and RMII in 100 Mbit/s mode.

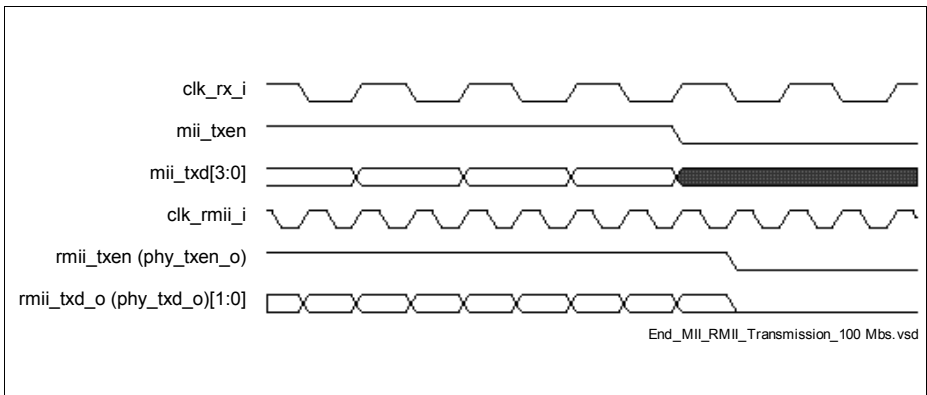


Figure 35-19 End of MII and RMII Transmission in 100 Mbit/s Mode

Figure 35-20 shows the start of MII transmission and the following RMII transmission in 10 Mbit/s mode.

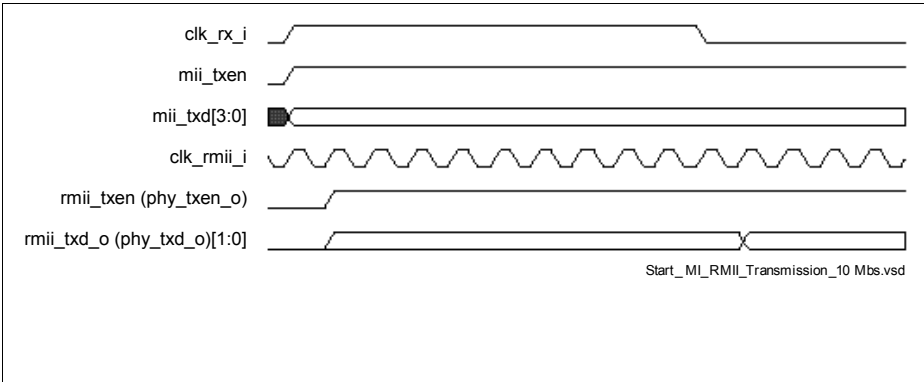


Figure 35-20 Start of MII and RMII Transmission in 10 Mbit/s Mode

Figure 35-21 shows the end of MII transmission and RMII transmission in 10 Mbit/s mode.

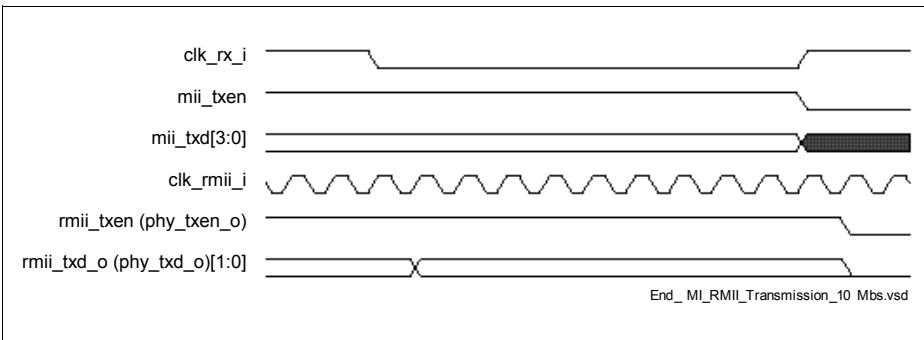


Figure 35-21 End of MII and RMII Transmission in 10 Mbit/s Mode

Receive Bit Ordering

Each nibble is transmitted to the MII from the di-bit received from the RMII in the nibble transmission order shown in **Figure 35-22**. The lower order bits (D0 and D1) are received first, followed by the higher order bits (D2 and D3).

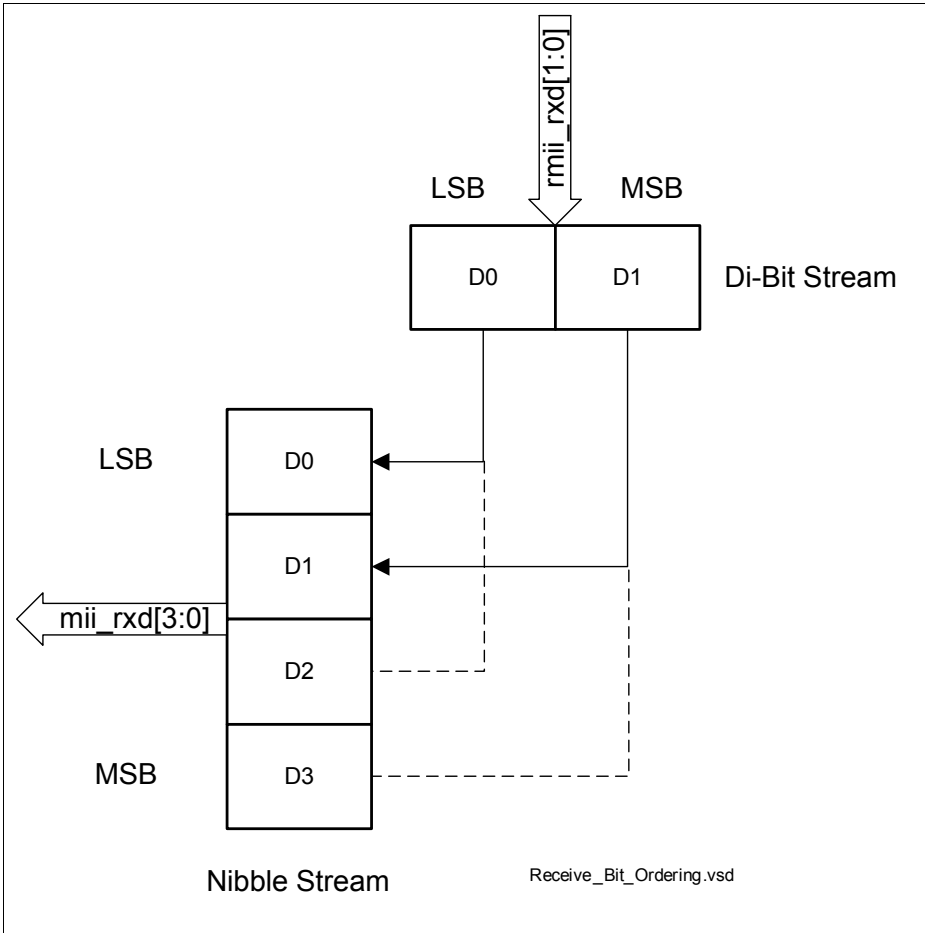


Figure 35-22 Receive Bit Ordering

35.2.11 Interrupts From the GMAC Core

Interrupts can be generated from the GMAC core as a result of various events in the optional modules in it.

The Interrupt Status register in the GMAC Register Map ([“GMAC Register Map” on Page 1-281](#)) describes the events that can cause an interrupt from the GMAC core. Each event can be prevented from asserting the interrupt by setting the corresponding mask bits in the Interrupt Mask register.

Ethernet MAC (ETH)

The interrupt register bits only indicate the block from which the event is reported. You must read the corresponding status registers and other registers to clear the interrupt. For example, Bit 0 of the Interrupt register, set high, indicates that the link status on the RGMII interface has changed. You must read Register 54 (the RGMII Status register) to clear this interrupt event.

The interrupts from the RGMII and PCS blocks are combined (OR'ed) and given as the GLI bit in the DMA Status register. The TTI, GPI, and GMI signals in [Figure 35-23](#) refer to the bits that can be read in the DMA Status register.

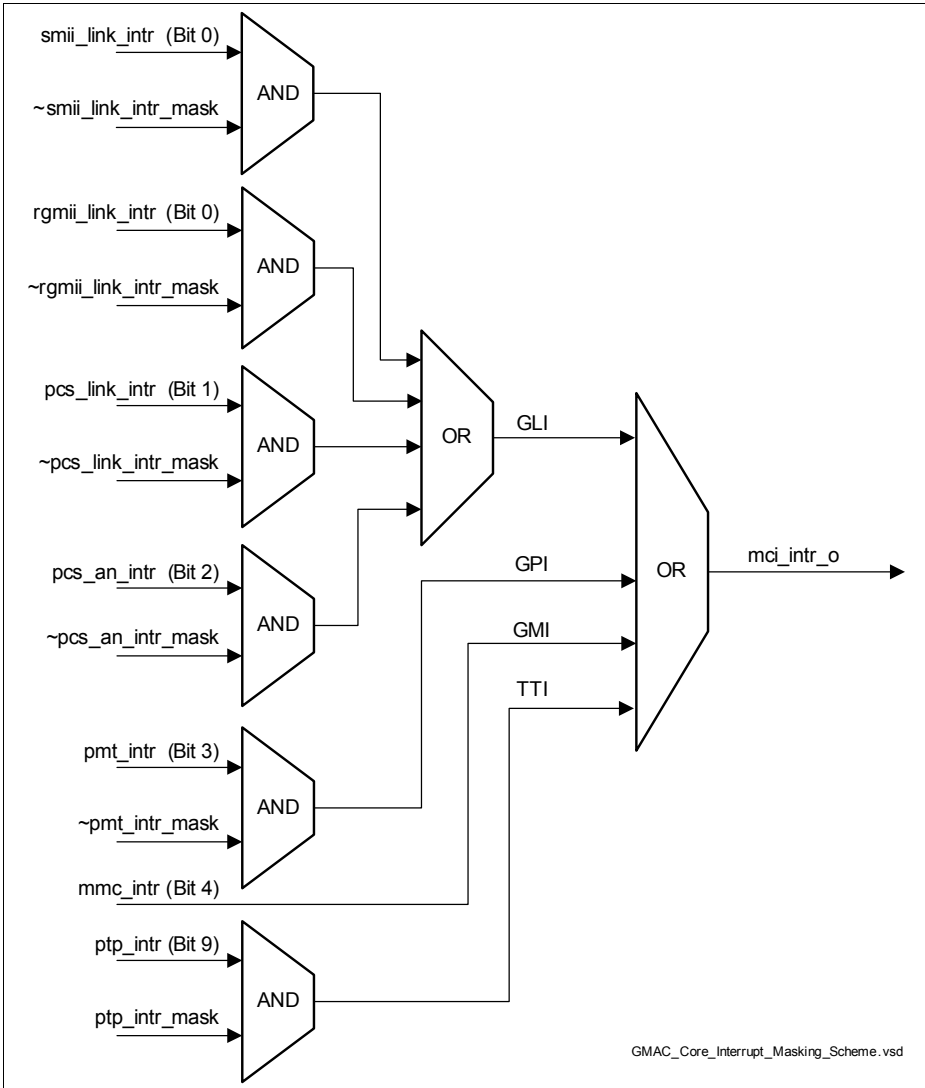


Figure 35-23 GMAC Core Interrupt Masking Scheme

35.3 Register

General notes on registers:

Ethernet MAC (ETH)

- All CSRs are implemented as 32-bit registers and can be accessed in 1 read/write cycle with a 32-bit MCI interface (or 32-bit AHB slave port). If you initiate a non-32-bit register access, the byte enable signals (mci_be_i[3:0]) qualify the corresponding register bytes. Only write operations are qualified with byte enables, while read operations are always 32-bit, unless the register is affected by a read operation. Byte enable signals qualify reads to such registers as, for example, the MMC counter registers.
- The transfer of particular register contents (e.g., MAC DA address register) to the Transmit or Receive clock domains are initiated when a particular byte is written. This is indicated in the register descriptions, where applicable.
- When any Register content is being transferred to a different clock domain after a write operation, there should not be any further writes to the same location until the first write is updated. Otherwise, the second write operation will not get updated to the destination clock domain. Thus the delay between two writes to the same register location should be at least 4 cycles of the destination clock (PHY receive clock or PHY transmit clock).

35.3.1 Register Maps

The register maps in this section provide high-level summaries of each register or group of registers. The detailed register descriptions in [“Register Description” on Page 1-292](#).

Reserved address space (marked as “do not use”) will generate no bus error (nBE).

All registers in the DMA and MAC Registers map (1000_H - 2058_H) are P = ACCEN protected. The registers for additional Module Control are ACCEN protected according to [Figure 35-22](#).

35.3.1.1 Register Overview

Table 35-19 Registers Address Space - ETH Module

Module	Base Address	End Address	Note
ETH	F001 D000 _H	F001 F0FF _H	-

35.3.1.2 DMA Register Map

Table 35-20 DMA Register Map

DMA Register No.	Offset Address	Register Name and Description
0	2000 _H	DMA Register 0 (Bus Mode Register) Controls the Host Interface Mode.
1	2004 _H	DMA Register 1 (Transmit Poll Demand Register) Used by the host to instruct the DMA to poll the Transmit Descriptor List.
2	2008 _H	DMA Register 2 (Receive Poll Demand Register) Used by the Host to instruct the DMA to poll the Receive Descriptor List.
3	200C _H	DMA Register 3 (Receive Descriptor List Address Register) Points the DMA to the start of the Receive Descriptor list.
4	2010 _H	DMA Register 4 (Transmit Descriptor List Address Register) Points the DMA to the start of the Transmit Descriptor List.
5	2014 _H	DMA Register 5 (Status Register) The Software driver (application) reads this register during interrupt service routine or polling to determine the status of the DMA.
6	2018 _H	DMA Register 6 (Operation Mode Register) Establishes the Receive and Transmit operating modes and command.
7	201C _H	DMA Register 7 (Interrupt Enable Register) Enables the interrupts reported by the Status Register.
8	2020 _H	DMA Register 8 (Missed Frame and Buffer Overflow Counter Register) Contains the counters for discarded frames because no host Receive Descriptor was available, and discarded frames because of Receive FIFO Overflow.
9	2024 _H	DMA Register 9 (Receive Interrupt Watchdog Timer Register) Watchdog time-out for Receive Interrupt (RI) from DMA
12-17	2030 _H - 2044 _H	Do not use

Table 35-20 DMA Register Map

DMA Register No.	Offset Address	Register Name and Description
18	2048 _H	DMA Register 18 (Current Host Transmit Descriptor Register) Points to the start of current Transmit Descriptor read by the DMA.
19	204C _H	DMA Register 19 (Current Host Receive Descriptor Register) Points to the start of current Receive Descriptor read by the DMA.
20	2050 _H	DMA Register 20 (Current Host Transmit Buffer Address Register) Points to the current Transmit Buffer address read by the DMA.
21	2054 _H	DMA Register 21 (Current Host Receive Buffer Address Register) Points to the current Receive Buffer address read by the DMA.
22	2058 _H	DMA Register 22 (HW Feature Register) Indicates the presence of the optional features of the core.
23-63	205C _H - 20FC _H	Do not use

35.3.1.3 GMAC Register Map

Table 1-59 provides the address map of the GMAC core registers.

Table 35-21 GMAC Register Map

GMAC Register No.	Offset Address	Register Name and Description
0	1000 _H	GMAC Register 0 (MAC Configuration Register) This is the operation mode register for the MAC.
1	1004 _H	GMAC Register 1 (MAC Frame Filter) Contains the frame filtering controls.

Table 35-21 GMAC Register Map

GMAC Register No.	Offset Address	Register Name and Description
2	1008 _H	GMAC Register 2 (Hash Table High Register) Contains the higher 32 bits of the Multicast Hash table. This register is present only when the Hash filter function is selected in coreConsultant. (See Table ??)
3	100C _H	GMAC Register 3 (Hash Table Low Register) Contains the lower 32 bits of the Multicast Hash table. This register is present only when the Hash filter function is selected in coreConsultant. (See Table ??)
4	1010 _H	GMAC Register 4 (GMII Address Register) Controls the management cycles to an external PHY. This register is present only when the Station Management (MDIO) feature is selected in coreConsultant. (See Table ??)
5	1014 _H	GMAC Register 5 (GMII Data Register) Contains the data to be written to or read from the PHY register. This register is present only when the Station Management (MDIO) feature is selected in coreConsultant. (See Table ??)
6	1018 _H	GMAC Register 6 (Flow Control Register) Controls the generation of control frames.
7	101C _H	GMAC Register 7 (VLAN Tag Register) Identifies IEEE 802.1Q VLAN type frames.
8	1020 _H	GMAC Register 8 (Version Register) Identifies the version of the Core
9	1024 _H	GMAC Register 9 (Debug Register) Gives the status of various internal blocks for debugging

Table 35-21 GMAC Register Map

GMAC Register No.	Offset Address	Register Name and Description
10	1028 _H	<p>Remote Wake-Up Frame Filter</p> <p>This is the address through which the remote Wake-up Frame Filter registers (wkupfilter_reg) are written/read by the Application. wkupfilter_reg is actually a pointer to eight (not transparent) such wkupfilter_reg registers. Eight sequential Writes to this address (028) will write all wkupfilter_reg registers. Eight sequential Reads from this address (028) will read all wkupfilter_reg registers.</p> <p>This register contains the higher 16 bits of the 7th MAC address.</p> <p>This register is present only when the PMT module Remote Wake-up feature is selected in coreConsultant. (See Table ??) Refer to “Remote Wake-Up Frame Filter Register” on page ?? for additional information.</p>
11	102C _H	<p>PMT Control and Status</p> <p>This register is present only when the PMT module is selected in coreConsultant. (See Table ??) Refer to “PMT Control and Status Register” on page ?? for additional information.</p>
12-13	1030 _H - 1034 _H	Do not use
14	1038 _H	<p>GMAC Register 14 (Interrupt Status Register)</p> <p>Contains the interrupt status.</p>
15	103C _H	<p>GMAC Register 15 (Interrupt Mask Register)</p> <p>Contains the masks for generating the interrupts.</p>
16	1040 _H	<p>GMAC Register 16 (MAC Address 0 High Register)</p> <p>Contains the higher 16 bits of the first MAC address.</p>
17	1044 _H	<p>GMAC Register 17 (MAC Address 0 Low Register)</p> <p>Contains the lower 32 bits of the first MAC address.</p>
18	1048 _H	<p>GMAC Register 18 (MAC Address 1 High Register)</p> <p>Contains the higher 16 bits of the second MAC address. This register is present only when Enable MAC Address 1 is selected in coreConsultant. (See Table ??).</p>

Table 35-21 GMAC Register Map

GMAC Register No.	Offset Address	Register Name and Description
19	104C _H	GMAC Register 19 (MAC Address 1 Low Register) Contains the lower 32 bits of the second MAC address. This register is present only when Enable MAC Address 1 is selected in coreConsultant. (See Table ??).
20	1050 _H	MAC Address 2 High Register Contains the higher 32 bits of the third MAC address. This register is present only when Enable MAC Address 2 is selected in coreConsultant. (See Table ??).
21	1054 _H	MAC Address 2 Low Register Contains the lower 32 bits of the third MAC address. This register is present only when Enable MAC Address 2 is selected in coreConsultant. (See Table ??).
22	1058 _H	MAC Address 3 High Register Contains the higher 16 bits of the fourth MAC address. This register is present only when Enable MAC Address 3 is selected in coreConsultant. (See Table ??).
23	105C _H	MAC Address 3 Low Register Contains the lower 32 bits of the fourth MAC address. This register is present only when Enable MAC Address 3 is selected in coreConsultant. (See Table ??).
55-63	10DC _H - 10FC _H	Do not use
64-191	1100 _H - 12FC _H	MMC Register Map See Table 1-17 “MMC Register Map” on Page 1-131 .
192-447	1300 _H - 16FC _H	Do not use
448	1700 _H	Register 448 (Time Stamp Control Register) Controls the time stamp generation and update logic.
449	1704 _H	Register 449 (Sub-Second Increment Register) Contains the 8-bit value by which the Sub-Second register is incremented. This register is only present when IEEE1588 time stamping is enabled without an external time stamp input.

Table 35-21 GMAC Register Map

GMAC Register No.	Offset Address	Register Name and Description
450	1708 _H	Register 450 (System Time - Seconds Register) Contains the lower 32 bits of the seconds field of the system time. This register is only present when IEEE1588 time stamping is enabled without an external time stamp input.
451	170C _H	Register 451 (System Time - Nanoseconds Register) Contains 32 bits of the nanoseconds field of the system time. This register is only present when IEEE1588 time stamping is enabled without an external time stamp input.
452	1710 _H	Register 452 (System Time - Seconds Update Register) Contains the lower 32 bits of the seconds field to be written to, added to, or subtracted from the System Time value. This register is only present when IEEE1588 time stamping is enabled without an external time stamp input.
453	1714 _H	Register 453 (System Time - Nanoseconds Update Register) Contains 32 bits of the nanoseconds field to be written to, added to, or subtracted from the System Time value. This register is only present when IEEE1588 time stamping is enabled without an external time stamp input.
454	1718 _H	Register 454 (Time Stamp Addend Register) This register is used by the software to readjust the clock frequency linearly to match the master clock frequency. This register is only present when IEEE1588 time stamping is enabled without an external time stamp input.
455	171C _H	Register 455 (Target Time Seconds Register) Contains the higher 32 bits of time to be compared with the system time for interrupt event generation. This register is only present when IEEE1588 time stamping is enabled without an external time stamp input.
456	1720 _H	Register 456 (Target Time Nanoseconds Register) Contains the lower 32 bits of time to be compared with the system time for interrupt event generation. This register is only present when IEEE1588 time stamping is enabled without an external time stamp input.

Table 35-21 GMAC Register Map

GMAC Register No.	Offset Address	Register Name and Description
457	1724 _H	Register 457 (System Time - Higher Word Seconds Register) Contains the most significant 16-bits of the time stamp seconds value. This register is optional and can be selected using the coreKit parameter identified in "IEEE 1588 Time Stamp Block" on page ??.
458	1728 _H	Register 458 (Time Stamp Status Register) Contains the PTP status. This register is available only when the advanced IEEE 1588 timestamp feature is selected.
459	172C _H	Register 459 (PPS Control Register) This register is used to control the interval of the PPS signal output, such that a duration of less than 1 second can be achieved between pulses.
460	1730 _H	Register 460 (Auxiliary Time Stamp - Nanoseconds Register) Contains the lower 32 bits (nanoseconds field) of the auxiliary time stamp register.
461	1734 _H	Register 461 (Auxiliary Time Stamp - Seconds Register) Contains the upper 32 bits (seconds field) of the auxiliary time stamp register.
462-511	1738 _H - 17FC _H	Do not use
512-511	1738 _H - 17FC _H	Do not use

35.3.1.4 Ethernet MAC Additional Module Control Registers

This section describes the additional module control registers of the Ethernet MAC module. All Ethernet MAC register names described in this section will be referenced in other parts of the TC26x B-Step User's Manual by the module name prefix "Ethernet MAC_" for the Ethernet MAC interface.

All registers in the Ethernet MAC address spaces are reset with the application reset (definition see SCU section "Reset Operation"). P = ACCEN protection

Table 35-22 Register Overview - ETH Add. Mod. Contr. Registers

Register Short Name	Register Long Name	Offset ¹⁾	Access Mode		Description see
			Read	Write	
Module Control Registers					
ETH_CLC	Clock Control Register	0000 _H	SV, U	SV, E, P	Page 35-46 2
ETH_ID	Module Identification Register	0004 _H	SV, U	nBE	Page 35-46 1
ETH_GPCTL	General Purpose Control Register	0008 _H	SV, U	SV	Page 35-46 3
BPI Kernel Registers					
ETH_ACCEN0	Access Enable Register 0	000C _H	U, SV	SV, SE	Page 35-46 5
ETH_ACCEN1	Access Enable Register 1	0010 _H	U, SV	SV, SE	Page 35-46 6
ETH_KRST0	Reset Control Register 0	0014 _H	U, SV	SV, P	Page 35-46 7
ETH_KRST1	Reset Control Register 1	0018 _H	U, SV	SV, P	Page 35-46 8
ETH_KRSTCLR	Reset Status Clear Reg.	001C _H	U, SV	SV, P	Page 35-46 9
-	reserved	0020 _H - 0FFF _H	nBE	nBE	

1) The absolute register address is calculated as follows:
Module Base Address ([Table 1-60](#)) + Offset Address (shown in this column)

35.3.2 Register Description

This section defines the bits for each register.

32-bit Register - MAC_Configuration

The MAC Configuration register establishes receive and transmit operating modes.

ETH_MAC_CONFIGURATION

Register 0 - MAC Configuration Register (1000_H)

Reset Value: 0000 8000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES ERV ED_ 31	SARC			TWO KPE	SFT ERR	CST	TC	WD	JD	BE	JE	IFG			DCR S
r	r			rw	r	rw	r	rw	rw	r	rw	rw			rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS	FES	DO	LM	DM	IPC	DR	LUD	ACS	BL		DC	TE	RE	PRELEN	
r	rw	rw	rw	rw	rw	rw	r	rw	rw		rw	rw	rw	rw	rw

Field	Bits	Type	Description
PRELEN	[1:0]	rw	<p>Preamble Length for Transmit Frames</p> <p>These bits control the number of preamble bytes that are added to the beginning of every Transmit frame. The preamble reduction occurs only when the MAC is operating in the full-duplex mode.</p> <ul style="list-style-type: none"> * 2'b00: 7 bytes of preamble * 2'b01: 5 byte of preamble * 2'b10: 3 bytes of preamble * 2'b11: reserved
RE	2	rw	<p>Receiver Enable</p> <p>When this bit is set, the receiver state machine of the MAC is enabled for receiving frames from the GMII or MII. When this bit is reset, the MAC receive state machine is disabled after the completion of the reception of the current frame, and does not receive any further frames from the GMII or MII.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
TE	3	rw	<p>Transmitter Enable</p> <p>When this bit is set, the transmit state machine of the MAC is enabled for transmission on the GMII or MII. When this bit is reset, the MAC transmit state machine is disabled after the completion of the transmission of the current frame, and does not transmit any further frames.</p>
DC	4	rw	<p>Deferral Check</p> <p>When this bit is set, the deferral check function is enabled in the MAC. The MAC issues a Frame Abort status, along with the excessive deferral error bit set in the transmit frame status, when the transmit state machine is deferred for more than 24,288 bit times in the 10 or 100 Mbps mode. If the MAC is configured for 1000 Mbps operation, or if the Jumbo frame mode is enabled in the 10 or 100 Mbps mode, the threshold for deferral is 155,680 bits times. Deferral begins when the transmitter is ready to transmit, but is prevented because of an active carrier sense signal (CRS) on GMII or MII. Deferral time is not cumulative. When the transmitter defers for 10,000 bit times, it transmits, collides, backs off, and then defers again after completion of back-off. The deferral timer resets to 0 and restarts.</p> <p>When this bit is reset, the deferral check function is disabled and the MAC defers until the CRS signal goes inactive. This bit is applicable only in the half-duplex mode and is reserved (RO) in the full-duplex-only configuration.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
BL	[6:5]	rw	<p>Back-Off Limit</p> <p>The Back-Off limit determines the random integer number (r) of slot time delays (4,096 bit times for 1000 Mbps and 512 bit times for 10/100 Mbps) for which the MAC waits before rescheduling a transmission attempt during retries after a collision. This bit is applicable only in the half-duplex mode and is reserved (RO) in the full-duplex-only configuration.</p> <p>* 00: $k = \min(n, 10)$ * 01: $k = \min(n, 8)$ * 10: $k = \min(n, 4)$ * 11: $k = \min(n, 1)$</p> <p>where n = retransmission attempt. The random integer r takes the value in the range $0 \leq r < k$th power of 2</p>
ACS	7	rw	<p>Automatic Pad or CRC Stripping</p> <p>When this bit is set, the MAC strips the Pad or FCS field on the incoming frames only if the value of the length field is less than 1,536 bytes. All received frames with length field greater than or equal to 1,536 bytes are passed to the application without stripping the Pad or FCS field.</p> <p>When this bit is reset, the MAC passes all incoming frames, without modifying them, to the Host.</p>
LUD	8	r	<p>Link Up or Down</p> <p>This bit indicates whether the link is up or down during the transmission of configuration in the RGMII, SGMII, or SMII interface:</p> <p>* 0: Link Down * 1: Link Up</p> <p>This bit is reserved (RO with default value) and is enabled when the RGMII, SGMII, or SMII interface is enabled during core configuration.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
DR	9	rw	<p>Disable Retry</p> <p>When this bit is set, the MAC attempts only one transmission. When a collision occurs on the GMII or MII interface, the MAC ignores the current frame transmission and reports a Frame Abort with excessive collision error in the transmit frame status.</p> <p>When this bit is reset, the MAC attempts retries based on the settings of the BL field (Bits [6:5]). This bit is applicable only in the half-duplex mode and is reserved (RO with default value) in the full-duplex-only configuration.</p>
IPC	10	rw	<p>Checksum Offload</p> <p>When this bit is set, the MAC calculates the 16-bit one's complement of the one's complement sum of all received Ethernet frame payloads. It also checks whether the IPv4 Header checksum (assumed to be bytes 2526 or 2930 (VLAN-tagged) of the received Ethernet frame) is correct for the received frame and gives the status in the receive status word. The MAC also appends the 16-bit checksum calculated for the IP header datagram payload (bytes after the IPv4 header) and appends it to the Ethernet frame transferred to the application (when Type 2 COE is deselected).</p> <p>When this bit is reset, this function is disabled.</p> <p>When Type 2 COE is selected, this bit, when set, enables the IPv4 header checksum checking and IPv4 or IPv6 TCP, UDP, or ICMP payload checksum checking. When this bit is reset, the COE function in the receiver is disabled and the corresponding PCE and IP HCE status bits are always cleared.</p> <p>If the IP Checksum Offload feature is not enabled during core configuration, this bit is reserved (RO with default value).</p>
DM	11	rw	<p>Duplex Mode</p> <p>When this bit is set, the MAC operates in the full-duplex mode where it can transmit and receive simultaneously. This bit is RO with default value of 1'b1 in the full-duplex-only configuration.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
LM	12	rw	<p>Loopback Mode</p> <p>When this bit is set, the MAC operates in the loopback mode at GMII or MII. The (G)MII Receive clock input (clk_rx_i) is required for the loopback to work properly, because the Transmit clock is not looped-back internally.</p>
DO	13	rw	<p>Disable Receive Own</p> <p>When this bit is set, the MAC disables the reception of frames when the gmii_txen_o is asserted in the half-duplex mode.</p> <p>When this bit is reset, the MAC receives all packets that are given by the PHY while transmitting.</p> <p>This bit is not applicable if the MAC is operating in the full-duplex mode. This bit is reserved (RO with default value) if the MAC is configured for the full-duplex-only operation.</p>
FES	14	rw	<p>Speed</p> <p>This bit selects the speed in the MII, RMII, SMII, RGMII, SGMII, or RevMII interface:</p> <ul style="list-style-type: none"> * 0: 10 Mbps * 1: 100 Mbps <p>This bit is reserved (RO) by default and is enabled only when the RMII, SMII, RGMII, SGMII, or RevMII interface is enabled during core configuration. This bit generates link speed encoding when TC (Bit 24) is set in the RGMII, SMII, or SGMII mode.</p> <p>This bit is always driven for RevMII. If the MAC is configured with an RMII, SMII, SGMII, or RGMII PHY interface, this bit can optionally be driven as an output signal (mac_speed_o[0]) to reflect the value of this bit in the mac_speed_o signal.</p> <p>In addition, this bit is reserved when RMII is enabled without enabling the <i>Output Port for speed selection</i> option during core configuration.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
PS	15	r	<p>Port Select</p> <p>This bit selects between GMII and MII: * 0: GMII (1000 Mbps) * 1: MII (10/100 Mbps)</p> <p>In the 10 or 100 Mbps-only (always 1) or 1000 Mbps-only (always 0) configurations, this bit is read-only with the appropriate value. In the default 10/100/1000 Mbps configurations, this bit is R_W. The mac_portselect_o signal reflects the value of this bit.</p>
DCRS	16	rw	<p>Disable Carrier Sense During Transmission</p> <p>When set high, this bit makes the MAC transmitter ignore the (G)MII CRS signal during frame transmission in the half-duplex mode. This request results in no errors generated because of Loss of Carrier or No Carrier during such transmission. When this bit is low, the MAC transmitter generates such errors because of Carrier Sense and can even abort the transmissions. This bit is reserved (and RO) in the full-duplex-only configurations.</p>
IFG	[19:17]	rw	<p>Inter-Frame Gap</p> <p>These bits control the minimum IFG between frames during transmission.</p> <ul style="list-style-type: none"> * 000: 96 bit times * 001: 88 bit times * 010: 80 bit times * ... * 111: 40 bit times <p>In the half-duplex mode, the minimum IFG can be configured only for 64 bit times (IFG = 100). Lower values are not considered. In the 1000-Mbps mode, the minimum IFG supported is 64 bit times (and above) in the GMAC-CORE configuration and 80 bit times (and above) in other configurations.</p>
JE	20	rw	<p>Jumbo Frame Enable</p> <p>When this bit is set, the MAC allows Jumbo frames of 9,018 bytes (9,022 bytes for VLAN tagged frames) without reporting a giant frame error in the receive frame status.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
BE	21	r	Frame Burst Enable When this bit is set, the MAC allows frame bursting during transmission in the GMII half-duplex mode. This bit is reserved (and RO) in the 10/100 Mbps only or full-duplex-only configurations.
JD	22	rw	Jabber Disable When this bit is set, the MAC disables the jabber timer on the transmitter. The MAC can transfer frames of up to 16,384 bytes. When this bit is reset, the MAC cuts off the transmitter if the application sends out more than 2,048 bytes of data (10,240 if JE is set high) during transmission.
WD	23	rw	Watchdog Disable When this bit is set, the MAC disables the watchdog timer on the receiver. The MAC can receive frames of up to 16,384 bytes. When this bit is reset, the MAC does not allow more than 2,048 bytes (10,240 if JE is set high) of the frame being received. The MAC cuts off any bytes received after 2,048 bytes.
TC	24	r	Transmit Configuration in RGMII, SGMII, or SMII When set, this bit enables the transmission of duplex mode, link speed, and link up or down information to the PHY in the RGMII, SMII, or SGMII port. When this bit is reset, no such information is driven to the PHY. This bit is reserved (and RO) if the RGMII, SMII, or SGMII PHY port is not selected during core configuration.
CST	25	rw	CRC Stripping of Type Frames When set, the last 4 bytes (FCS) of all frames of Ether type (type field greater than 0x0600) are stripped and dropped before forwarding the frame to the application. This function is not valid when the IP Checksum Engine (Type 1) is enabled in the MAC receiver.
SFTERR	26	r	SMII Force Transmit Error When set, this bit indicates to the PHY to force a transmit error in the SMII frame being transmitted. This bit is reserved if the SMII PHY port is not selected during core configuration.

Ethernet MAC (ETH)

Field	Bits	Type	Description
TWOKPE	27	rw	<p>IEEE 802.3as support for 2K packets Enable</p> <p>When set, the MAC considers all frames, with up to 2,000 bytes length, as normal packets. When Bit 20 (Jumbo Enable) is not set, the MAC considers all received frames of size more than 2K bytes as Giant frames.</p> <p>When this bit is reset and Bit 20 (Jumbo Enable) is not set, the MAC considers all received frames of size more than 1,518 bytes (1,522 bytes for tagged) as Giant frames.</p> <p>When Bit 20 (Jumbo Enable) is set, setting this bit has no effect on Giant Frame status.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
SARC	[30:28]	r	<p>Source Address Insertion or Replacement Control</p> <p>This field controls the source address insertion or replacement for all transmitted frames. Bit 30 specifies which MAC Address register (0 or 1) is used for source address insertion or replacement based on the values of Bits [29:28]:</p> <ul style="list-style-type: none"> * 2'b0x: The input signals mti_sa_ctrl_i and ati_sa_ctrl_i control the SA field generation. * 2'b10: <ul style="list-style-type: none"> - If Bit 30 is set to 0, the MAC inserts the content of the MAC Address 0 registers (registers 16 and 17) in the SA field of all transmitted frames. - If Bit 30 is set to 1 and the Enable MAC Address Register 1 option is selected during core configuration, the MAC inserts the content of the MAC Address 1 registers (registers 18 and 19) in the SA field of all transmitted frames. * 2'b11: <ul style="list-style-type: none"> - If Bit 30 is set to 0, the MAC replaces the content of the MAC Address 0 registers (registers 16 and 17) in the SA field of all transmitted frames. - If Bit 30 is set to 1 and the Enable MAC Address Register 1 option is selected during core configuration, the MAC replaces the content of the MAC Address 1 registers (registers 18 and 19) in the SA field of all transmitted frames. <p>Note:</p> <ul style="list-style-type: none"> - Changes to this field take effect only on the start of a frame. If you write this register field when a frame is being transmitted, only the subsequent frame can use the updated value, that is, the current frame does not use the updated value. - These bits are reserved and RO when the Enable SA, VLAN, and CRC Insertion on TX feature is not selected during core configuration.
RESERVE D_31	31	r	RESERVED_31

32-bit Register - MAC_Frame_Filter

The MAC Frame Filter register contains the filter controls for receiving frames. Some of the controls from this register go to the address check block of the MAC, which performs the first level of address filtering. The second level of filtering is performed on the incoming frame, based on other controls such as Pass Bad Frames and Pass Control Frames.

ETH_MAC_FRAME_FILTER
Register 1 - MAC Frame Filter
(1004_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
RA	RESERVED_30_22										DNT U	IPFE	RESERVED_19_17			VTF E
rw	r										r	r	r			rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RESERVED_15_11					HPF	SAF	SAIF	PCF	DBF	PM	DAIF	HMC	HUC	PR		
r					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		

Field	Bits	Type	Description
PR	0	rw	Promiscuous Mode When this bit is set, the Address Filter module passes all incoming frames regardless of its destination or source address. The SA or DA Filter Fails status bits of the Receive Status Word are always cleared when PR is set.
HUC	1	rw	Hash Unicast When set, MAC performs destination address filtering of unicast frames according to the hash table. When reset, the MAC performs a perfect destination address filtering for unicast frames, that is, it compares the DA field with the values programmed in DA registers. If Hash Filter is not selected during core configuration, this bit is reserved (and RO).

Ethernet MAC (ETH)

Field	Bits	Type	Description
HMC	2	rw	<p>Hash Multicast</p> <p>When set, MAC performs destination address filtering of received multicast frames according to the hash table. When reset, the MAC performs a perfect destination address filtering for multicast frames, that is, it compares the DA field with the values programmed in DA registers. If Hash Filter is not selected during core configuration, this bit is reserved (and RO).</p>
DAIF	3	rw	<p>DA Inverse Filtering</p> <p>When this bit is set, the Address Check block operates in inverse filtering mode for the DA address comparison for both unicast and multicast frames. When reset, normal filtering of frames is performed.</p>
PM	4	rw	<p>Pass All Multicast</p> <p>When set, this bit indicates that all received frames with a multicast destination address (first bit in the destination address field is '1') are passed. When reset, filtering of multicast frame depends on HMC bit.</p>
DBF	5	rw	<p>Disable Broadcast Frames</p> <p>When this bit is set, the AFM module filters all incoming broadcast frames. In addition, it overrides all other filter settings. When this bit is reset, the AFM module passes all received broadcast frames.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
PCF	[7:6]	rw	<p>Pass Control Frames</p> <p>These bits control the forwarding of all control frames (including unicast and multicast PAUSE frames).</p> <ul style="list-style-type: none"> * 00: MAC filters all control frames from reaching the application. * 01: MAC forwards all control frames except PAUSE control frames to application even if they fail the Address filter. * 10: MAC forwards all control frames to application even if they fail the Address Filter. * 11: MAC forwards control frames that pass the Address Filter. <p>The following conditions should be true for the PAUSE control frames processing:</p> <ul style="list-style-type: none"> * Condition 1: The MAC is in the full-duplex mode and flow control is enabled by setting Bit 2 (RFE) of Register 6 (Flow Control Register) to 1. * Condition 2: The destination address (DA) of the received frame matches the special multicast address or the MAC Address 0 when Bit 3 (UP) of the Register 6 (Flow Control Register) is set. * Condition 3: The Type field of the received frame is 0x8808 and the OPCODE field is 0x0001. <p>Note:</p> <p>This field should be set to 01 only when the Condition 1 is true, that is, the MAC is programmed to operate in the full-duplex mode and the RFE bit is enabled. Otherwise, the PAUSE frame filtering may be inconsistent. When Condition 1 is false, the PAUSE frames are considered as generic control frames. Therefore, to pass all control frames (including PAUSE control frames) when the full-duplex mode and flow control is not enabled, you should set the PCF field to 10 or 11 (as required by the application).</p>

Ethernet MAC (ETH)

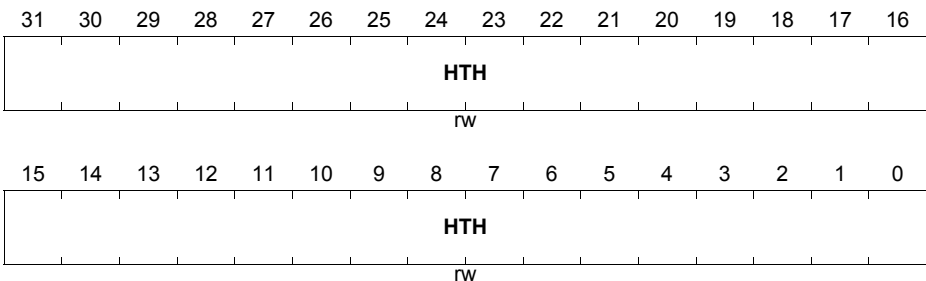
Field	Bits	Type	Description
SAIF	8	rw	<p>SA Inverse Filtering</p> <p>When this bit is set, the Address Check block operates in inverse filtering mode for the SA address comparison. The frames whose SA matches the SA registers are marked as failing the SA Address filter.</p> <p>When this bit is reset, frames whose SA does not match the SA registers are marked as failing the SA Address filter.</p>
SAF	9	rw	<p>Source Address Filter Enable</p> <p>When this bit is set, the MAC compares the SA field of the received frames with the values programmed in the enabled SA registers. If the comparison matches, then the SA Match bit of RxStatus Word is set high. When this bit is set high and the SA filter fails, the MAC drops the frame.</p> <p>When this bit is reset, the MAC forwards the received frame to the application and with the updated SA Match bit of the RxStatus depending on the SA address comparison.</p>
HPF	10	rw	<p>Hash or Perfect Filter</p> <p>When this bit is set, it configures the address filter to pass a frame if it matches either the perfect filtering or the hash filtering as set by the HMC or HUC bits.</p> <p>When this bit is low and the HUC or HMC bit is set, the frame is passed only if it matches the Hash filter. This bit is reserved (and RO) if the Hash filter is not selected during core configuration.</p>
RESERVE D_15_11	[15:11]	r	RESERVED_15_11
VTFE	16	rw	<p>VLAN Tag Filter Enable</p> <p>When set, this bit enables the MAC to drop VLAN tagged frames that do not match the VLAN Tag comparison.</p> <p>When reset, the MAC forwards all frames irrespective of the match status of the VLAN Tag.</p>
RESERVE D_19_17	[19:17]	r	RESERVED_19_17

Ethernet MAC (ETH)

Field	Bits	Type	Description
IPFE	20	r	<p>Layer 3 and Layer 4 Filter Enable</p> <p>When set, this bit enables the MAC to drop frames that do not match the enabled Layer 3 and Layer 4 filters. If Layer 3 or Layer 4 filters are not enabled for matching, this bit does not have any effect.</p> <p>When reset, the MAC forwards all frames irrespective of the match status of the Layer 3 and Layer 4 fields.</p> <p>If the Layer 3 and Layer 4 Filtering feature is not selected during core configuration, this bit is reserved (RO with default value).</p>
DNTU	21	r	<p>Drop non-TCP/UDP over IP Frames</p> <p>When set, this bit enables the MAC to drop the non-TCP or UDP over IP frames. The MAC forward only those frames that are processed by the Layer 4 filter.</p> <p>When reset, this bit enables the MAC to forward all non-TCP or UDP over IP frames.</p> <p>If the Layer 3 and Layer 4 Filtering feature is not selected during core configuration, this bit is reserved (RO with default value).</p>
RESERVE D_30_22	[30:22]	r	RESERVED_30_22
RA	31	rw	<p>Receive All</p> <p>When this bit is set, the MAC Receiver module passes all received frames, irrespective of whether they pass the address filter or not, to the Application. The result of the SA or DA filtering is updated (pass or fail) in the corresponding bits in the Receive Status Word.</p> <p>When this bit is reset, the Receiver module passes only those frames to the Application that pass the SA or DA address filter.</p>

32-bit Register - Hash_Table_High

The 64-bit Hash table is used for group address filtering. For hash filtering, the contents of the destination address in the incoming frame is passed through the CRC logic, and the upper 6 bits of the CRC register are used to index the contents of the Hash table. The most significant bit determines the register to be used (Hash Table High or Hash Table Low), and the other 5 bits determine which bit within the register. A hash value of 5b'00000 selects Bit 0 of the selected register, and a value of 5b'11111 selects Bit 31 of the selected register. The hash value of the destination address is calculated in the following way: 1. Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
 2. Perform bitwise reversal for the value obtained in Step 1.
 3. Take the upper 6 bits from the value obtained in Step 2.
 For example, if the DA of the incoming frame is received as 0x1F52419CB6AF (0x1F is the first byte received on GMII interface), then the internally calculated 6-bit Hash value is 0x2C and Bit 12 of Hash Table High register is checked for filtering. If the DA of the incoming frame is received as 0xA00A98000045, then the calculated 6-bit Hash value is 0x07 and Bit 7 of Hash Table Low register is checked for filtering. Note: To help you program the hash table, a sample C routine that generates a DA's 6-bit hash is included in the /sample_codes/ directory of your workspace. If the corresponding bit value of the register is 1'b1, the frame is accepted. Otherwise, it is rejected. If the PM (Pass All Multicast) bit is set in Register 1, then all multicast frames are accepted regardless of the multicast hash values. If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table High or Low registers are written. Consecutive writes to these register should be performed only after at least four clock cycles in the destination clock domain when double-synchronization is enabled. The Hash Table High register contains the higher 32 bits of the Hash table.

ETH_HASH_TABLE_HIGH
Register 2 - Hash Table High Register (1008_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
HTH	[31:0]	rw	Hash Table High This field contains the upper 32 bits of the Hash table.

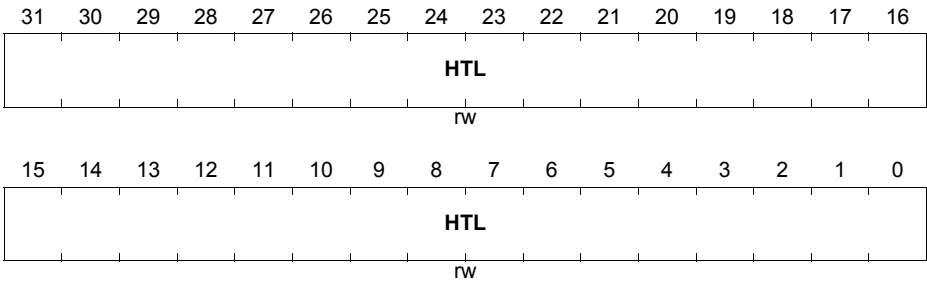
32-bit Register - Hash_Table_Low

The Hash Table Low register contains the lower 32 bits of the Hash table. Both Register 2 and Register 3 are reserved if the Hash Filter Function is disabled or the 128-bit or 256-bit Hash Table is selected during core configuration.

ETH_HASH_TABLE_LOW

Register 3 - Hash Table Low Register (100C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
HTL	[31:0]	rw	Hash Table Low This field contains the lower 32 bits of the Hash table.

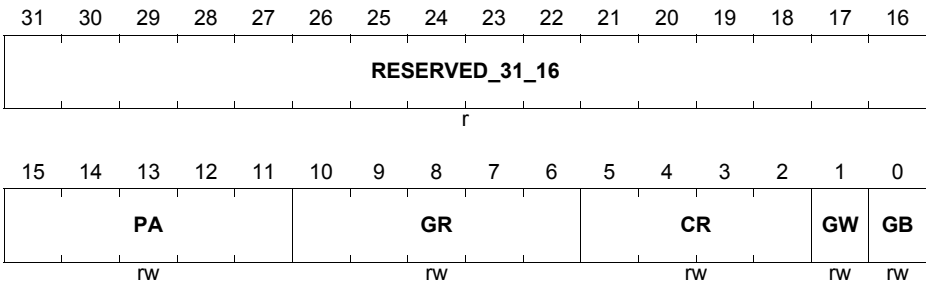
32-bit Register - GMII_Address

The GMII Address register controls the management cycles to the external PHY through the management interface.

ETH_GMII_ADDRESS

Register 4 - GMII Address Register (1010_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
GB	0	rw	<p>GMII Busy</p> <p>This bit should read logic 0 before writing to Register 4 and Register 5. During a PHY or RevMII register access, the software sets this bit to 1'b1 to indicate that a Read or Write access is in progress.</p> <p>The Register 5 is invalid until this bit is cleared by the MAC. Therefore, Register 5 (GMII Data) should be kept valid until the MAC clears this bit during a PHY Write operation. Similarly for a read operation, the contents of Register 5 are not valid until this bit is cleared.</p> <p>The subsequent read or write operation should happen only after the previous operation is complete. Because there is no acknowledgment from the PHY to MAC after a read or write operation is completed, there is no change in the functionality of this bit even when the PHY is not present.</p>
GW	1	rw	<p>GMII Write</p> <p>When set, this bit indicates to the PHY or RevMII that this is a Write operation using the GMII Data register. If this bit is not set, it indicates that this is a Read operation, that is, placing the data in the GMII Data register.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
CR	[5:2]	rw	<p>CSR Clock Range</p> <p>The CSR Clock Range selection determines the frequency of the MDC clock according to the clk_csr_i frequency used in your design. The suggested range of clk_csr_i frequency applicable for each value (when Bit[5] = 0) ensures that the MDC clock is approximately between the frequency range 1.0 MHz - 2.5 MHz.</p> <ul style="list-style-type: none"> - 0000: The frequency of the clk_csr_i clock is 60-100 MHz and the MDC clock is clk_csr_i/42. - 0001: The frequency of the clk_csr_i clock is 100-150 MHz and the MDC clock is clk_csr_i/62. - 0010: The frequency of the clk_csr_i clock is 20-35 MHz and the MDC clock is clk_csr_i/16. - 0011: The frequency of the clk_csr_i clock is 35-60 MHz and the MDC clock is clk_csr_i/26. - 0100: The frequency of the clk_csr_i clock is 150-250 MHz and the MDC clock is clk_csr_i/102. - 0100: The frequency of the clk_csr_i clock is 250-300 MHz and the MDC clock is clk_csr_i/124. - 0110 and 0111: Reserved <p>When Bit 5 is set, you can achieve MDC clock of frequency higher than the IEEE 802.3 specified frequency limit of 2.5 MHz and program a clock divider of lower value. For example, when clk_csr_i is of 100 MHz frequency and you program these bits as 1010, then the resultant MDC clock is of 12.5 MHz which is outside the limit of IEEE 802.3 specified range. Program the following values only if the interfacing chips support faster MDC clocks:</p> <ul style="list-style-type: none"> - 1000: clk_csr_i/4 - 1001: clk_csr_i/6 - 1010: clk_csr_i/8 - 1011: clk_csr_i/10 - 1100: clk_csr_i/12 - 1101: clk_csr_i/14 - 1110: clk_csr_i/16 - 1111: clk_csr_i/18 <p>These bits are not used for accessing RevMII. These bits are read-only if the RevMII interface is selected as single PHY interface.</p>

Ethernet MAC (ETH)

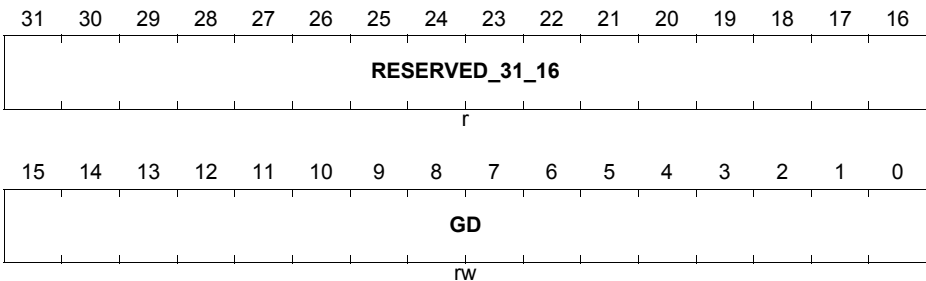
Field	Bits	Type	Description
GR	[10:6]	rw	GMII Register These bits select the desired GMII register in the selected PHY device. For RevMII, these bits select the desired CSR register in the RevMII Registers set.
PA	[15:11]	rw	Physical Layer Address This field indicates which of the 32 possible PHY devices are being accessed. For RevMII, this field gives the PHY Address of the RevMII module.
RESERVE D_31_16	[31:16]	r	RESERVED_31_16

32-bit Register - GMII_Data

The GMII Data register stores Write data to be written to the PHY register located at the address specified in Register 4 (GMII Address Register). This register also stores the Read data from the PHY register located at the address specified by Register 4.

ETH_GMII_DATA

Register 5 - GMII Data Register (1014_H) **Reset Value: 0000 0000_H**



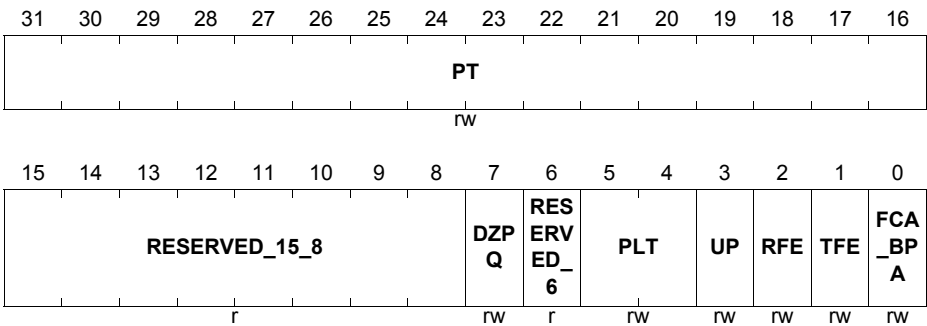
Field	Bits	Type	Description
GD	[15:0]	rw	GMII Data This field contains the 16-bit data value read from the PHY or RevMII after a Management Read operation or the 16-bit data value to be written to the PHY or RevMII before a Management Write operation.
RESERVE D_31_16	[31:16]	r	RESERVED_31_16

32-bit Register - Flow_Control

The Flow Control register controls the generation and reception of the Control (Pause Command) frames by the MAC's Flow control module. A Write to a register with the Busy bit set to '1' triggers the Flow Control block to generate a Pause Control frame. The fields of the control frame are selected as specified in the 802.3x specification, and the Pause Time value from this register is used in the Pause Time field of the control frame. The Busy bit remains set until the control frame is transferred onto the cable. The Host must make sure that the Busy bit is cleared before writing to the register.

ETH_FLOW_CONTROL

Register 6 - Flow Control Register (1018_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
FCA_BPA	0	rw	<p>Flow Control Busy or Backpressure Activate</p> <p>This bit initiates a Pause Control frame in the full-duplex mode and activates the backpressure function in the half-duplex mode if the TFE bit is set.</p> <p>In the full-duplex mode, this bit should be read as 1'b0 before writing to the Flow Control register. To initiate a Pause control frame, the Application must set this bit to 1'b1. During a transfer of the Control Frame, this bit continues to be set to signify that a frame transmission is in progress. After the completion of Pause control frame transmission, the MAC resets this bit to 1'b0. The Flow Control register should not be written to until this bit is cleared.</p> <p>In the half-duplex mode, when this bit is set (and TFE is set), then backpressure is asserted by the MAC. During backpressure, when the MAC receives a new frame, the transmitter starts sending a JAM pattern resulting in a collision. This control register bit is logically ORed with the mti_flowctrl_i input signal for the backpressure function. When the MAC is configured for the full-duplex mode, the BPA is automatically disabled.</p>
TFE	1	rw	<p>Transmit Flow Control Enable</p> <p>In the full-duplex mode, when this bit is set, the MAC enables the flow control operation to transmit Pause frames. When this bit is reset, the flow control operation in the MAC is disabled, and the MAC does not transmit any Pause frames.</p> <p>In half-duplex mode, when this bit is set, the MAC enables the back-pressure operation. When this bit is reset, the back-pressure feature is disabled.</p>
RFE	2	rw	<p>Receive Flow Control Enable</p> <p>When this bit is set, the MAC decodes the received Pause frame and disables its transmitter for a specified (Pause) time. When this bit is reset, the decode function of the Pause frame is disabled.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
UP	3	rw	<p>Unicast Pause Frame Detect</p> <p>When this bit is set, then in addition to the detecting Pause frames with the unique multicast address, the MAC detects the Pause frames with the station's unicast address specified in the MAC Address0 High Register and MAC Address0 Low Register. When this bit is reset, the MAC detects only a Pause frame with the unique multicast address specified in the 802.3x standard.</p>
PLT	[5:4]	rw	<p>Pause Low Threshold</p> <p>This field configures the threshold of the PAUSE timer at which the input flow control signal <code>mti_flowctrl_i</code> (or <code>sbd_flowctrl_i</code>) is checked for automatic retransmission of PAUSE Frame.</p> <p>The threshold values should be always less than the Pause Time configured in Bits[31:16]. For example, if <code>PT = 100H</code> (256 slot-times), and <code>PLT = 01</code>, then a second PAUSE frame is automatically transmitted if the <code>mti_flowctrl_i</code> signal is asserted at 228 (256 - 28) slot times after the first PAUSE frame is transmitted.</p> <p>The following list provides the threshold values for different values:</p> <ul style="list-style-type: none"> - 00: The threshold is Pause time minus 4 slot times (PT - 4 slot times). - 01: The threshold is Pause time minus 28 slot times (PT - 28 slot times). - 10: The threshold is Pause time minus 144 slot times (PT - 144 slot times). - 11: The threshold is Pause time minus 256 slot times (PT - 256 slot times). <p>The slot time is defined as the time taken to transmit 512 bits (64 bytes) on the GMII or MII interface.</p>
RESERVE D_6	6	r	RESERVED_6

Ethernet MAC (ETH)

Field	Bits	Type	Description
DZPQ	7	rw	<p>Disable Zero-Quanta Pause</p> <p>When this bit is set, it disables the automatic generation of the Zero-Quanta Pause Control frames on the de-assertion of the flow-control signal from the FIFO layer (MTL or external sideband flow control signal <code>sbd_flowctrl_i/mti_flowctrl_i</code>).</p> <p>When this bit is reset, normal operation with automatic Zero-Quanta Pause Control frame generation is enabled.</p>
RESERVE D_15_8	[15:8]	r	RESERVED_15_8
PT	[31:16]	rw	<p>Pause Time</p> <p>This field holds the value to be used in the Pause Time field in the transmit control frame. If the Pause Time bits is configured to be double-synchronized to the (G)MII clock domain, then consecutive writes to this register should be performed only after at least four clock cycles in the destination clock domain.</p>

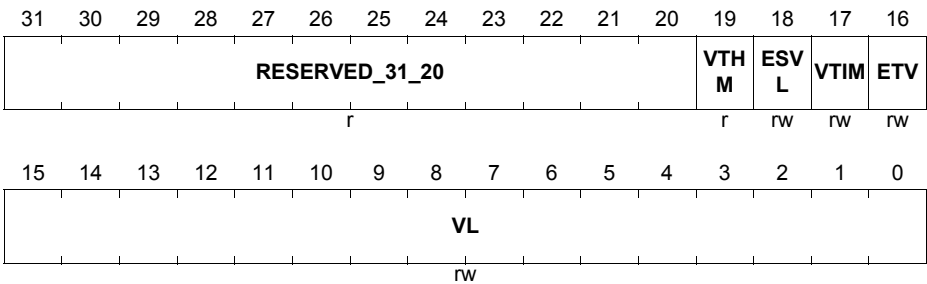
Ethernet MAC (ETH)

32-bit Register - VLAN_Tag

The VLAN Tag register contains the IEEE 802.1Q VLAN Tag to identify the VLAN frames. The MAC compares the 13th and 14th bytes of the receiving frame (Length/Type) with 16'h8100, and the following two bytes are compared with the VLAN tag. If a match occurs, the MAC sets the received VLAN bit in the receive frame status. The legal length of the frame is increased from 1,518 bytes to 1,522 bytes. If the VLAN Tag register is configured to be double-synchronized to the (G)MII clock domain, then consecutive writes to these register should be performed only after at least four clock cycles in the destination clock domain.

ETH_VLAN_TAG

Register 7 - VLAN Tag Register (101C_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
VL	[15:0]	rw	<p>VLAN Tag Identifier for Receive Frames</p> <p>This field contains the 802.1Q VLAN tag to identify the VLAN frames and is compared to the 15th and 16th bytes of the frames being received for VLAN frames. The following list describes the bits of this field:</p> <ul style="list-style-type: none"> * Bits [15:13]: User Priority * Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) * Bits[11:0]: VLAN tag's VLAN Identifier (VID) field <p>When the ETV bit is set, only the VID (Bits[11:0]) is used for comparison.</p> <p>If VL (VL[11:0] if ETV is set) is all zeros, the MAC does not check the fifteenth and 16th bytes for VLAN tag comparison, and declares all frames with a Type field value of 0x8100 or 0x88a8 as VLAN frames.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
ETV	16	rw	<p>Enable 12-Bit VLAN Tag Comparison</p> <p>When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged frame. Similarly, when enabled, only 12 bits of the VLAN tag in the received frame are used for hash-based VLAN filtering.</p> <p>When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN frame are used for comparison and VLAN hash filtering.</p>
VTIM	17	rw	<p>VLAN Tag Inverse Match Enable</p> <p>When set, this bit enables the VLAN Tag inverse matching. The frames that do not have matching VLAN Tag are marked as matched.</p> <p>When reset, this bit enables the VLAN Tag perfect matching. The frames with matched VLAN Tag are marked as matched.</p>
ESVL	18	rw	<p>Enable S-VLAN</p> <p>When this bit is set, the MAC transmitter and receiver also consider the S-VLAN (Type = 0x88A8) frames as valid VLAN tagged frames.</p>
VTHM	19	r	<p>VLAN Tag Hash Table Match Enable</p> <p>When set, the most significant four bits of the VLAN tag;Cs CRC are used to index the content of Register 354 (VLAN Hash Table Register). A value of 1 in the VLAN Hash Table register, corresponding to the index, indicates that the frame matched the VLAN hash table. When Bit 16 (ETV) is set, the CRC of the 12-bit VLAN Identifier (VID) is used for comparison whereas when ETV is reset, the CRC of the 16-bit VLAN tag is used for comparison.</p> <p>When reset, the VLAN Hash Match operation is not performed. If the VLAN Hash feature is not enabled during core configuration, this bit is reserved (RO with default value).</p>
RESERVE D_31_20	[31:20]	r	RESERVED_31_20

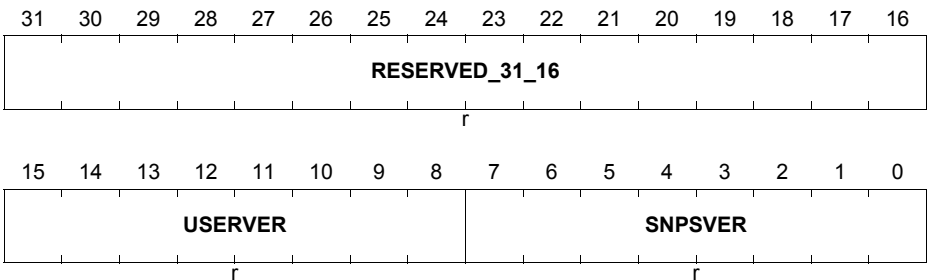
Ethernet MAC (ETH)

32-bit Register - Version

The Version registers identifies the version of the DWC_gmac. This register contains two bytes: one that Synopsys uses to identify the core release number, and the other that you set during core configuration.

ETH_VERSION

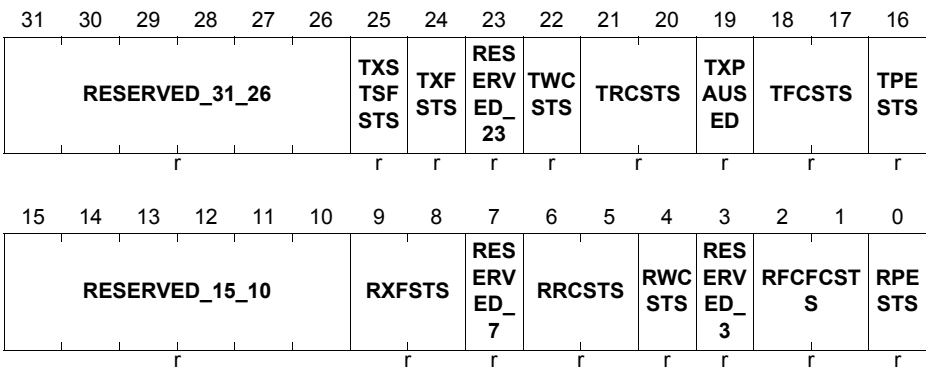
Register 8 - Version Register (1020_H) **Reset Value: 0000 5537_H**



Field	Bits	Type	Description
SNPSVER	[7:0]	r	SNPSVER Synopsys-defined Version (3.7)
USERVER	[15:8]	r	USERVER User-defined Version (Configured with the coreConsultant)
RESERVE D_31_16	[31:16]	r	RESERVED_31_16

32-bit Register - Debug

The Debug register gives the status of all main modules of the transmit and receive data-paths and the FIFOs. An all-zero status indicates that the MAC is in idle state (and FIFOs are empty) and no activity is going on in the data-paths. Note: The reset values, given for the Debug register, are valid only if the following clocks are present during the reset operation: * clk_csr_i, clk_app_i, hclk_i, or aclk_i * clk_tx_i * clk_rx_i

ETH_DEBUG
Register 9 - Debug Register
(1024_H)
Reset Value: 0000 0100_H


Field	Bits	Type	Description
RPESTS	0	r	MAC GMII or MII Receive Protocol Engine Status When high, this bit indicates that the MAC GMII or MII receive protocol engine is actively receiving data and not in IDLE state.
RFCFCST S	[2:1]	r	MAC Receive Frame Controller FIFO Status When high, this field indicates the active state of the small FIFO Read and Write controllers of the MAC Receive Frame Controller Module.
RESERVED_3	3	r	RESERVED_3
RWCSTS	4	r	MTL Rx FIFO Write Controller Active Status When high, this bit indicates that the MTL Rx FIFO Write Controller is active and is transferring a received frame to the FIFO.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RRCSTS	[6:5]	r	MTL Rx FIFO Read Controller State This field gives the state of the Rx FIFO read Controller: * 00: IDLE state * 01: Reading frame data * 10: Reading frame status (or timestamp) * 11: Flushing the frame data and status
RESERVE D_7	7	r	RESERVED_7
RXFSTS	[9:8]	r	MTL Rx FIFO Fill-level Status This field gives the status of the fill-level of the Rx FIFO: * 00: Rx FIFO Empty * 01: Rx FIFO fill level is below the flow-control deactivate threshold * 10: Rx FIFO fill level is above the flow-control activate threshold * 11: Rx FIFO Full
RESERVE D_15_10	[15:10]	r	RESERVED_15_10
TPESTS	16	r	MAC GMII or MII Transmit Protocol Engine Status When high, this bit indicates that the MAC GMII or MII transmit protocol engine is actively transmitting data and is not in the IDLE state.
TFCSTS	[18:17]	r	MAC Transmit Frame Controller Status This field indicates the state of the MAC Transmit Frame Controller module: * 00: IDLE state * 01: Waiting for Status of previous frame or IFG or backoff period to be over * 10: Generating and transmitting a PAUSE control frame (in the full-duplex mode) * 11: Transferring input frame for transmission
TXPAUSE D	19	r	MAC transmitter in PAUSE When high, this bit indicates that the MAC transmitter is in the PAUSE condition (in the full-duplex only mode) and hence does not schedule any frame for transmission.

Ethernet MAC (ETH)

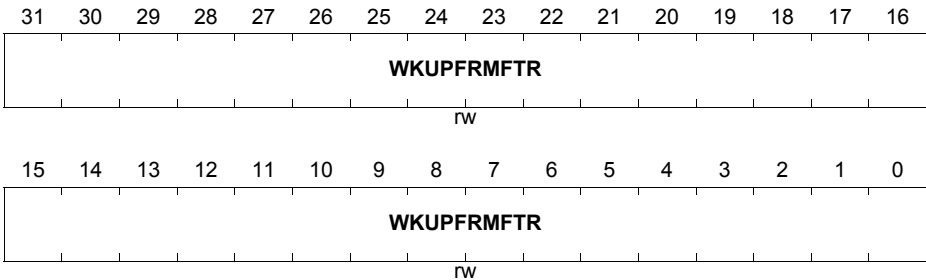
Field	Bits	Type	Description
TRCSTS	[21:20]	r	MTL Tx FIFO Read Controller Status This field indicates the state of the Tx FIFO Read Controller: * 00: IDLE state * 01: READ state (transferring data to MAC transmitter) * 10: Waiting for TxStatus from MAC transmitter * 11: Writing the received TxStatus or flushing the Tx FIFO
TWCSTS	22	r	MTL Tx FIFO Write Controller Active Status When high, this bit indicates that the MTL Tx FIFO Write Controller is active and transferring data to the Tx FIFO.
RESERVE D_23	23	r	RESERVED_23
TXFSTS	24	r	MTL Tx FIFO Not Empty Status When high, this bit indicates that the MTL Tx FIFO is not empty and some data is left for transmission.
TXSTSFS TS	25	r	MTL TxStatus FIFO Full Status When high, this bit indicates that the MTL TxStatus FIFO is full. Therefore, the MTL cannot accept any more frames for transmission. This bit is reserved in the GMAC-AHB and GMAC-DMA configurations.
RESERVE D_31_26	[31:26]	r	RESERVED_31_26

32-bit Register - Remote_Wake_Up_Frame_Filter

This is the address through which the application writes or reads the remote wake-up frame filter registers (wkupfilter_reg). The wkupfilter_reg register is a pointer to eight wkupfilter_reg registers. The wkupfilter_reg register is loaded by sequentially loading the eight register values. Eight sequential writes to this address (0x0028) writes all wkupfilter_reg registers. Similarly, eight sequential reads from this address (0x0028) read all wkupfilter_reg registers. This register contains the higher 16 bits of the seventh MAC address. This register is present only when you select the PMT module Remote Wake-up feature in coreConsultant.

ETH_REMOTE_WAKE_UP_FRAME_FILTER

Register 10 - Remote Wake-Up Frame Filter Register (1028_H) **Reset Value: 0000 0000_H**



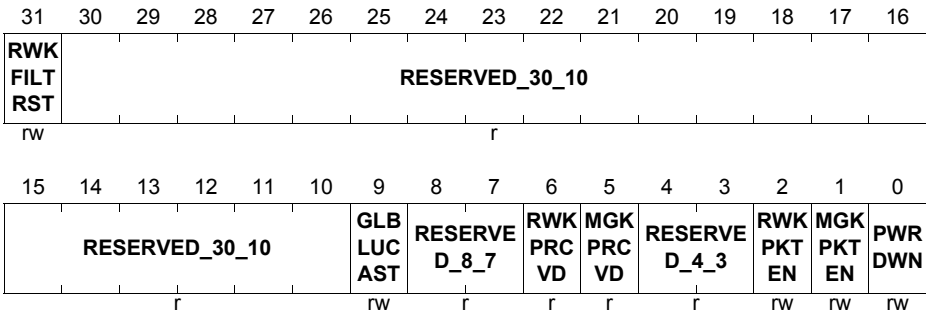
Field	Bits	Type	Description
WKUPFR MFTR	[31:0]	rw	WKUPFRMFTR Remote Wake-Up Frame Filter

32-bit Register - PMT_Control_Status

This register is present only when you select the PMT module in the coreConsultant.

ETH_PMT_CONTROL_STATUS

Register 11 - PMT Control and Status Register (102C_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
PWRDWN	0	rw	Power Down When set, the MAC receiver drops all received frames until it receives the expected magic packet or wake-up frame. This bit is then self-cleared and the power-down mode is disabled. The Software can also clear this bit before the expected magic packet or wake-up frame is received. The frames, received by the MAC after this bit is cleared, are forwarded to the application. This bit must only be set when the Magic Packet Enable, Global Unicast, or Wake-Up Frame Enable bit is set high. Note: You can gate-off the CSR clock during the power-down mode. However, when the CSR clock is gated-off, you cannot perform any read or write operations on this register. Therefore, the Software cannot clear this bit.
MGKPKTEN	1	rw	Magic Packet Enable When set, enables generation of a power management event because of magic packet reception.
RWKPKTEN	2	rw	Wake-Up Frame Enable When set, enables generation of a power management event because of wake-up frame reception.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RESERVE D_4_3	[4:3]	r	RESERVED_4_3
MGKPRC VD	5	r	Magic Packet Received When set, this bit indicates that the power management event is generated because of the reception of a magic packet. This bit is cleared by a Read into this register.
RWKPRC VD	6	r	Wake-Up Frame Received When set, this bit indicates the power management event is generated because of the reception of a wake-up frame. This bit is cleared by a Read into this register.
RESERVE D_8_7	[8:7]	r	RESERVED_8_7
GLBLUCA ST	9	rw	Global Unicast When set, enables any unicast packet filtered by the MAC (DAF) address recognition to be a wake-up frame.
RESERVE D_30_10	[30:10]	r	RESERVED_30_10
RWKFILTER RST	31	rw	Wake-Up Frame Filter Register Pointer Reset When set, resets the remote wake-up frame filter register pointer to 3jÇb000. It is automatically cleared after 1 clock cycle.

32-bit Register - Interrupt_Status

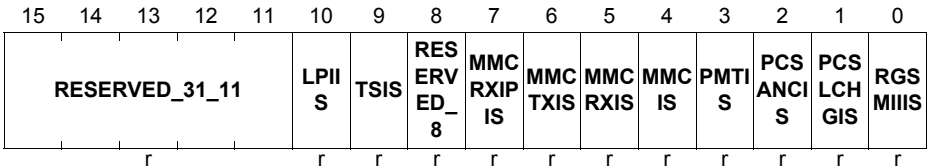
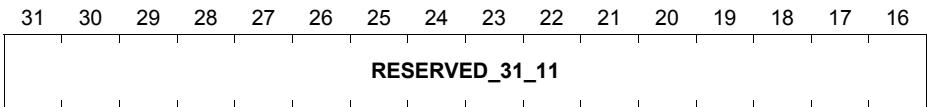
The Interrupt Status register identifies the events in the MAC that can generate interrupt. All interrupt events are generated only when the corresponding optional feature is selected during core configuration and enabled during operation. Therefore, these bits are reserved when the corresponding features are not present in the core.

ETH_INTERRUPT_STATUS

Register 14 - Interrupt Register

(1038_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RGSMIIS	0	r	<p>RGMII or SMII Interrupt Status</p> <p>This bit is set because of any change in value of the Link Status of RGMII or SMII interface (Bit 3 in Register 54 (SGMII/RGMII/SMII Status Register)). This bit is cleared when you perform a read operation on the SGMII/RGMII/SMII Status Register.</p> <p>This bit is valid only when you select the optional RGMII or SMII PHY interface during core configuration and operation.</p>
PCSLCHGIS	1	r	<p>PCS Link Status Changed</p> <p>This bit is set because of any change in Link Status in the TBI, RTBI, or SGMII PHY interface (Bit 2 in Register 49 (AN Status Register)). This bit is cleared when you perform a read operation on the AN Status register.</p> <p>This bit is valid only when you select the optional TBI, RTBI, or SGMII PHY interface during core configuration and operation.</p>

Ethernet MAC (ETH)

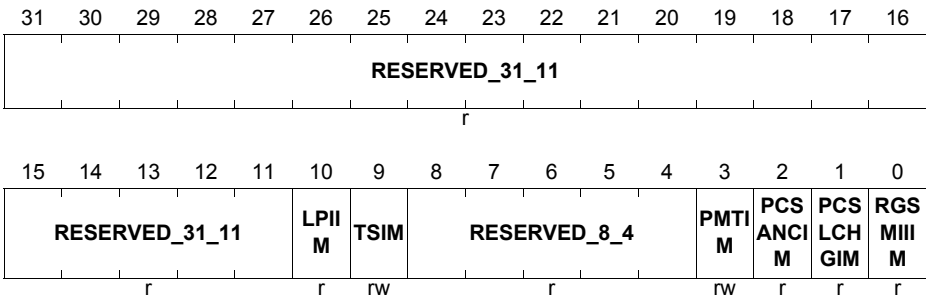
Field	Bits	Type	Description
PCSANCI S	2	r	<p>PCS Auto-Negotiation Complete</p> <p>This bit is set when the Auto-negotiation is completed in the TBI, RTBI, or SGMII PHY interface (Bit 5 in Register 49 (AN Status Register)). This bit is cleared when you perform a read operation to the AN Status register. This bit is valid only when you select the optional TBI, RTBI, or SGMII PHY interface during core configuration and operation.</p>
PMTIS	3	r	<p>PMT Interrupt Status</p> <p>This bit is set when a Magic packet or Wake-on-LAN frame is received in the power-down mode (see Bits 5 and 6 in the PMT Control and Status Register). This bit is cleared when both Bits[6:5] are cleared because of a read operation to the PMT Control and Status register. This bit is valid only when you select the optional PMT module during core configuration.</p>
MMCIS	4	r	<p>MMC Interrupt Status</p> <p>This bit is set high when any of the Bits [7:5] is set high and cleared only when all of these bits are low. This bit is valid only when you select the optional MMC module during core configuration.</p>
MMCRXIS	5	r	<p>MMC Receive Interrupt Status</p> <p>This bit is set high when an interrupt is generated in the MMC Receive Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. This bit is valid only when you select the optional MMC module during core configuration.</p>
MMCTXIS	6	r	<p>MMC Transmit Interrupt Status</p> <p>This bit is set high when an interrupt is generated in the MMC Transmit Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. This bit is valid only when you select the optional MMC module during core configuration.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
MMCRXIPIS	7	r	MMC Receive Checksum Offload Interrupt Status This bit is set high when an interrupt is generated in the MMC Receive Checksum Offload Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. This bit is valid only when you select the optional MMC module and Checksum Offload Engine (Type 2) during core configuration.
RESERVE D_8	8	r	RESERVED_8
TSIS	9	r	Timestamp Interrupt Status When the Advanced Timestamp feature is enabled, this bit is set when any of the following conditions is true: <ul style="list-style-type: none"> * The system time value equals or exceeds the value specified in the Target Time High and Low registers. * There is an overflow in the seconds register. * The Auxiliary snapshot trigger is asserted. This bit is cleared on reading Bit 0 of the Register 458 (Timestamp Status Register). If default Timestamping is enabled, when set, this bit indicates that the system time value is equal to or exceeds the value specified in the Target Time registers. In this mode, this bit is cleared after the completion of the read of this bit. In all other modes, this bit is reserved.
LPIIS	10	r	LPI Interrupt Status When the Energy Efficient Ethernet feature is enabled, this bit is set for any LPI state entry or exit in the MAC Transmitter or Receiver. This bit is cleared on reading Bit 0 of Register 12 (LPI Control and Status Register). In all other modes, this bit is reserved.
RESERVE D_31_11	[31:11]	r	RESERVED_31_11

32-bit Register - Interrupt_Mask

The Interrupt Mask Register bits enable you to mask the interrupt signal because of the corresponding event in the Interrupt Status Register. The interrupt signal is `sbd_intr_o` in the GMAC-AHB, GMAC-AXI, and GMAC-DMA configuration and `mci_intr_o` in the GMAC-MTL and GMAC-CORE configuration.

ETH_INTERRUPT_MASK
Register 15 - Interrupt Mask Register (103C_H)
Reset Value: 0000 0000_H


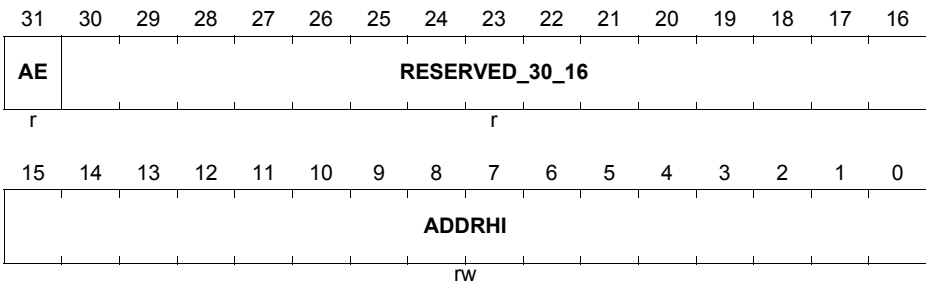
Field	Bits	Type	Description
RGSMIIIM	0	r	RGMIIM or SMII Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of the RGMIIM or SMII Interrupt Status bit in Register 14 (Interrupt Status Register).
PCSLCHGIM	1	r	PCS Link Status Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of the PCS Link-status changed bit in Register 14 (Interrupt Status Register).
PCSANCI M	2	r	PCS AN Completion Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of PCS Auto-negotiation complete bit in Register 14 (Interrupt Status Register).
PMTIM	3	rw	PMT Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of PMT Interrupt Status bit in Register 14 (Interrupt Status Register).
RESERVE D_8_4	[8:4]	r	RESERVED_8_4

Ethernet MAC (ETH)

Field	Bits	Type	Description
TSIM	9	rw	Timestamp Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of Timestamp Interrupt Status bit in Register 14 (Interrupt Status Register). This bit is valid only when IEEE1588 timestamping is enabled. In all other modes, this bit is reserved.
LPIIM	10	r	LPI Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of the LPI Interrupt Status bit in Register 14 (Interrupt Status Register). This bit is valid only when you select the Energy Efficient Ethernet feature during core configuration. In all other modes, this bit is reserved.
RESERVE D_31_11	[31:11]	r	RESERVED_31_11

32-bit Register - MAC_Address0_High

The MAC Address0 High register holds the upper 16 bits of the first 6-byte MAC address of the station. The first DA byte that is received on the (G)MII interface corresponds to the LS byte (Bits [7:0]) of the MAC Address Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (G)MII as the destination address, then the MacAddress0 Register [47:0] is compared with 0x665544332211. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address0 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS0_HIGH
Register 16 - MAC Address0 High Register (1040_H) **Reset Value: 8000 FFFF_H**


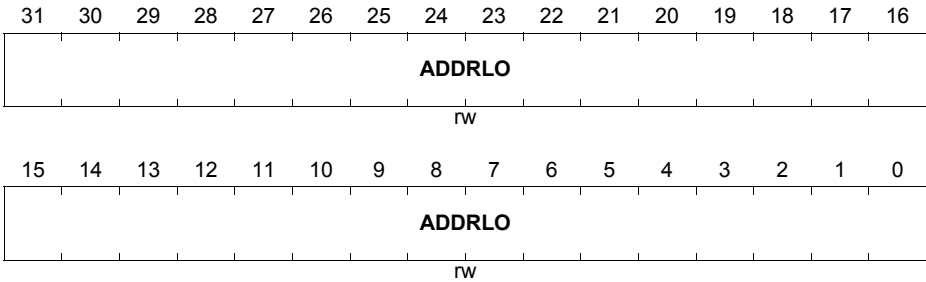
Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address0 [47:32] This field contains the upper 16 bits (47:32) of the first 6-byte MAC address. The MAC uses this field for filtering the received frames and inserting the MAC address in the Transmit Flow Control (PAUSE) Frames.
RESERVE D_30_16	[30:16]	r	RESERVED_30_16
AE	31	r	Address Enable This bit is always set to 1.

32-bit Register - MAC_Address0_Low

The MAC Address0 Low register holds the lower 32 bits of the first 6-byte MAC address of the station.

ETH_MAC_ADDRESS0_LOW

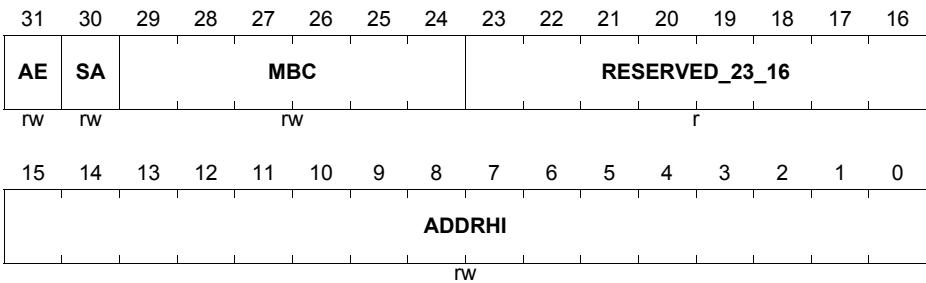
Register 17 - MAC Address0 Low Register (1044_H) **Reset Value: FFFF FFFF_H**



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	<p>MAC Address0 [31:0]</p> <p>This field contains the lower 32 bits of the first 6-byte MAC address. This is used by the MAC for filtering the received frames and inserting the MAC address in the Transmit Flow Control (PAUSE) Frames.</p>

Ethernet MAC (ETH)
32-bit Register - MAC_Address1_High

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS1_HIGH
Register 18 - MAC Address1 High Register (1048_H) **Reset Value: 0000 FFFF_H**


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address1 [47:32] This field contains the upper 16 bits (47:32) of the second 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16

Ethernet MAC (ETH)

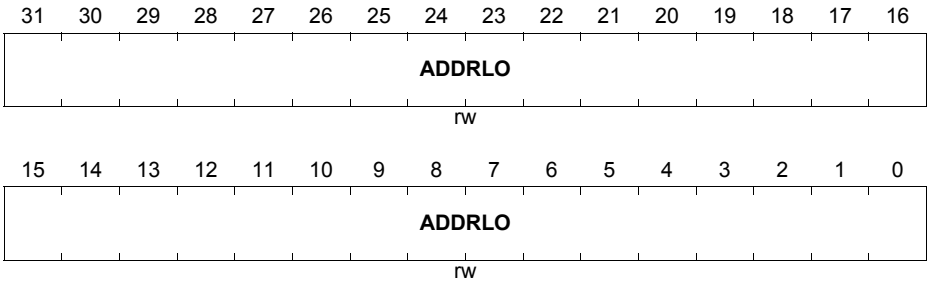
Field	Bits	Type	Description
MBC	[29:24]	rw	<p>Mask Byte Control</p> <p>These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows:</p> <ul style="list-style-type: none"> * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0] <p>You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address.</p>
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address1_Low

The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

ETH_MAC_ADDRESS1_LOW

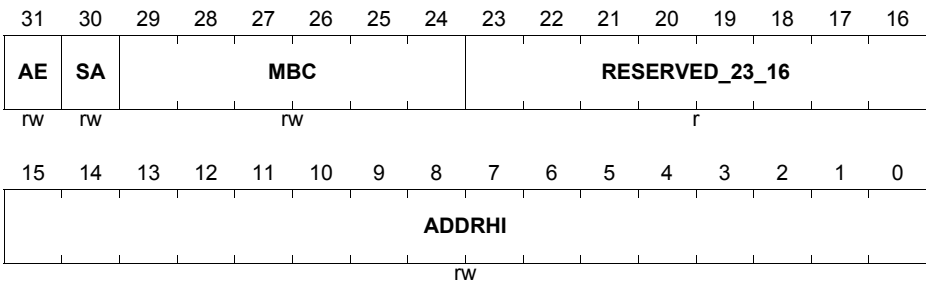
Register 19 - MAC Address1 Low Register (104C_H) **Reset Value: FFFF FFFF_H**



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address1 [31:0] This field contains the lower 32 bits of the second 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address2_High

The MAC Address2 High register holds the upper 16 bits of the third 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address2 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address2 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS2_HIGH
Register 20 - MAC Address2 High Register (1050_H) **Reset Value: 0000 FFFF_H**


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address2 [47:32] This field contains the upper 16 bits (47:32) of the third 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address2 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

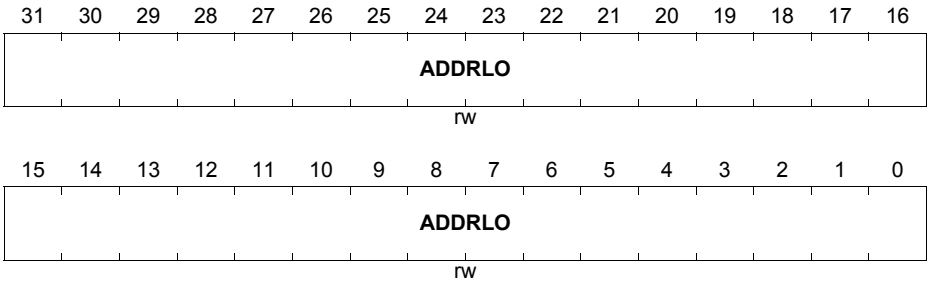
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address2[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address2[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the third MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address2_Low

The MAC Address2 Low register holds the lower 32 bits of the third 6-byte MAC address of the station.

ETH_MAC_ADDRESS2_LOW

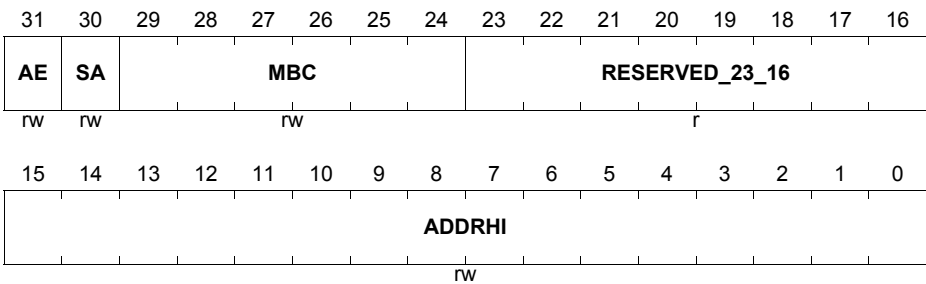
Register 21 - MAC Address2 Low Register (1054_H) **Reset Value: FFFF FFFF_H**



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address2 [31:0] This field contains the lower 32 bits of the third 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address3_High

The MAC Address3 High register holds the upper 16 bits of the fourth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address3 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address3 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS3_HIGH
Register 22 - MAC Address3 High Register (1058_H) **Reset Value: 0000 FFFF_H**


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address3 [47:32] This field contains the upper 16 bits (47:32) of the fourth 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address3 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

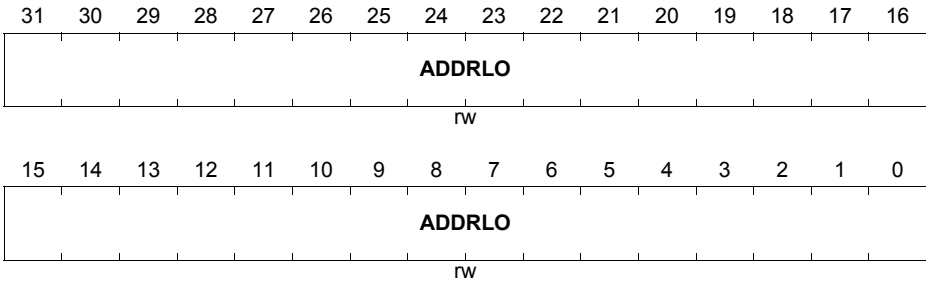
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address3[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address3[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the fourth MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address3_Low

The MAC Address3 Low register holds the lower 32 bits of the fourth 6-byte MAC address of the station.

ETH_MAC_ADDRESS3_LOW

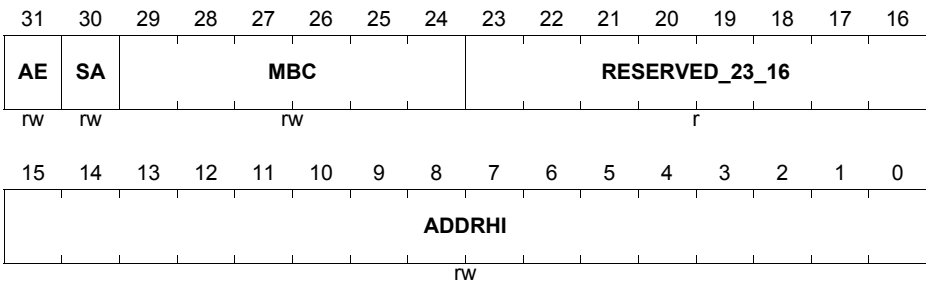
Register 23 - MAC Address3 Low Register (105C_H) **Reset Value: FFFF FFFF_H**



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address3 [31:0] This field contains the lower 32 bits of the fourth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address4_High

The MAC Address4 High register holds the upper 16 bits of the fifth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address4 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address4 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS4_HIGH
Register 24 - MAC Address4 High Register (1060_H) **Reset Value: 0000 FFFF_H**


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address4 [47:32] This field contains the upper 16 bits (47:32) of the fifth 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address4 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

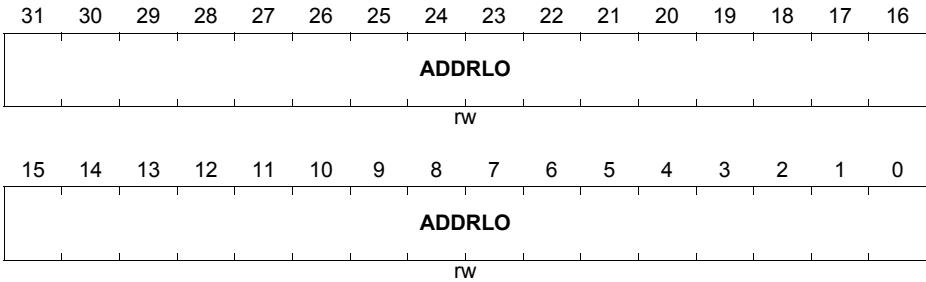
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address4[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address4[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the fifth MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address4_Low

The MAC Address4 Low register holds the lower 32 bits of the fifth 6-byte MAC address of the station.

ETH_MAC_ADDRESS4_LOW

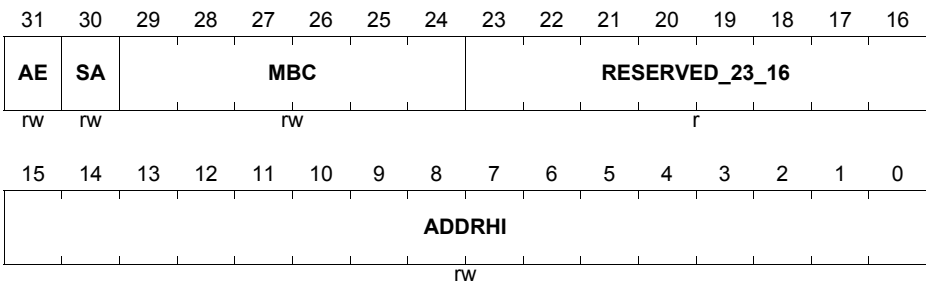
Register 25 - MAC Address4 Low Register (1064_H) **Reset Value: FFFF FFFF_H**



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address4 [31:0] This field contains the lower 32 bits of the fifth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address5_High

The MAC Address5 High register holds the upper 16 bits of the sixth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address5 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address5 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS5_HIGH
Register 26 - MAC Address5 High Register (1068_H) **Reset Value: 0000 FFFF_H**


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address5 [47:32] This field contains the upper 16 bits (47:32) of the sixth 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address5 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

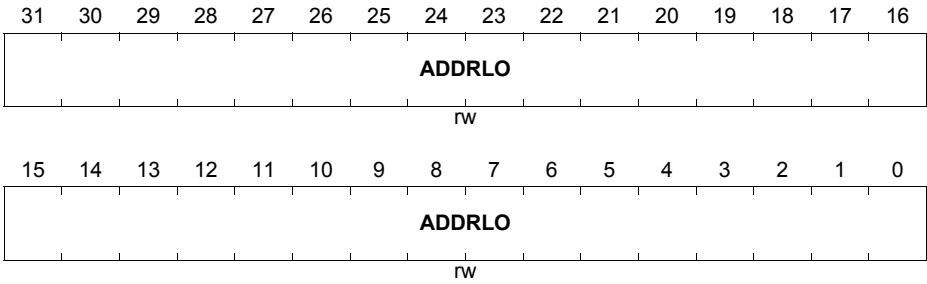
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address5[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address5[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the sixth MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address5_Low

The MAC Address5 Low register holds the lower 32 bits of the sixth 6-byte MAC address of the station.

ETH_MAC_ADDRESS5_LOW

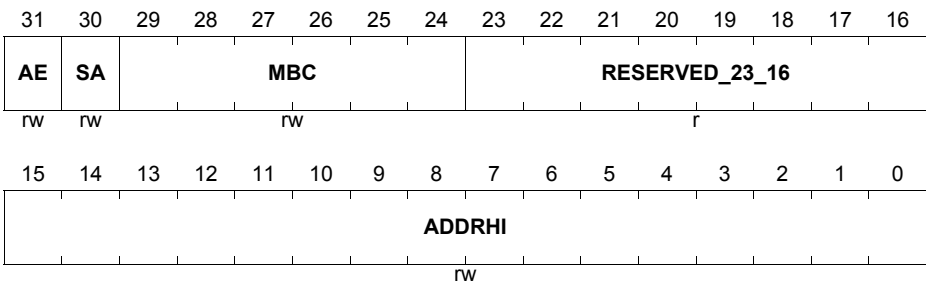
Register 27 - MAC Address5 Low Register (106C_H) **Reset Value: FFFF FFFF_H**



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address5 [31:0] This field contains the lower 32 bits of the sixth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address6_High

The MAC Address6 High register holds the upper 16 bits of the seventh 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address6 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address6 Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS6_HIGH
Register 28 - MAC Address6 High Register (1070_H) **Reset Value: 0000 FFFF_H**


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address6 [47:32] This field contains the upper 16 bits (47:32) of the seventh 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address6 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

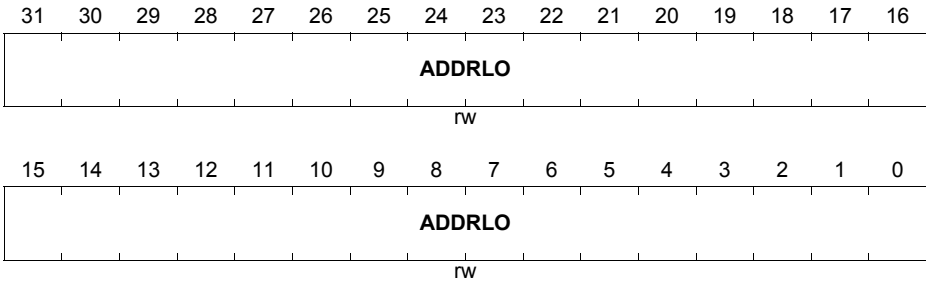
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address6[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address6[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the seventh MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address6_Low

The MAC Address6 Low register holds the lower 32 bits of the seventh 6-byte MAC address of the station.

ETH_MAC_ADDRESS6_LOW

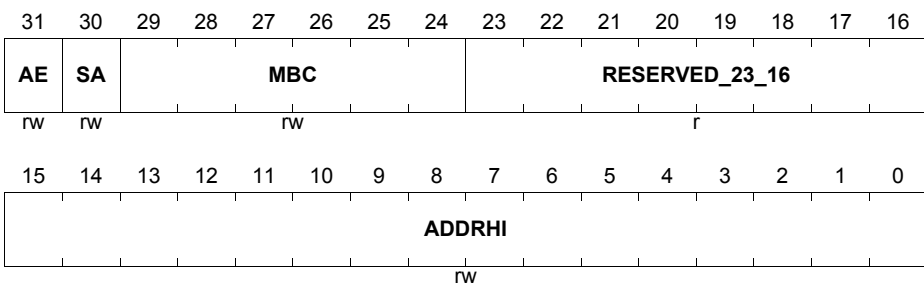
Register 29 - MAC Address6 Low Register (1074_H) **Reset Value: FFFF FFFF_H**



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	<p>MAC Address6 [31:0]</p> <p>This field contains the lower 32 bits of the seventh 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.</p>

32-bit Register - MAC_Address7_High

The MAC Address7 High register holds the upper 16 bits of the eighth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address7 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address7 Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS7_HIGH
Register 30 - MAC Address7 High Register (1078_H) **Reset Value: 0000 FFFF_H**


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address7 [47:32] This field contains the upper 16 bits (47:32) of the eighth 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address6 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

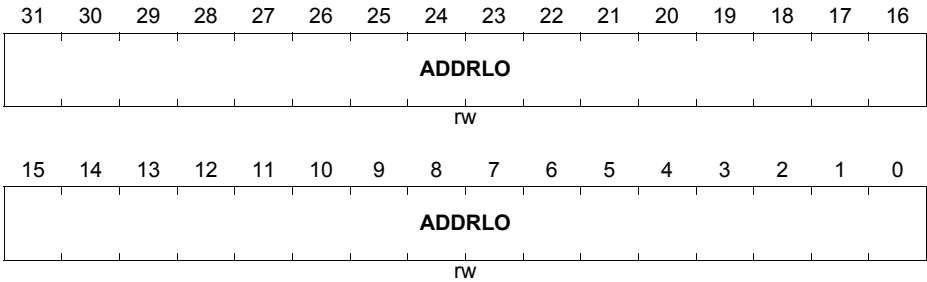
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address7[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address7[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the eighth MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address7_Low

The MAC Address7 Low register holds the lower 32 bits of the eighth 6-byte MAC address of the station.

ETH_MAC_ADDRESS7_LOW

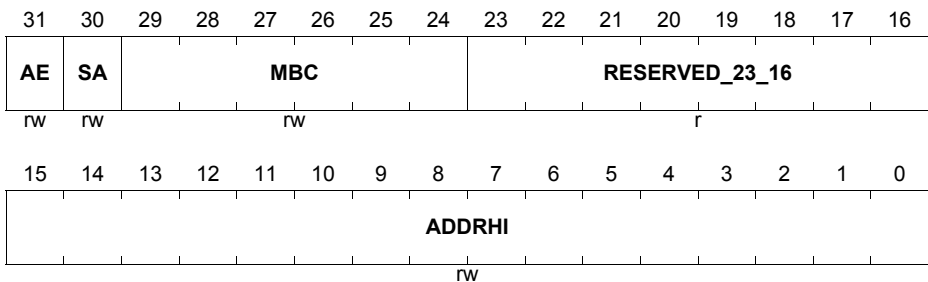
Register 31 - MAC Address7 Low Register (107C_H) **Reset Value: FFFF FFFF_H**



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address7 [31:0] This field contains the lower 32 bits of the eighth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address8_High

The MAC Address8 High register holds the upper 16 bits of the ninth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address8 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address8 Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS8_HIGH
Register 32 - MAC Address8 High Register (1080_H) **Reset Value: 0000 FFFF_H**


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address8 [47:32] This field contains the upper 16 bits (47:32) of the ninth 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address8 registers. Each bit controls the masking of the bytes as follows: <ul style="list-style-type: none"> * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

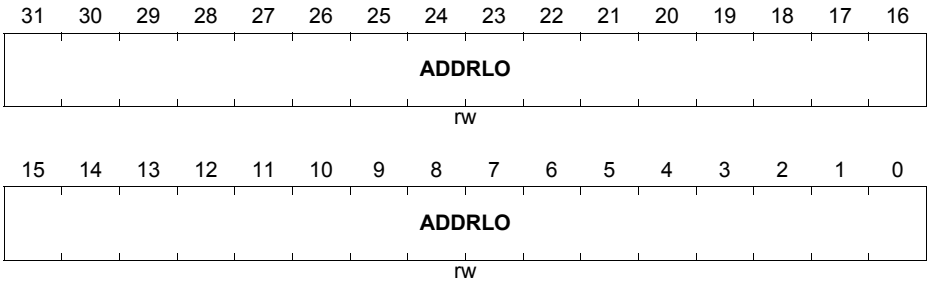
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address8[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address8[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the ninth MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address8_Low

The MAC Address8 Low register holds the lower 32 bits of the ninth 6-byte MAC address of the station.

ETH_MAC_ADDRESS8_LOW

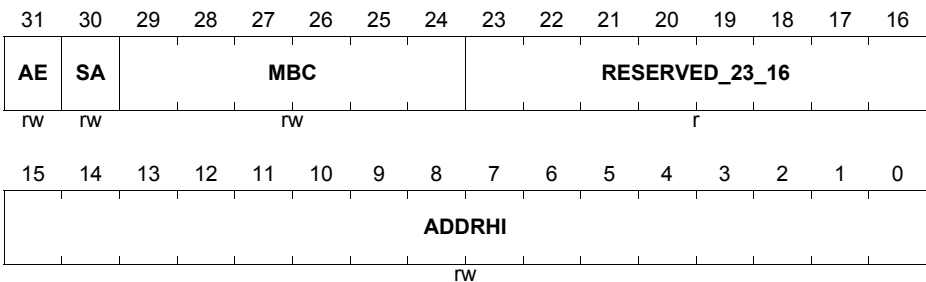
Register 33 - MAC Address8 Low Register (1084_H) **Reset Value: FFFF FFFF_H**



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address8 [31:0] This field contains the lower 32 bits of the ninth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address9_High

The MAC Address9 High register holds the upper 16 bits of the tenth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address9 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address9 Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS9_HIGH
Register 34 - MAC Address9 High Register (1088_H) **Reset Value: 0000 FFFF_H**


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address9 [47:32] This field contains the upper 16 bits (47:32) of the tenth 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address9 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

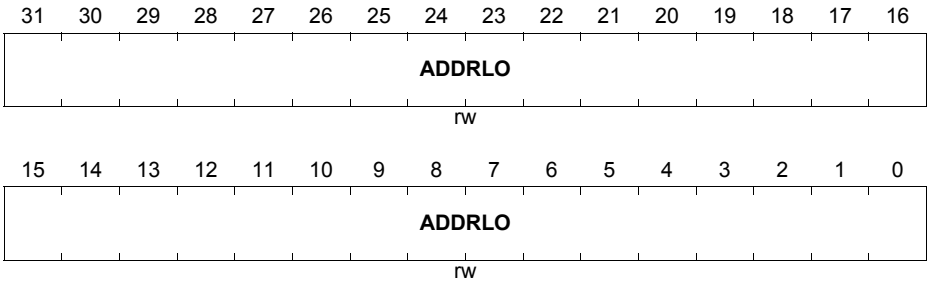
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address9[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address9[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the tenth MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address9_Low

The MAC Address9 Low register holds the lower 32 bits of the tenth 6-byte MAC address of the station.

ETH_MAC_ADDRESS9_LOW

Register 35 - MAC Address9 Low Register (108C_H) **Reset Value: FFFF FFFF_H**



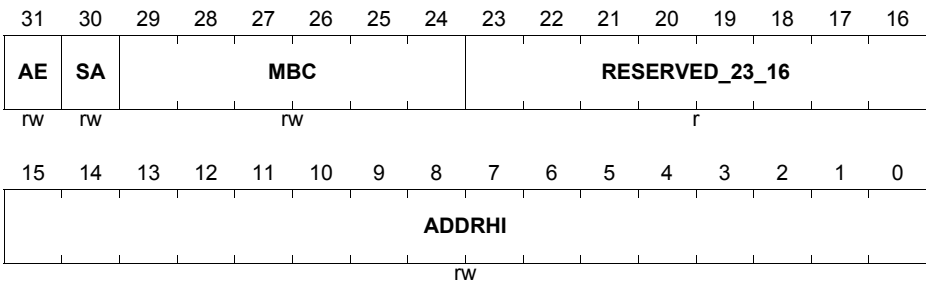
Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address9 [31:0] This field contains the lower 32 bits of the tenth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address10_High

The MAC Address10 High register holds the upper 16 bits of the 11th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address10 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address10 Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS10_HIGH

Register 36 - MAC Address10 High Register (1090_H) **Reset Value: 0000 FFFF_H**



Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address10 [47:32] This field contains the upper 16 bits (47:32) of the 11th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address10 registers. Each bit controls the masking of the bytes as follows: <ul style="list-style-type: none"> * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address10[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address10[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 11th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.</p>

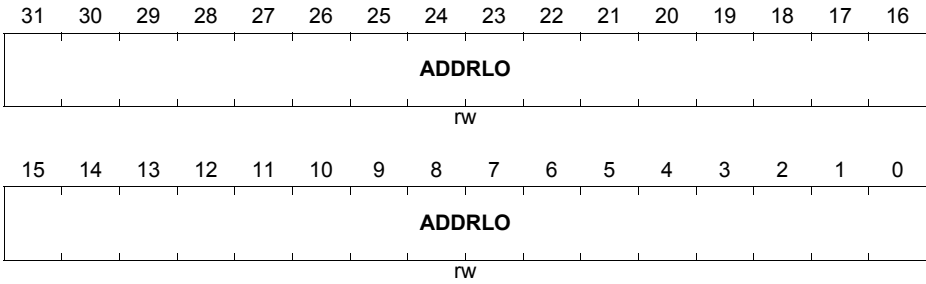
Ethernet MAC (ETH)

32-bit Register - MAC_Address10_Low

The MAC Address10 Low register holds the lower 32 bits of the 11th 6-byte MAC address of the station.

ETH_MAC_ADDRESS10_LOW

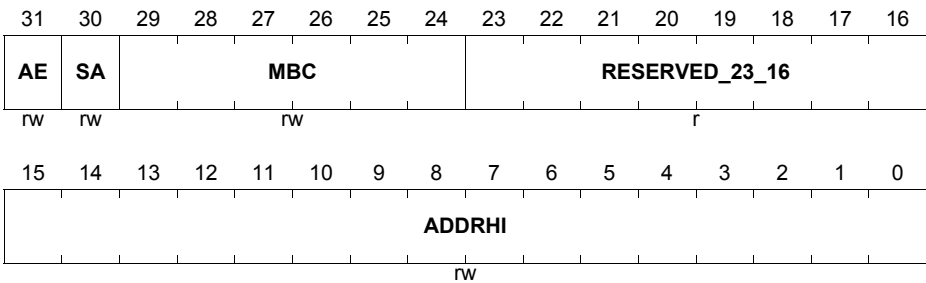
Register 37 - MAC Address10 Low Register (1094_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address10 [31:0] This field contains the lower 32 bits of the 11th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address11_High

The MAC Address11 High register holds the upper 16 bits of the 12th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address11 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address11 Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS11_HIGH
Register 38 - MAC Address11 High Register (1098_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address11 [47:32] This field contains the upper 16 bits (47:32) of the 12th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address11 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

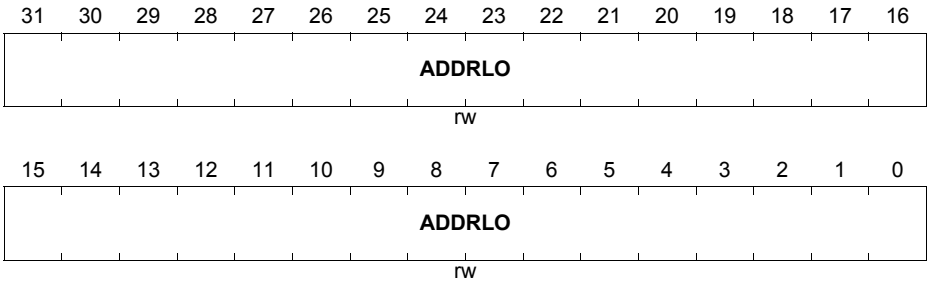
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address11[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address11[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the twelfth MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address11_Low

The MAC Address11 Low register holds the lower 32 bits of the 12th 6-byte MAC address of the station.

ETH_MAC_ADDRESS11_LOW

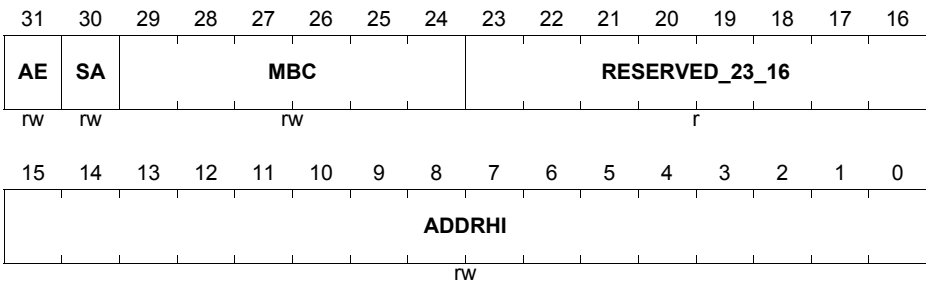
Register 39 - MAC Address1 Low Register (109C_H) **Reset Value: FFFF FFFF_H**



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address11 [31:0] This field contains the lower 32 bits of the 12th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address12_High

The MAC Address12 High register holds the upper 16 bits of the 13th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address13 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address12 Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS12_HIGH
Register 40 - MAC Address12 High Register (10A0_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address12 [47:32] This field contains the upper 16 bits (47:32) of the 13th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address12 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address12[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address12[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 13th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.</p>

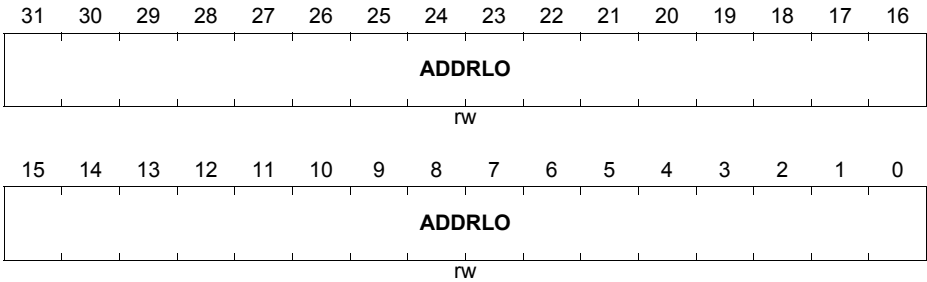
Ethernet MAC (ETH)

32-bit Register - MAC_Address12_Low

The MAC Address12 Low register holds the lower 32 bits of the 13th 6-byte MAC address of the station.

ETH_MAC_ADDRESS12_LOW

Register 41 - MAC Address12 Low Register (10A4_H) Reset Value: FFFF FFFF_H



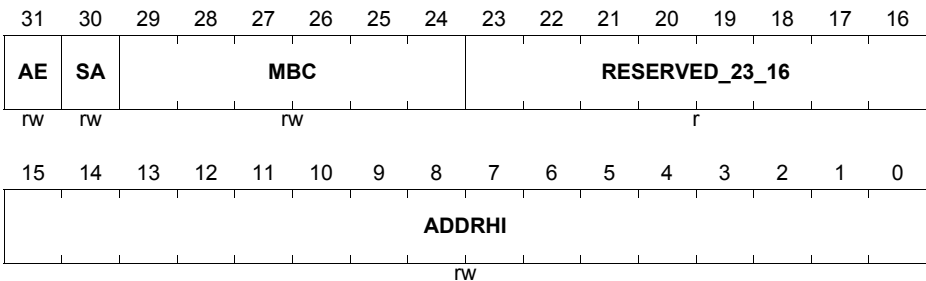
Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address12 [31:0] This field contains the lower 32 bits of the 13th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address13_High

The MAC Address13 High register holds the upper 16 bits of the 14th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address13 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address13 Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS13_HIGH

Register 42 - MAC Address13 High Register (10A8_H) **Reset Value: 0000 FFFF_H**



Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address13 [47:32] This field contains the upper 16 bits (47:32) of the 14th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address13 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

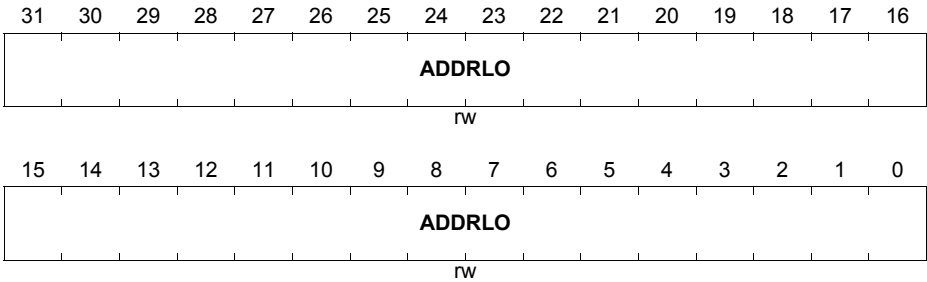
Field	Bits	Type	Description
SA	30	rw	Source Address When this bit is set, the MAC Address13[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address13[47:0] is used to compare with the DA fields of the received frame.
AE	31	rw	Address Enable When this bit is set, the address filter module uses the 14th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.

32-bit Register - MAC_Address13_Low

The MAC Address13 Low register holds the lower 32 bits of the 14th 6-byte MAC address of the station.

ETH_MAC_ADDRESS13_LOW

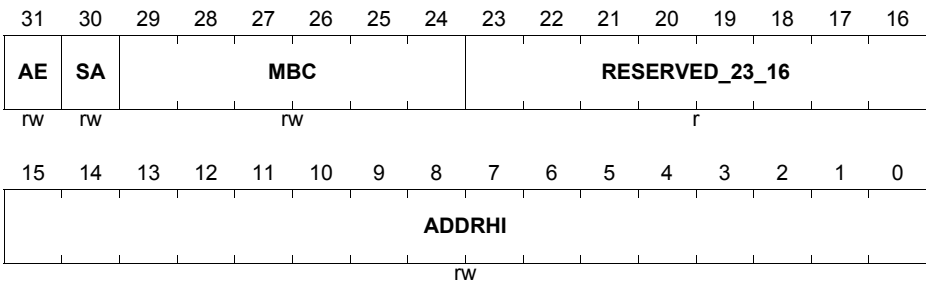
Register 43 - MAC Address13 Low Register (10AC_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address13 [31:0] This field contains the lower 32 bits of the 14th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address14_High

The MAC Address14 High register holds the upper 16 bits of the 15th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address15 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address14 Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS14_HIGH
Register 44 - MAC Address14 High Register (10B0_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address14 [47:32] This field contains the upper 16 bits (47:32) of the 15th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address14 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address14[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address14[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 15th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.</p>

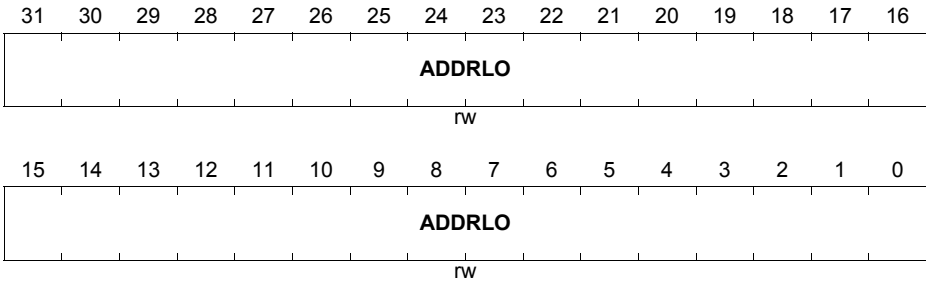
Ethernet MAC (ETH)

32-bit Register - MAC_Address14_Low

The MAC Address14 Low register holds the lower 32 bits of the 15th 6-byte MAC address of the station.

ETH_MAC_ADDRESS14_LOW

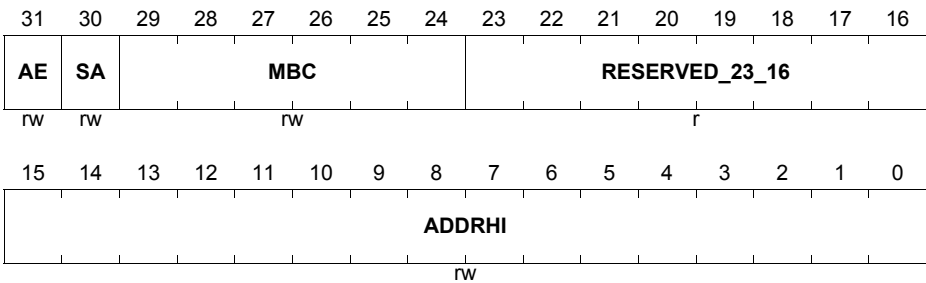
Register 45 - MAC Address14 Low Register (10B4_H) **Reset Value: FFFF FFFF_H**



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address14 [31:0] This field contains the lower 32 bits of the 15th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address15_High

The MAC Address15 High register holds the upper 16 bits of the 16th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address15 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address15 Low Register should be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS15_HIGH
Register 46 - MAC Address15 High Register (10B8_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	ADDRHI MAC Address15 [47:32] This field contains the upper 16 bits (47:32) of the 16th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address15 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

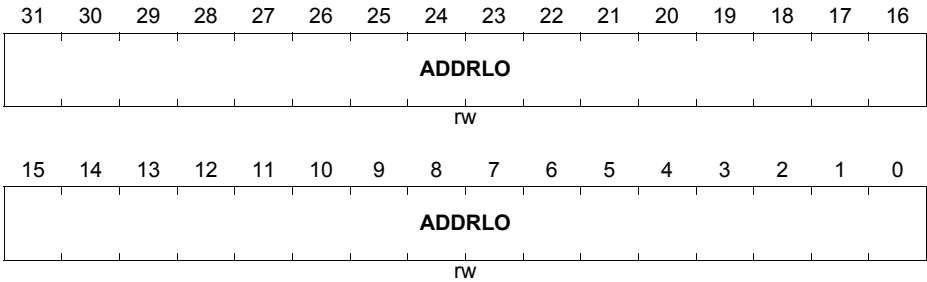
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address15[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address15[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 16th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address15_Low

The MAC Address15 Low register holds the lower 32 bits of the 16th 6-byte MAC address of the station.

ETH_MAC_ADDRESS15_LOW

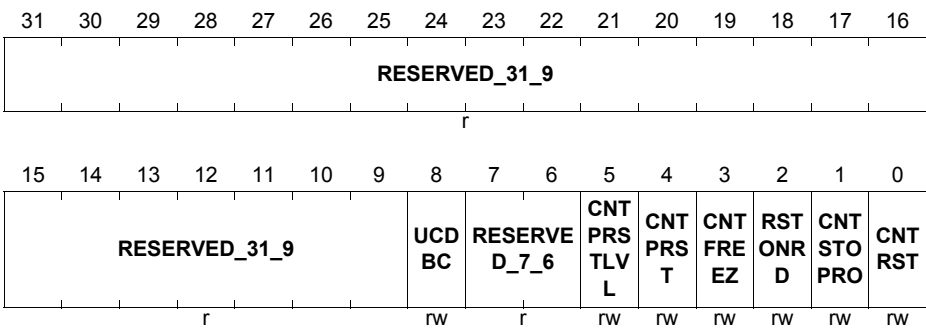
Register 47 - MAC Address15 Low Register (10BC_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address15 [31:0] This field contains the lower 32 bits of the 16th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MMC_Control

The MMC Control register establishes the operating mode of the management counters. Note: The bit 0 (Counters Reset) has higher priority than bit 4 (Counter Preset). Therefore, when the Software tries to set both bits in the same write cycle, all counters are cleared and the bit 4 is not set.

ETH_MMC_CONTROL
Register 64 - MMC Control Register (1100_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CNTRST	0	rw	Counters Reset When this bit is set, all counters are reset. This bit is cleared automatically after one clock cycle.
CNTSTOPRO	1	rw	Counters Stop Rollover When this bit is set, after reaching maximum value, the counter does not roll over to zero.
RSTONRD	2	rw	Reset on Read When this bit is set, the MMC counters are reset to zero after Read (self-clearing after reset). The counters are cleared when the least significant byte lane (bits[7:0]) is read.
CNTFREEZ	3	rw	MMC Counter Freeze When this bit is set, it freezes all MMC counters to their current value. Until this bit is reset to 0, no MMC counter is updated because of any transmitted or received frame. If any MMC counter is read with the Reset on Read bit set, then that counter is also cleared in this mode.

Ethernet MAC (ETH)

Field	Bits	Type	Description
CNTPRST	4	rw	Counters Preset When this bit is set, all counters are initialized or preset to almost full or almost half according to bit 5. This bit is cleared automatically after 1 clock cycle. This bit, along with bit 5, is useful for debugging and testing the assertion of interrupts because of MMC counter becoming half-full or full.
CNTPRST LVL	5	rw	Full-Half Preset When low and bit 4 is set, all MMC counters get preset to almost-half value. All octet counters get preset to 0x7FFF_F800 (half - 2KBytes) and all frame-counters gets preset to 0x7FFF_FFF0 (half - 16). When this bit is high and bit 4 is set, all MMC counters get preset to almost-full value. All octet counters get preset to 0xFFFF_F800 (full - 2KBytes) and all frame-counters gets preset to 0xFFFF_FFF0 (full - 16). For 16-bit counters, the almost-half preset values are 0x7800 and 0x7FF0 for the respective octet and frame counters. Similarly, the almost-full preset values for the 16-bit counters are 0xF800 and 0xFFFF0.
RESERVE D_7_6	[7:6]	r	RESERVED_7_6
UCDBC	8	rw	Update MMC Counters for Dropped Broadcast Frames When set, this bit enables MAC to update all the related MMC Counters for Broadcast frames dropped due to setting of DBF bit (Disable Broadcast Frames) of MAC Filter Register at offset 0x0004. When reset, MMC Counters are not updated for dropped Broadcast frames.
RESERVE D_31_9	[31:9]	r	RESERVED_31_9

32-bit Register - MMC_Receive_Interrupt

The MMC Receive Interrupt register maintains the interrupts that are generated when the following happens: * Receive statistic counters reach half of their maximum values (0x8000_0000 for 32-bit counter and 0x8000 for 16-bit counter). * Receive statistic counters cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When the Counter Stop Rollover is set, then interrupts are set but the counter remains at all-ones. The MMC Receive Interrupt register is a 32-bit wide register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read in order to clear the interrupt bit.

ETH_MMC_RECEIVE_INTERRUPT

Register 65 - MMC Receive Interrupt Register (1104_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED_31_26						RXC TRL FIS	RXR CVE RRFI S	RXW DOG FIS	RXV LAN GBFI S	RXF OVFI S	RXP AUS FIS	RXO RAN GEFI S	RXL ENE RFIS	RXU CGFI S	RX1 024T MAX OCT GBFI
r						r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX5 12T1 023O CTG BFIS	RX2 56T5 11O CTG BFIS	RX1 28T2 55O CTG BFIS	RX6 5T12 7OC TGB FIS	RX6 4OC TGB FIS	RXO SIZE GFIS	RXU SIZE GFIS	RXJ ABE RFIS	RXR UNT FIS	RXA LGN ERFI S	RXC RCE RFIS	RXM CGFI S	RXB CGFI S	RXG OCTI S	RXG BOC TIS	RXG BFR MIS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
RXGBFRM IS	0	r	MMC Receive Good Bad Frame Counter Interrupt Status This bit is set when the rxframecount_bg counter reaches half of the maximum value or the maximum value.
RXGBOCT IS	1	r	MMC Receive Good Bad Octet Counter Interrupt Status This bit is set when the rxoctetcount_bg counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RXGOCTIS	2	r	MMC Receive Good Octet Counter Interrupt Status. This bit is set when the rxoctetcount_g counter reaches half of the maximum value or the maximum value.
RXBCGFIS	3	r	MMC Receive Broadcast Good Frame Counter Interrupt Status. This bit is set when the rxbroadcastframes_g counter reaches half of the maximum value or the maximum value.
RXMCGFIS	4	r	MMC Receive Multicast Good Frame Counter Interrupt Status This bit is set when the rxmulticastframes_g counter reaches half of the maximum value or the maximum value.
RXCRCERFIS	5	r	MMC Receive CRC Error Frame Counter Interrupt Status This bit is set when the rxrcrcerror counter reaches half of the maximum value or the maximum value.
RXALGNERFIS	6	r	MMC Receive Alignment Error Frame Counter Interrupt Status This bit is set when the rxalignmenterror counter reaches half of the maximum value or the maximum value.
RXRUNTFIS	7	r	MMC Receive Runt Frame Counter Interrupt Status This bit is set when the rxrunterror counter reaches half of the maximum value or the maximum value.
RXJABERFIS	8	r	MMC Receive Jabber Error Frame Counter Interrupt Status This bit is set when the rxjabbererror counter reaches half of the maximum value or the maximum value.
RXUSIZEGFIS	9	r	MMC Receive Undersize Good Frame Counter Interrupt Status This bit is set when the rxundersize_g counter reaches half of the maximum value or the maximum value.
RXOSIZEGFIS	10	r	MMC Receive Oversize Good Frame Counter Interrupt Status This bit is set when the rxoversize_g counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RX64OCTGBFIS	11	r	MMC Receive 64 Octet Good Bad Frame Counter Interrupt Status This bit is set when the rx64octets_gb counter reaches half of the maximum value or the maximum value.
RX65T127OCTGBFIS	12	r	MMC Receive 65 to 127 Octet Good Bad Frame Counter Interrupt Status This is set when the rx65to127octets_gb counter reaches half of the maximum value or the maximum value.
RX128T255OCTGBFIS	13	r	MMC Receive 128 to 255 Octet Good Bad Frame Counter Interrupt Status This bit is set when the rx128to255octets_gb counter reaches half of the maximum value or the maximum value.
RX256T511OCTGBFIS	14	r	MMC Receive 256 to 511 Octet Good Bad Frame Counter Interrupt Status This bit is set when the rx256to511octets_gb counter reaches half of the maximum value or the maximum value.
RX512T1023OCTGBFIS	15	r	MMC Receive 512 to 1023 Octet Good Bad Frame Counter Interrupt Status This bit is set when the rx512to1023octets_gb counter reaches half of the maximum value or the maximum value.
RX1024TMAXOCTGBFIS	16	r	MMC Receive 1024 to Maximum Octet Good Bad Frame Counter Interrupt Status This bit is set when the rx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value.
RXUCGFIS	17	r	MMC Receive Unicast Good Frame Counter Interrupt Status This bit is set when the rxunicastframes_gb counter reaches half of the maximum value or the maximum value.
RXLENERFIS	18	r	MMC Receive Length Error Frame Counter Interrupt Status This bit is set when the rxlengtherror counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RXORAN GEFIS	19	r	MMC Receive Out Of Range Error Frame Counter Interrupt Status This bit is set when the rxoutofrangetype counter reaches half of the maximum value or the maximum value.
RXPAUSFI S	20	r	MMC Receive Pause Frame Counter Interrupt Status This bit is set when the rxpauseframe counter reaches half of the maximum value or the maximum value.
RXFOVFIS	21	r	MMC Receive FIFO Overflow Frame Counter Interrupt Status This bit is set when the rxfifooverflow counter reaches half of the maximum value or the maximum value.
RXVLANG BFIS	22	r	MMC Receive VLAN Good Bad Frame Counter Interrupt Status This bit is set when the rxvlanframes_gb counter reaches half of the maximum value or the maximum value.
RXWDOG FIS	23	r	MMC Receive Watchdog Error Frame Counter Interrupt Status This bit is set when the rxwatchdogerror counter reaches half of the maximum value or the maximum value.
RXRCVER RFIS	24	r	MMC Receive Error Frame Counter Interrupt Status This bit is set when the rxrcverror counter reaches half of the maximum value or the maximum value.
RXCTRLFI S	25	r	MMC Receive Control Frame Counter Interrupt Status This bit is set when the rxctrlframes_g counter reaches half of the maximum value or the maximum value.
RESERVE D_31_26	[31:26]	r	RESERVED_31_26

32-bit Register - MMC_Transmit_Interrupt

The MMC Transmit Interrupt register maintains the interrupts generated when transmit statistic counters reach half of their maximum values (0x8000_0000 for 32-bit counter and 0x8000 for 16-bit counter), and the maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When Counter Stop Rollover is set, then interrupts are set but the counter remains at all-ones. The MMC Transmit Interrupt register is a 32-bit wide register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read in order to clear the interrupt bit.

ETH_MMC_TRANSMIT_INTERRUPT
Register 66 - MMC Transmit Interrupt Register (1108_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED_31_26						TXO SIZE GFIS	TXV LAN GFIS	TXP AUS FIS	TXE XDE FFIS	TXG FRMI S	TXG OCTI S	TXC ARE RFIS	TXE XCO LFIS	TXL ATC OLFI S	TXD EFFI S
r						r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXM COL GFIS	TXS COL GFIS	TXU FLO WER FIS	TXB CGB FIS	TXM CGB FIS	TXU CGB FIS	TX10 24T MAX OCT GBFI	TX51 2T10 23O CTG BFIS	TX25 6T51 1OC TGB FIS	TX12 8T25 5OC TGB FIS	TX65 T127 OCT GBFI S	TX64 OCT GBFI S	TXM CGFI S	TXB CGFI S	TXG BFR MIS	TXG BOC TIS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
TXGBOCTIS	0	r	MMC Transmit Good Bad Octet Counter Interrupt Status This bit is set when the txoctetcount_gb counter reaches half of the maximum value or the maximum value.
TXGBFRMIS	1	r	MMC Transmit Good Bad Frame Counter Interrupt Status This bit is set when the txframecount_gb counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
TXBCGFIS	2	r	MMC Transmit Broadcast Good Frame Counter Interrupt Status This bit is set when the txbroadcastframes_g counter reaches half of the maximum value or the maximum value.
TXMCGFIS	3	r	MMC Transmit Multicast Good Frame Counter Interrupt Status This bit is set when the txmulticastframes_g counter reaches half of the maximum value or the maximum value.
TX64OCTGBFIS	4	r	MMC Transmit 64 Octet Good Bad Frame Counter Interrupt Status. This bit is set when the tx64octets_gb counter reaches half of the maximum value or the maximum value.
TX65T127OCTGBFIS	5	r	MMC Transmit 65 to 127 Octet Good Bad Frame Counter Interrupt Status This bit is set when the tx65to127octets_gb counter reaches half of the maximum value or the maximum value.
TX128T255OCTGBFIS	6	r	MMC Transmit 128 to 255 Octet Good Bad Frame Counter Interrupt Status This bit is set when the tx128to255octets_gb counter reaches half of the maximum value or the maximum value.
TX256T511OCTGBFIS	7	r	MMC Transmit 256 to 511 Octet Good Bad Frame Counter Interrupt Status This bit is set when the tx256to511octets_gb counter reaches half of the maximum value or the maximum value.
TX512T1023OCTGBFIS	8	r	MMC Transmit 512 to 1023 Octet Good Bad Frame Counter Interrupt Status This bit is set when the tx512to1023octets_gb counter reaches half of the maximum value or the maximum value.
TX1024TMAXOCTGBFIS	9	r	MMC Transmit 1024 to Maximum Octet Good Bad Frame Counter Interrupt Status This bit is set when the tx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
TXUCGBFIS	10	r	MMC Transmit Unicast Good Bad Frame Counter Interrupt Status This bit is set when the txunicastframes_gb counter reaches half of the maximum value or the maximum value.
TXMCGBFIS	11	r	MMC Transmit Multicast Good Bad Frame Counter Interrupt Status This bit is set when the txmulticastframes_gb counter reaches half of the maximum value or the maximum value.
TXBCGBFIS	12	r	MMC Transmit Broadcast Good Bad Frame Counter Interrupt Status This bit is set when the txbroadcastframes_gb counter reaches half of the maximum value or the maximum value.
TXUFLOWERFIS	13	r	MMC Transmit Underflow Error Frame Counter Interrupt Status This bit is set when the txunderflowerror counter reaches half of the maximum value or the maximum value.
TXSCOLGFIS	14	r	MMC Transmit Single Collision Good Frame Counter Interrupt Status This bit is set when the txsinglecol_g counter reaches half of the maximum value or the maximum value.
TXMCOLGFIS	15	r	MMC Transmit Multiple Collision Good Frame Counter Interrupt Status This bit is set when the txmulticol_g counter reaches half of the maximum value or the maximum value.
TXDEFFIS	16	r	MMC Transmit Deferred Frame Counter Interrupt Status This bit is set when the txdeferred counter reaches half of the maximum value or the maximum value.
TXLATCOLFIS	17	r	MMC Transmit Late Collision Frame Counter Interrupt Status This bit is set when the txlatecol counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
TXEXCOL FIS	18	r	MMC Transmit Excessive Collision Frame Counter Interrupt Status This bit is set when the txexcesscol counter reaches half of the maximum value or the maximum value.
TXCARER FIS	19	r	MMC Transmit Carrier Error Frame Counter Interrupt Status This bit is set when the txcarriererror counter reaches half of the maximum value or the maximum value.
TXGOCTIS	20	r	MMC Transmit Good Octet Counter Interrupt Status This bit is set when the txoctetcount_g counter reaches half of the maximum value or the maximum value.
TXGFRMIS	21	r	MMC Transmit Good Frame Counter Interrupt Status This bit is set when the txframecount_g counter reaches half of the maximum value or the maximum value.
TXEXDEF FIS	22	r	MMC Transmit Excessive Deferral Frame Counter Interrupt Status This bit is set when the txexcessdef counter reaches half of the maximum value or the maximum value.
TXPAUSFIS	23	r	MMC Transmit Pause Frame Counter Interrupt Status This bit is set when the txpauseframeserror counter reaches half of the maximum value or the maximum value.
TXVLANG FIS	24	r	MMC Transmit VLAN Good Frame Counter Interrupt Status This bit is set when the txvlanframes_g counter reaches half of the maximum value or the maximum value.
TXOSIZEG FIS	25	r	MMC Transmit Oversize Good Frame Counter Interrupt Status This bit is set when the txoversize_g counter reaches half of the maximum value or the maximum value.
RESERVE D_31_26	[31:26]	r	RESERVED_31_26

32-bit Register - MMC_Receive_Interrupt_Mask

ETH_MMC_RECEIVE_INTERRUPT_MASK

- (110C_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED_31_26						RXC TRL FIM	RXR CVE RRFI M	RXW DOG FIM	RXV LAN GBFI M	RXF OVFI M	RXP AUS FIM	RXO RAN GEFI M	RXL ENE RFIM	RXU CGFI M	RX1 024T MAX OCT GBFI
r						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX5 12T1 023O CTG BFIM	RX2 56T5 11O CTG BFIM	RX1 28T2 55O CTG BFIM	RX6 5T12 7OC TGB FIM	RX6 4OC TGB FIM	RXO SIZE GFI M	RXU SIZE GFI M	RXJ ABE RFIM	RXR UNT FIM	RXA LGN ERFI M	RXC RCE RFIM	RXM CGFI M	RXB CGFI M	RXG OCTI M	RXG BOC TIM	RXG BFR MIM
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RXGBFRM IM	0	rw	MMC Receive Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxframecount_gb counter reaches half of the maximum value or the maximum value.
RXGBOCT IM	1	rw	MMC Receive Good Bad Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoctetcount_gb counter reaches half of the maximum value or the maximum value.
RXGOCTI M	2	rw	MMC Receive Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoctetcount_g counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RXBCGFIM	3	rw	MMC Receive Broadcast Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxbroadcastframes_g counter reaches half of the maximum value or the maximum value.
RXMCGFIM	4	rw	MMC Receive Multicast Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxmulticastframes_g counter reaches half of the maximum value or the maximum value.
RXRCRCEFIM	5	rw	MMC Receive CRC Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxrcrcerror counter reaches half of the maximum value or the maximum value.
RXALGNERFIM	6	rw	MMC Receive Alignment Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxalignmenterror counter reaches half of the maximum value or the maximum value.
RXRUNTFIM	7	rw	MMC Receive Runt Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxrunterror counter reaches half of the maximum value or the maximum value.
RXJABERFIM	8	rw	MMC Receive Jabber Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxjabbererror counter reaches half of the maximum value or the maximum value.
RXUSIZEGFIM	9	rw	MMC Receive Undersize Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxundersize_g counter reaches half of the maximum value or the maximum value.
RXOSIZEGFIM	10	rw	MMC Receive Oversize Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxoversize_g counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RX64OCTGBFIM	11	rw	MMC Receive 64 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx64octets_gb counter reaches half of the maximum value or the maximum value.
RX65T127OCTGBFIM	12	rw	MMC Receive 65 to 127 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx65to127octets_gb counter reaches half of the maximum value or the maximum value.
RX128T255OCTGBFIM	13	rw	MMC Receive 128 to 255 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx128to255octets_gb counter reaches half of the maximum value or the maximum value.
RX256T511OCTGBFIM	14	rw	MMC Receive 256 to 511 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx256to511octets_gb counter reaches half of the maximum value or the maximum value.
RX512T1023OCTGBFIM	15	rw	MMC Receive 512 to 1023 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx512to1023octets_gb counter reaches half of the maximum value or the maximum value.
RX1024TMAXOCTGBFIM	16	rw	MMC Receive 1024 to Maximum Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value.
RXUCGFIM	17	rw	MMC Receive Unicast Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxunicastframes_g counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RXLENERFIM	18	rw	MMC Receive Length Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxlengtherror counter reaches half of the maximum value or the maximum value.
RXORANGEFIM	19	rw	MMC Receive Out Of Range Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxoutofrangetype counter reaches half of the maximum value or the maximum value.
RXPAUSFIM	20	rw	MMC Receive Pause Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxpauseframes counter reaches half of the maximum value or the maximum value.
RXFOVFI M	21	rw	MMC Receive FIFO Overflow Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxfifooverflow counter reaches half of the maximum value or the maximum value.
RXVLANGBFIM	22	rw	MMC Receive VLAN Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxvlanframes_gb counter reaches half of the maximum value or the maximum value.
RXWDOG FIM	23	rw	MMC Receive Watchdog Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxwatchdog counter reaches half of the maximum value or the maximum value.
RXRCVERRFIM	24	rw	MMC Receive Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxrcverror error counter reaches half the maximum value, and also when it reaches the maximum value.
RXCTRLFI M	25	rw	MMC Receive Control Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxctrlframes counter reaches half the maximum value, and also when it reaches the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RESERVE D_31_26	[31:26]	r	RESERVED_31_26

32-bit Register - MMC_Transmit_Interrupt_Mask

The MMC Transmit Interrupt Mask register maintains the masks for the interrupts generated when the transmit statistic counters reach half of their maximum value or maximum value. This register is 32-bits wide.

ETH_MMC_TRANSMIT_INTERRUPT_MASK

Register 68 - MMC Transmit Interrupt Mask Register (1110_H) Reset Value: 0000 0000_H

31							30							29							28							27							26							25							24							23							22							21							20							19							18							17							16						
RESERVED_31_26																												TXO SIZE GFI M				TXV LAN GFI M				TXP AUS FIM				TXE XDE FFIM				TXG FRMI M				TXG OCTI M				TXC ARE RFIM				TXE XCO LFIM				TXL ATC OLFI M				TXD EFFI M																																															
r																												rw				rw				rw				rw				rw				rw				rw				rw				rw				rw																																															
15							14							13							12							11							10							9							8							7							6							5							4							3							2							1							0						
TXM COL GFI M				TXS COL GFI M				TXU FLO WER FIM				TXB CGB FIM				TXM CGB FIM				TXU CGB FIM				TX10 24T MAX OCT GBFI				TX51 2T10 23O CTG BFIM				TX25 6T51 10C TGB FIM				TX12 8T25 5OC TGB FIM				TX65 T127 OCT GBFI M				TXM CGFI M				TXB CGFI M				TXG BFR MIM				TXG BOC TIM																																																							
rw				rw				rw				rw				rw				rw				rw				rw				rw				rw				rw				rw				rw				rw																																																											

Field	Bits	Type	Description
TXGBOCTIM	0	rw	MMC Transmit Good Bad Octet Counter Interrupt Mask Setting this bit masks the interrupt when the txoctetcount_gb counter reaches half of the maximum value or the maximum value.
TXGBFRMIM	1	rw	MMC Transmit Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txframecount_gb counter reaches half of the maximum value or the maximum value.
TXBCGFI M	2	rw	MMC Transmit Broadcast Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txbroadcastframes_g counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
TXMCGFIM	3	rw	MMC Transmit Multicast Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticastframes_gb counter reaches half of the maximum value or the maximum value.
TX64OCTGBFIM	4	rw	MMC Transmit 64 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx64octets_gb counter reaches half of the maximum value or the maximum value.
TX65T127OCTGBFIM	5	rw	MMC Transmit 65 to 127 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx65to127octets_gb counter reaches half of the maximum value or the maximum value.
TX128T255OCTGBFIM	6	rw	MMC Transmit 128 to 255 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx128to255octets_gb counter reaches half of the maximum value or the maximum value.
TX256T511OCTGBFIM	7	rw	MMC Transmit 256 to 511 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx256to511octets_gb counter reaches half of the maximum value or the maximum value.
TX512T1023OCTGBFIM	8	rw	MMC Transmit 512 to 1023 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx512to1023octets_gb counter reaches half of the maximum value or the maximum value.
TX1024TMAXOCTGBFIM	9	rw	MMC Transmit 1024 to Maximum Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
TXUCGBF IM	10	rw	MMC Transmit Unicast Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txunicastframes_gb counter reaches half of the maximum value or the maximum value.
TXMCGBF IM	11	rw	MMC Transmit Multicast Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticastframes_gb counter reaches half of the maximum value or the maximum value.
TXBCGBF IM	12	rw	MMC Transmit Broadcast Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txbroadcastframes_gb counter reaches half of the maximum value or the maximum value.
TXUFLOW ERFIM	13	rw	MMC Transmit Underflow Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txunderflowerror counter reaches half of the maximum value or the maximum value.
TXSCOLG FIM	14	rw	MMC Transmit Single Collision Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txsinglecol_g counter reaches half of the maximum value or the maximum value.
TXMCOLG FIM	15	rw	MMC Transmit Multiple Collision Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticol_g counter reaches half of the maximum value or the maximum value.
TXDEFFIM	16	rw	MMC Transmit Deferred Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txdeferred counter reaches half of the maximum value or the maximum value.

Field	Bits	Type	Description
TXLATCO LFIM	17	rw	MMC Transmit Late Collision Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txlatecol counter reaches half of the maximum value or the maximum value.
TXEXCOL FIM	18	rw	MMC Transmit Excessive Collision Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txexcesscol counter reaches half of the maximum value or the maximum value.
TXCARER FIM	19	rw	MMC Transmit Carrier Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txcarriererror counter reaches half of the maximum value or the maximum value.
TXGOCTI M	20	rw	MMC Transmit Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the txoctetcount_g counter reaches half of the maximum value or the maximum value.
TXGFRMI M	21	rw	MMC Transmit Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txframecount_g counter reaches half of the maximum value or the maximum value.
TXEXDEF FIM	22	rw	MMC Transmit Excessive Deferral Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txexcessdef counter reaches half of the maximum value or the maximum value.
TXPAUSFI M	23	rw	MMC Transmit Pause Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txpauseframes counter reaches half of the maximum value or the maximum value.
TXVLANG FIM	24	rw	MMC Transmit VLAN Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txvlanframes_g counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
TXOSIZEG FIM	25	rw	MMC Transmit Oversize Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txoversize_g counter reaches half of the maximum value or the maximum value.
RESERVE D_31_26	[31:26]	r	RESERVED_31_26

32-bit Register - Tx_Octet_Count_Good_Bad

This register maintains the number of bytes transmitted in good and bad frames exclusive of preamble and retried bytes.

ETH_TX_OCTET_COUNT_GOOD_BAD

Register 69 - Transmit Octet Count for Good and Bad Frames (1114_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
TXOCTGB	[31:0]	r	TXOCTGB This field indicates the number of bytes transmitted in good and bad frames exclusive of preamble and retried bytes.

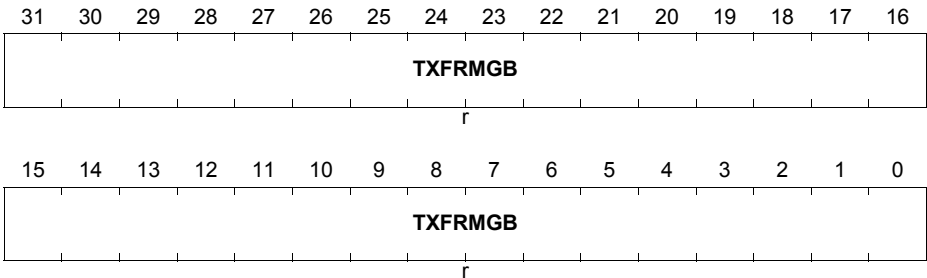
Ethernet MAC (ETH)

32-bit Register - Tx_Frame_Count_Good_Bad

This register maintains the number of good and bad frames transmitted, exclusive of retried frames.

ETH_TX_FRAME_COUNT_GOOD_BAD

Register 70 - Transmit Frame Count for Good and Bad Frames (1118_H) **Reset Value: 0000 0000_H**



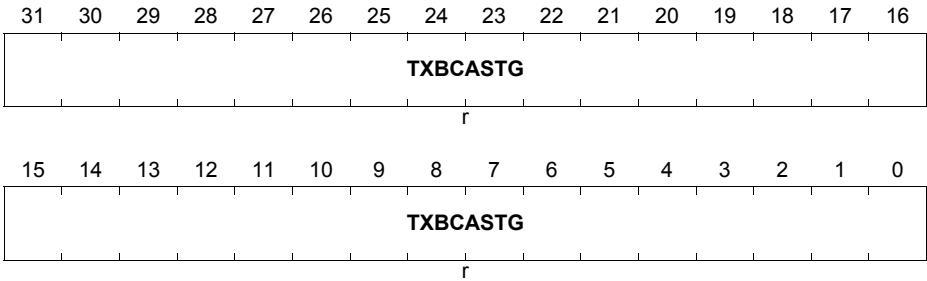
Field	Bits	Type	Description
TXFRMGB	[31:0]	r	TXFRMGB This field indicates the number of good and bad frames transmitted, exclusive of retried frames

32-bit Register - Tx_Broadcast_Frames_Good

This register maintains the number of transmitted good broadcast frames.

ETH_TX_BROADCAST_FRAMES_GOOD

Register 71 - Transmit Frame Count for Good Broadcast Frames (111C_H) **Reset Value: 0000 0000_H**



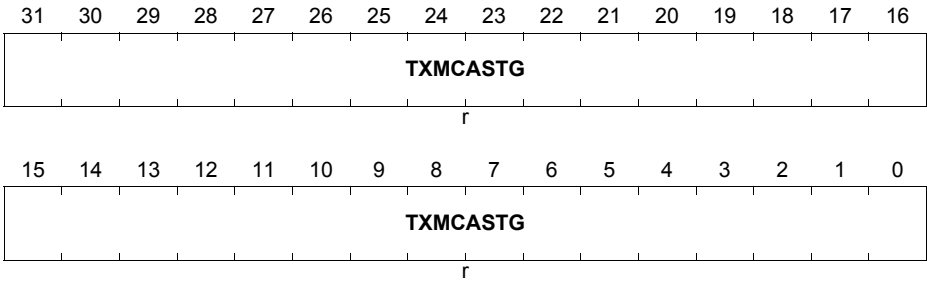
Field	Bits	Type	Description
TXBCASTG	[31:0]	r	TXBCASTG This field indicates the number of transmitted good broadcast frames.

32-bit Register - Tx_Multicast_Frames_Good

This register maintains the number of transmitted good multicast frames.

ETH_TX_MULTICAST_FRAMES_GOOD

Register 72 - Transmit Frame Count for Good Multicast Frames (1120_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
TXMCASTG	[31:0]	r	TXMCASTG This field indicates the number of transmitted good multicast frames.

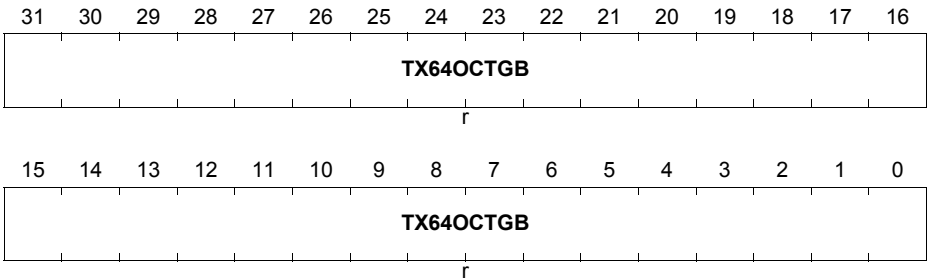
32-bit Register - Tx_64Octets_Frames_Good_Bad

This register maintains the number of transmitted good and bad frames with length of 64 bytes, exclusive of preamble and retried frames.

ETH_TX_64OCTETS_FRAMES_GOOD_BAD

Register 73 - Transmit Octet Count for Good and Bad 64 Byte Frames (1124_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TX64OCTGB	[31:0]	r	TX64OCTGB This field indicates the number of transmitted good and bad frames with length of 64 bytes, exclusive of preamble and retried frames.

32-bit Register - Tx_65To127Octets_Frames_Good_Bad

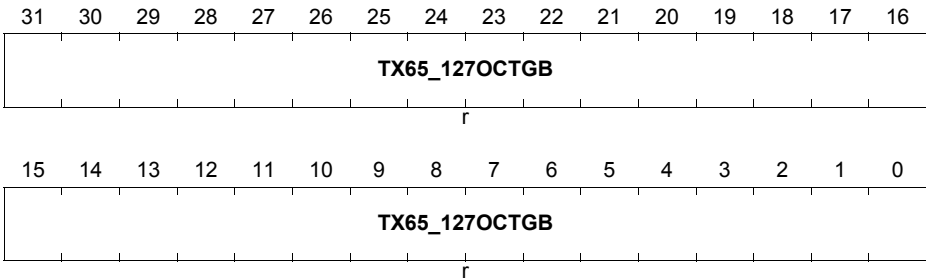
This register maintains the number of transmitted good and bad frames with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried frames.

ETH_TX_65TO127OCTETS_FRAMES_GOOD_BAD

Register 74 - Transmit Octet Count for Good and Bad 65 to 127 Bytes Frames

(1128_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TX65_127 OCTGB	[31:0]	r	TX65_127OCTGB This field indicates the number of transmitted good and bad frames with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried frames.

Ethernet MAC (ETH)

32-bit Register - Tx_128To255Octets_Frames_Good_Bad

This register maintains the number of transmitted good and bad frames with length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried frames.

ETH_TX_128TO255OCTETS_FRAMES_GOOD_BAD

Register 75 - Transmit Octet Count for Good and Bad 128 to 255 Bytes Frames (112C_H)
Reset Value: 0000 0000_H



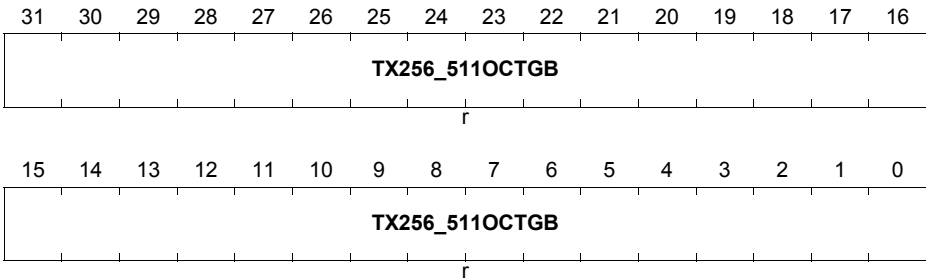
Field	Bits	Type	Description
TX128_255OCTGB	[31:0]	r	TX128_255OCTGB This field indicates the number of transmitted good and bad frames with length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried frames.

32-bit Register - Tx_256To511Octets_Frames_Good_Bad

This register maintains the number of transmitted good and bad frames with length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried frames.

ETH_TX_256TO511OCTETS_FRAMES_GOOD_BAD

Register 76 - Transmit Octet Count for Good and Bad 256 to 511 Bytes Frames (1130_H)
Reset Value: 0000 0000_H



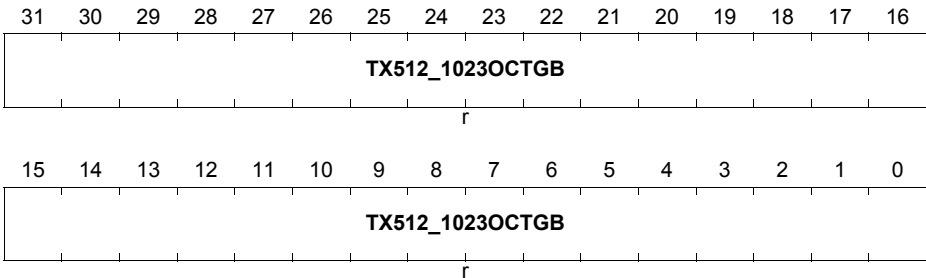
Field	Bits	Type	Description
TX256_511OCTGB	[31:0]	r	TX256_511OCTGB This field indicates the number of transmitted good and bad frames with length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried frames.

32-bit Register - Tx_512To1023Octets_Frames_Good_Bad

This register maintains the number of transmitted good and bad frames with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble and retried frames.

ETH_TX_512TO1023OCTETS_FRAMES_GOOD_BAD

Register 77 - Transmit Octet Count for Good and Bad 512 to 1023 Bytes Frames (1134_H)
Reset Value: 0000 0000_H



Field	Bits	Type	Description
TX512_1023OCTGB	[31:0]	r	TX512_1023OCTGB This field indicates the number of transmitted good and bad frames with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble and retried frames.

Ethernet MAC (ETH)

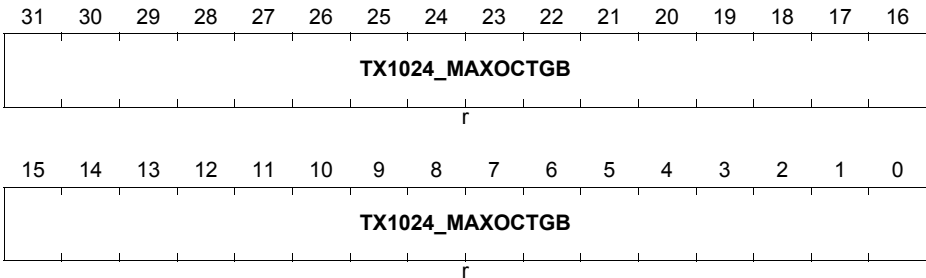
32-bit Register - Tx_1024ToMaxOctets_Frames_Good_Bad

This register maintains the number of transmitted good and bad frames with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble and retried frames.

ETH_TX_1024TOMAXOCTETS_FRAMES_GOOD_BAD

Register 78 - Transmit Octet Count for Good and Bad 1024 to Maxsize Bytes

Frames (1138_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
TX1024_M AXOCTGB	[31:0]	r	TX1024_MAXOCTGB This field indicates the number of good and bad frames transmitted with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble and retried frames.

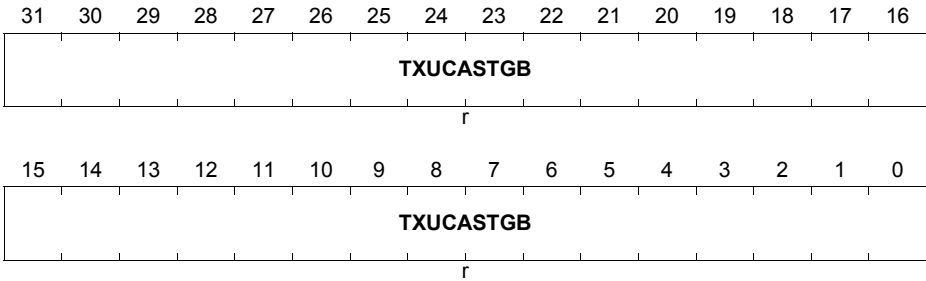
32-bit Register - Tx_Unicast_Frames_Good_Bad

This register maintains the number of transmitted good and bad unicast frames.

ETH_TX_UNICAST_FRAMES_GOOD_BAD

Register 79 - Transmit Frame Count for Good and Bad Unicast Frames (113C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TXUCASTGB	[31:0]	r	TXUCASTGB This field indicates the number of transmitted good and bad unicast frames.

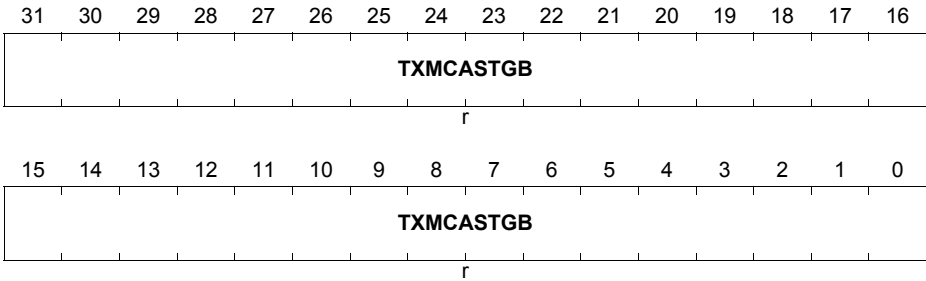
32-bit Register - Tx_Multicast_Frames_Good_Bad

This register maintains the number of transmitted good and bad multicast frames.

ETH_TX_MULTICAST_FRAMES_GOOD_BAD

Register 80 - Transmit Frame Count for Good and Bad Multicast Frames (1140_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TXMCASTGB	[31:0]	r	TXMCASTGB This field indicates the number of transmitted good and bad multicast frames.

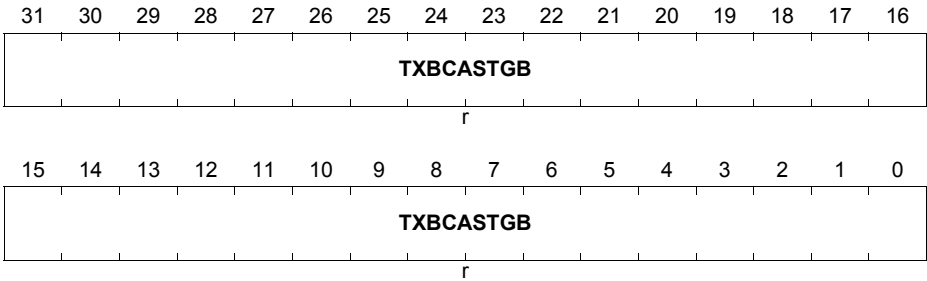
32-bit Register - Tx_Broadcast_Frames_Good_Bad

This register maintains the number of transmitted good and bad broadcast frames.

ETH_TX_BROADCAST_FRAMES_GOOD_BAD

Register 81 - Transmit Frame Count for Good and Bad Broadcast Frames (1144_H)

Reset Value: 0000 0000_H



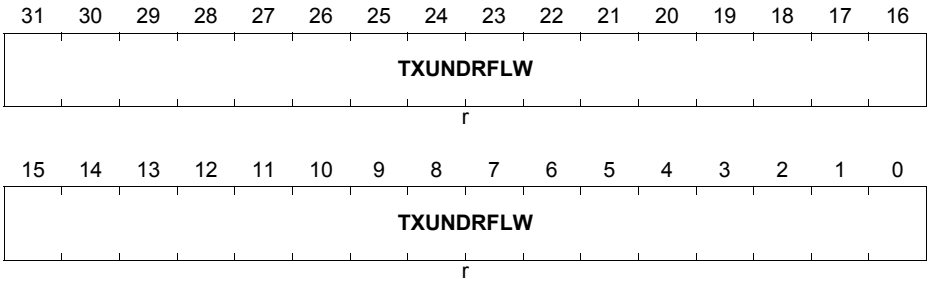
Field	Bits	Type	Description
TXBCAST GB	[31:0]	r	TXBCASTGB This field indicates the number of transmitted good and bad broadcast frames.

32-bit Register - Tx_Underflow_Error_Frames

This register maintains the number of frames aborted because of frame underflow error.

ETH_TX_UNDERFLOW_ERROR_FRAMES

Register 82 - Transmit Frame Count for Underflow Error Frames (1148_H) **Reset Value: 0000 0000_H**



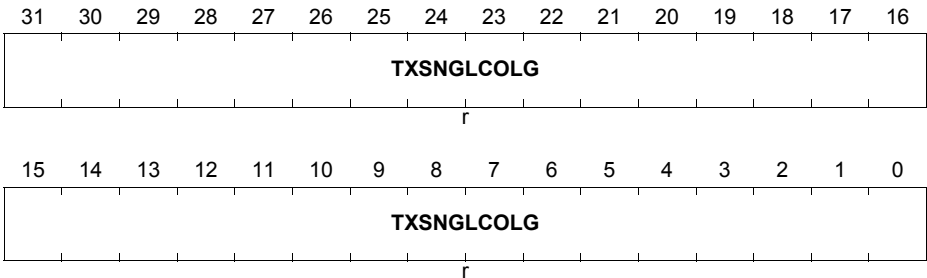
Field	Bits	Type	Description
TXUNDRFLW	[31:0]	r	TXUNDRFLW This field indicates the number of frames aborted because of frame underflow error.

32-bit Register - Tx_Single_Collision_Good_Frames

This register maintains the number of successfully transmitted frames after a single collision in the half-duplex mode.

ETH_TX_SINGLE_COLLISION_GOOD_FRAMES

Register 83 - Transmit Frame Count for Frames Transmitted after Single Collision (114C_H)
Reset Value: 0000 0000_H



Field	Bits	Type	Description
TXSNGLCOLG	[31:0]	r	TXSNGLCOLG This field indicates the number of successfully transmitted frames after a single collision in the half-duplex mode.

32-bit Register - Tx_Multiple_Collision_Good_Frames

This register maintains the number of successfully transmitted frames after multiple collisions in the half-duplex mode.

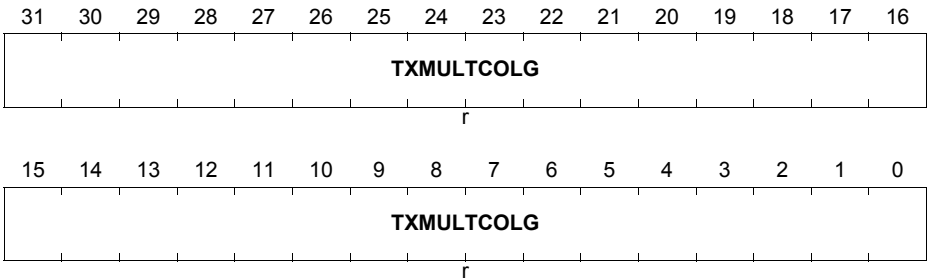
ETH_TX_MULTIPLE_COLLISION_GOOD_FRAMES

Register 84 - Transmit Frame Count for Frames Transmitted after Multiple

Collision

(1150_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TXMULTCOLG	[31:0]	r	TXMULTCOLG This field indicates the number of successfully transmitted frames after multiple collisions in the half-duplex mode.

32-bit Register - Tx_Deferred_Frames

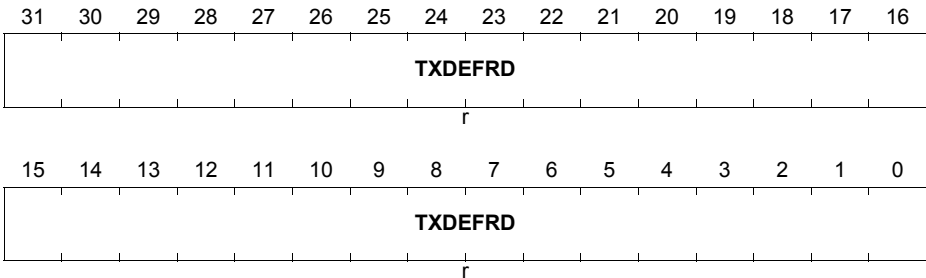
This register maintains the number of successfully transmitted frames after a deferral in the half-duplex mode.

ETH_TX_DEFERRED_FRAMES

Register 85 - Transmit Frame Count for Deferred Frames (1154_H)

Reset Value:

0000 0000_H



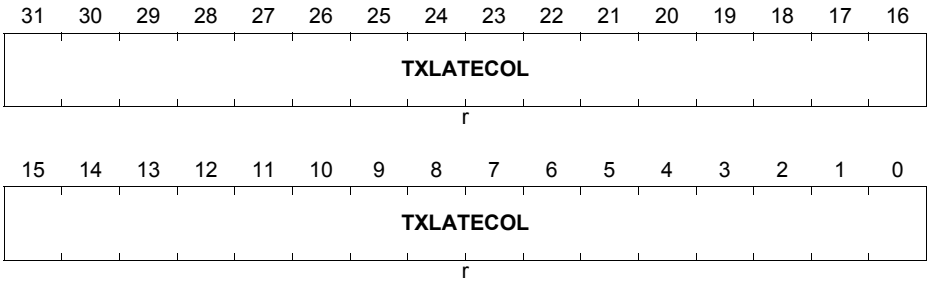
Field	Bits	Type	Description
TXDEFRD	[31:0]	r	TXDEFRD This field indicates the number of successfully transmitted frames after a deferral in the half-duplex mode.

32-bit Register - Tx_Late_Collision_Frames

This register maintains the number of frames aborted because of late collision error.

ETH_TX_LATE_COLLISION_FRAMES

Register 86 - Transmit Frame Count for Late Collision Error Frames (1158_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
TXLATECOL	[31:0]	r	TXLATECOL This field indicates the number of frames aborted because of late collision error.

32-bit Register - Tx_Excessive_Collision_Frames

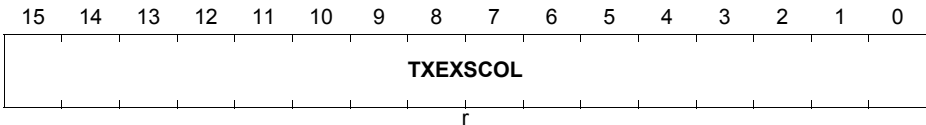
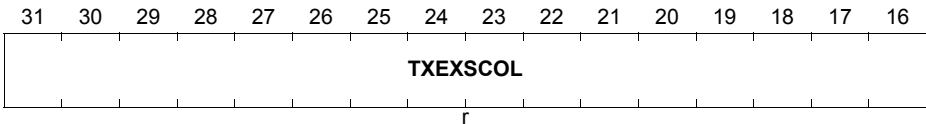
This register maintains the number of frames aborted because of excessive (16) collision error.

ETH_TX_EXCESSIVE_COLLISION_FRAMES

Register 87 - Transmit Frame Count for Excessive Collision Error Frames

(115C_H)

Reset Value: 0000 0000_H



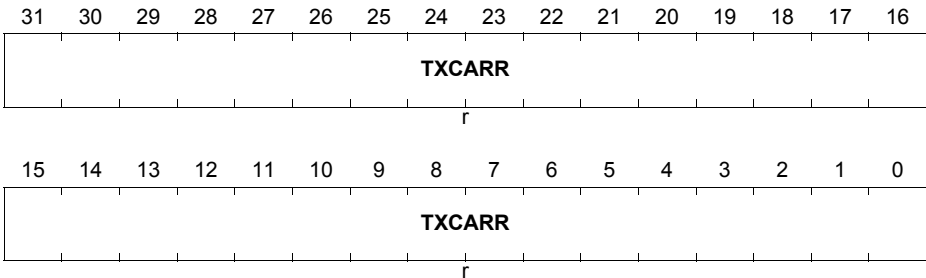
Field	Bits	Type	Description
TXEXSCOL	[31:0]	r	TXEXSCOL This field indicates the number of frames aborted because of excessive (16) collision error.

32-bit Register - Tx_Carrier_Error_Frames

This register maintains the number of frames aborted because of carrier sense error (no carrier or loss of carrier).

ETH_TX_CARRIER_ERROR_FRAMES

Register 88 - Transmit Frame Count for Carrier Sense Error Frames (1160_H) Reset Value: 0000 0000_H



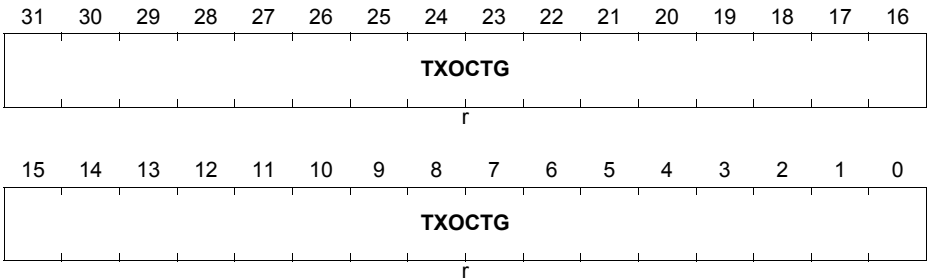
Field	Bits	Type	Description
TXCARR	[31:0]	r	TXCARR This field indicates the number of frames aborted because of carrier sense error (no carrier or loss of carrier).

32-bit Register - Tx_Octet_Count_Good

This register maintains the number of bytes transmitted, exclusive of preamble, in good frames.

ETH_TX_OCTET_COUNT_GOOD

Register 89 - Transmit Octet Count for Good Frames (1164_H) Reset Value: 0000 0000_H



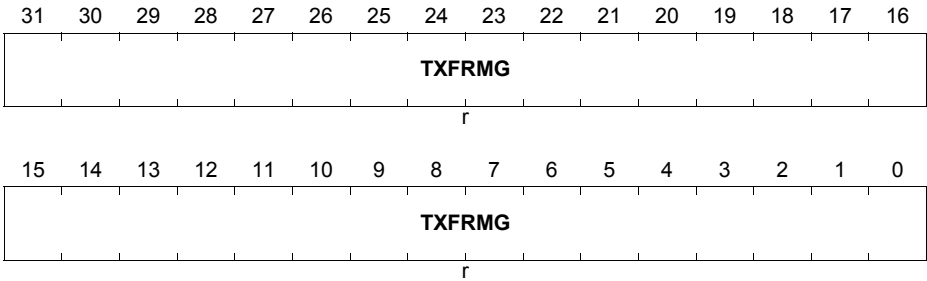
Field	Bits	Type	Description
TXOCTG	[31:0]	r	TXOCTG This field indicates the number of bytes transmitted, exclusive of preamble, in good frames.

32-bit Register - Tx_Frame_Count_Good

This register maintains the number of transmitted good frames, exclusive of preamble.

ETH_TX_FRAME_COUNT_GOOD

Register 90 - Transmit Frame Count for Good Frames (1168_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
TXFRMG	[31:0]	r	TXFRMG This field indicates the number of transmitted good frames, exclusive of preamble.

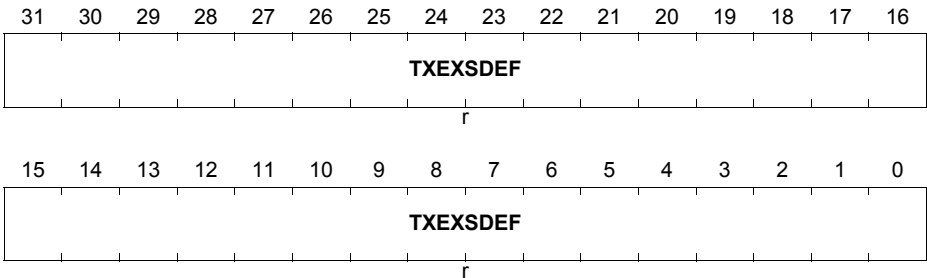
32-bit Register - Tx_Excessive_Deferral_Error

This register maintains the number of frames aborted because of excessive deferral error, that is, frames deferred for more than two max-sized frame times.

ETH_TX_EXCESSIVE_DEFERRAL_ERROR

Register 91 - Transmit Frame Count for Excessive Deferral Error Frames (116C_H)

Reset Value: 0000 0000_H



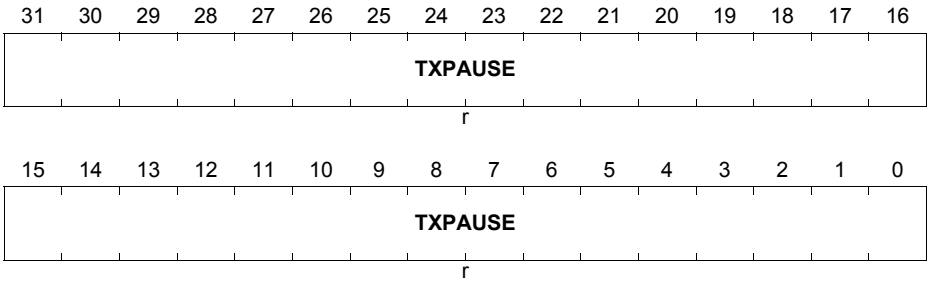
Field	Bits	Type	Description
TXEXSDEF	[31:0]	r	<p>TXEXSDEF</p> <p>This field indicates the number of frames aborted because of excessive deferral error, that is, frames deferred for more than two max-sized frame times.</p>

32-bit Register - Tx_Pause_Frames

This register maintains the number of transmitted good PAUSE frames.

ETH_TX_PAUSE_FRAMES

Register 92 - Transmit Frame Count for Good PAUSE Frames (1170_H) **Reset Value: 0000 0000_H**



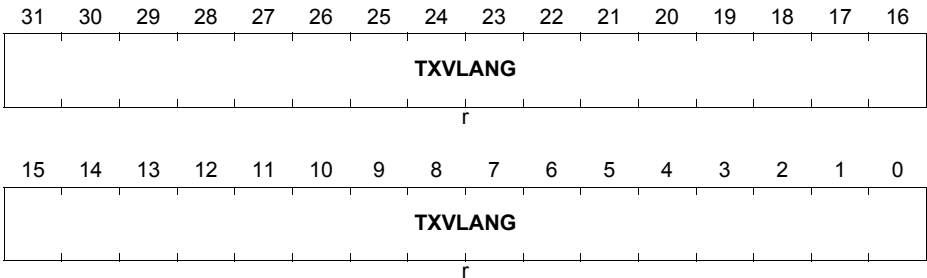
Field	Bits	Type	Description
TXPAUSE	[31:0]	r	TXPAUSE This field indicates the number of transmitted good PAUSE frames.

32-bit Register - Tx_VLAN_Frames_Good

This register maintains the number of transmitted good VLAN frames, exclusive of retried frames.

ETH_TX_VLAN_FRAMES_GOOD

Register 93 - Transmit Frame Count for Good VLAN Frames (1174_H) Reset Value: 0000 0000_H



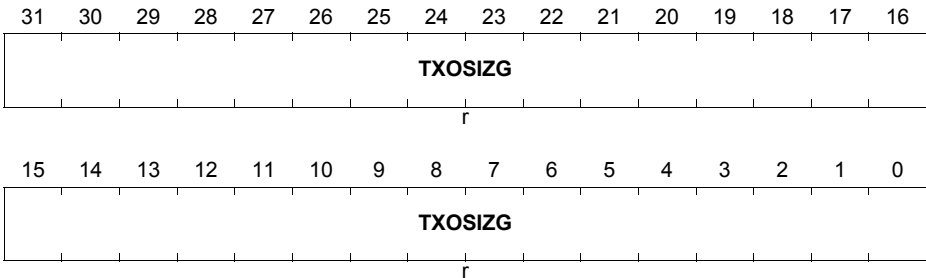
Field	Bits	Type	Description
TXVLANG	[31:0]	r	<p>TXVLANG This register maintains the number of transmitted good VLAN frames, exclusive of retried frames.</p>

32-bit Register - Tx_OSize_Frames_Good

This register maintains the number of transmitted good Oversize frames, exclusive of retried frames.

ETH_TX_OSIZG_FRAMES_GOOD

Register 94 - Transmit Frame Count for Good Oversize Frames (1178_H) **Reset Value: 0000 0000_H**



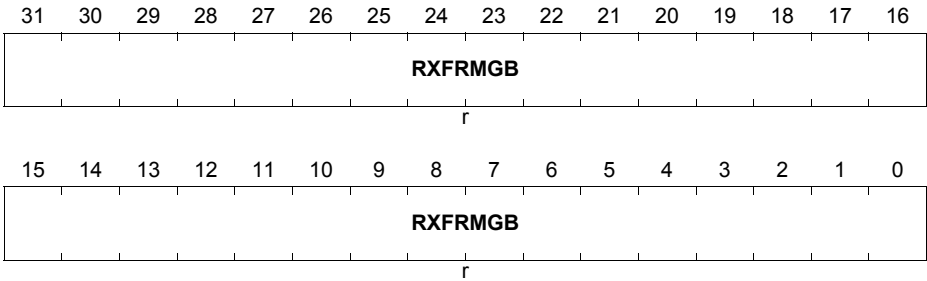
Field	Bits	Type	Description
TXOSIZG	[31:0]	r	TXOSIZG This field indicates the number of frames transmitted without errors and with length greater than the maxsize (1,518 or 1,522 bytes for VLAN tagged frames; 2000 bytes if enabled in bit 27 of Register 0 (MAC Configuration Register)).

32-bit Register - Rx_Frames_Count_Good_Bad

This register maintains the number of received good and bad frames.

ETH_RX_FRAMES_COUNT_GOOD_BAD

Register 96 - Receive Frame Count for Good and Bad Frames (1180_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RXFRMGB	[31:0]	r	RXFRMGB This field indicates the number of received good and bad frames.

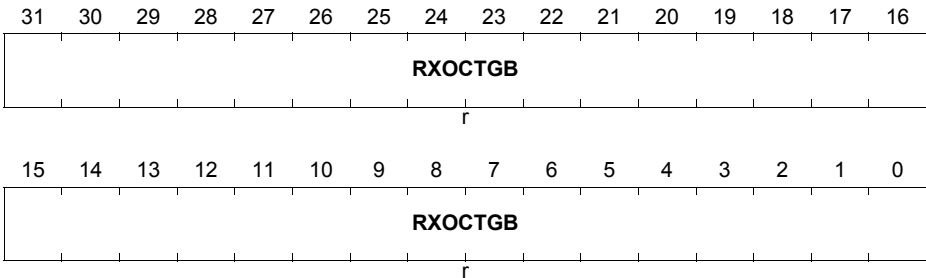
Ethernet MAC (ETH)

32-bit Register - Rx_Octet_Count_Good_Bad

This register maintains the number of bytes received, exclusive of preamble, in good and bad frames.

ETH_RX_OCTET_COUNT_GOOD_BAD

Register 97 - Receive Octet Count for Good and Bad Frames (1184_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
RXOCTGB	[31:0]	r	RXOCTGB This field indicates the number of bytes received, exclusive of preamble, in good and bad frames.

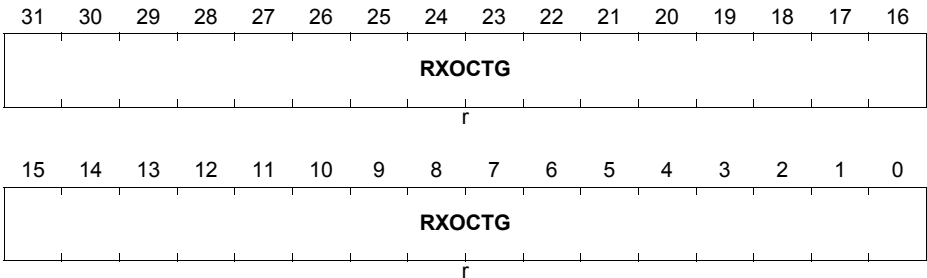
Ethernet MAC (ETH)

32-bit Register - Rx_Octet_Count_Good

This register maintains the number of bytes received, exclusive of preamble, only in good frames.

ETH_RX_OCTET_COUNT_GOOD

Register 98 - Receive Octet Count for Good Frames (1188_H) Reset Value: 0000 0000_H



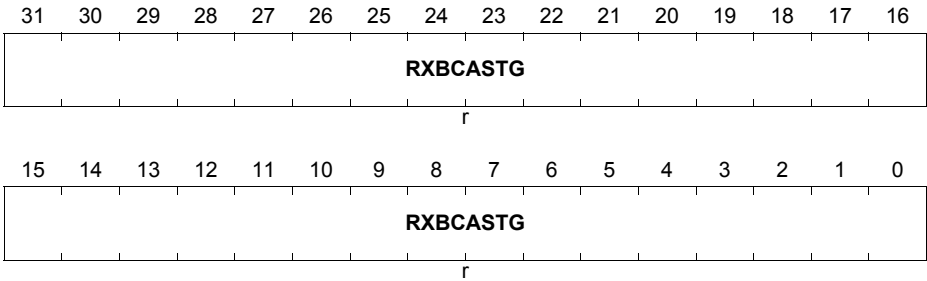
Field	Bits	Type	Description
RXOCTG	[31:0]	r	RXOCTG This field indicates the number of bytes received, exclusive of preamble, only in good frames.

32-bit Register - Rx_Broadcast_Frames_Good

This register maintains the number of received good broadcast frames.

ETH_RX_BROADCAST_FRAMES_GOOD

Register 99 - Receive Frame Count for Good Broadcast Frames (118C_H) **Reset Value: 0000 0000_H**



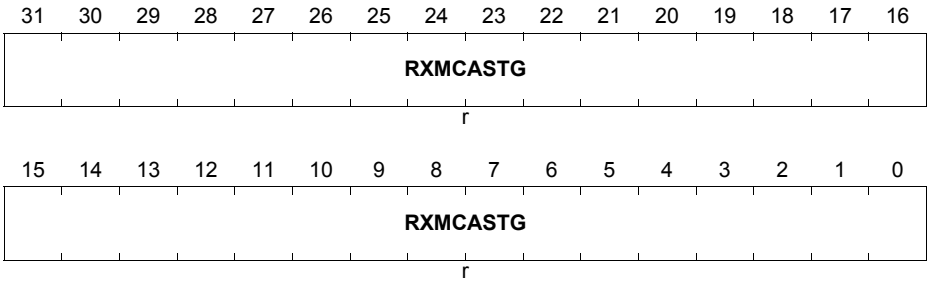
Field	Bits	Type	Description
RXBCASTG	[31:0]	r	RXBCASTG This field indicates the number of received good broadcast frames.

32-bit Register - Rx_Multicast_Frames_Good

This register maintains the number of received good multicast frames.

ETH_RX_MULTICAST_FRAMES_GOOD

Register 100 - Receive Frame Count for Good Multicast Frames (1190_H) **Reset Value: 0000 0000_H**



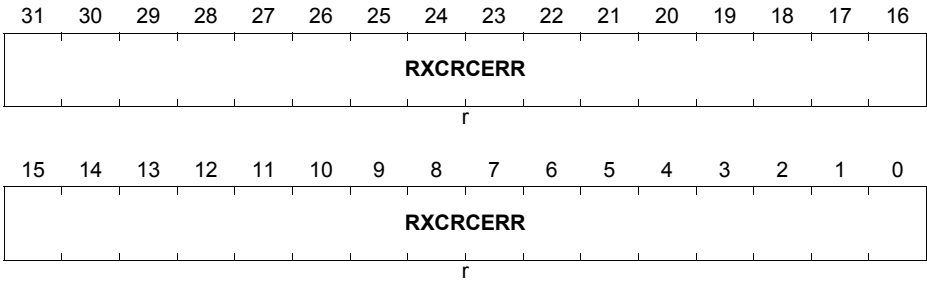
Field	Bits	Type	Description
RXMCASTG	[31:0]	r	RXMCASTG This field indicates the number of received good multicast frames.

32-bit Register - Rx_CRC_Error_Frames

This register maintains the number of frames received with CRC error.

ETH_RX_CRC_ERROR_FRAMES

Register 101 - Receive Frame Count for CRC Error Frames (1194_H) **Reset Value: 0000 0000_H**



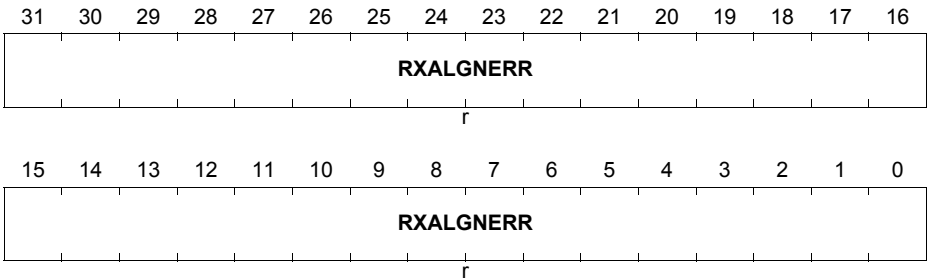
Field	Bits	Type	Description
RXCRCERR	[31:0]	r	RXCRCERR This field indicates the number of frames received with CRC error.

32-bit Register - Rx_Alignment_Error_Frames

This register maintains the number of frames received with alignment (dribble) error. This field is valid only in the 10 or 100 Mbps mode.

ETH_RX_ALIGNMENT_ERROR_FRAMES

Register 102 - Receive Frame Count for Alignment Error Frames (1198_H) Reset Value: 0000 0000_H



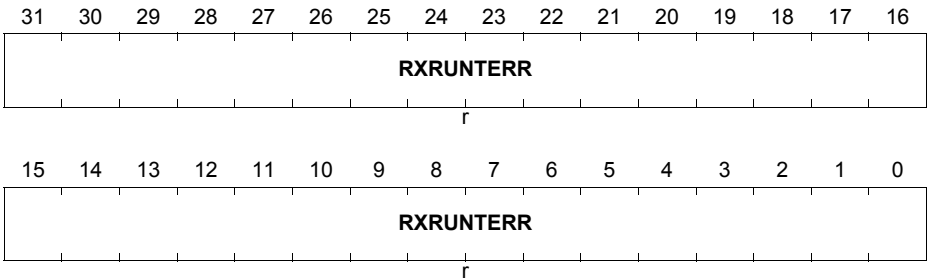
Field	Bits	Type	Description
RXALGNERR	[31:0]	r	RXALGNERR This field indicates the number of frames received with alignment (dribble) error. This field is valid only in the 10 or 100 Mbps mode.

32-bit Register - Rx_Runt_Error_Frames

This register maintains the number of frames received with runt error(<64 bytes and CRC error).

ETH_RX_RUNT_ERROR_FRAMES

Register 103 - Receive Frame Count for Runt Error Frames (119C_H) Reset Value: 0000 0000_H



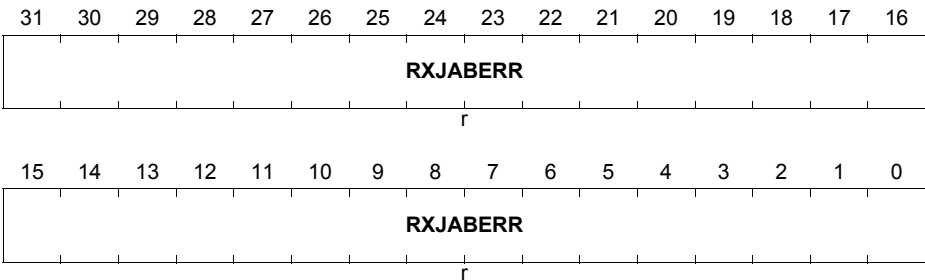
Field	Bits	Type	Description
RXRUNTE RR	[31:0]	r	RXRUNTERR This field indicates the number of frames received with runt error(<64 bytes and CRC error).

32-bit Register - Rx_Jabber_Error_Frames

This register maintains the number of giant frames received with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Frame mode is enabled, then frames of length greater than 9,018 bytes (9,022 for VLAN tagged) are considered as giant frames.

ETH_RX_JABBER_ERROR_FRAMES

Register 104 - Receive Frame Count for Jabber Error Frames (11A0_H) **Reset Value: 0000 0000_H**



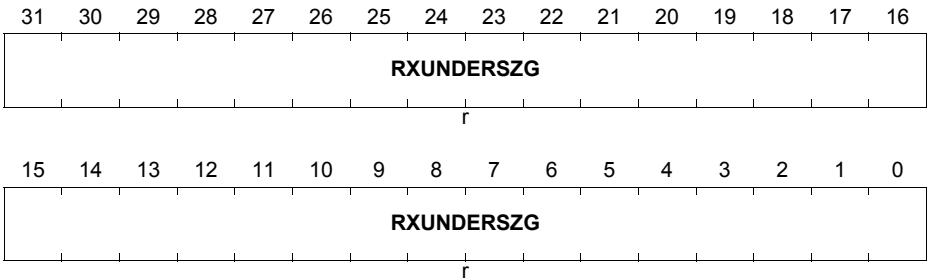
Field	Bits	Type	Description
RXJABERR	[31:0]	r	RXJABERR This field indicates the number of giant frames received with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Frame mode is enabled, then frames of length greater than 9,018 bytes (9,022 for VLAN tagged) are considered as giant frames.

32-bit Register - Rx_Undersize_Frames_Good

This register maintains the number of frames received with length less than 64 bytes and without errors.

ETH_RX_UNDERSIZE_FRAMES_GOOD

Register 105 - Receive Frame Count for Undersize Frames (11A4_H) Reset Value: 0000 0000_H



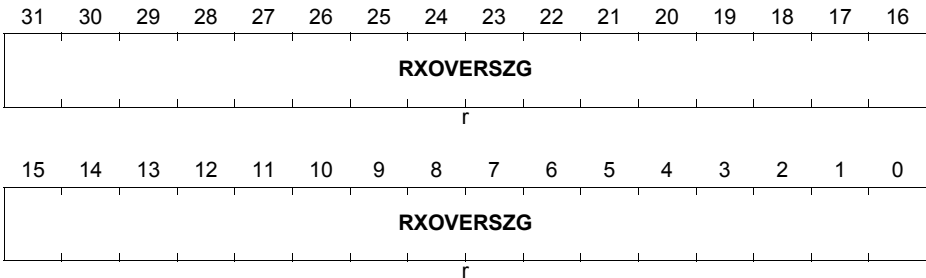
Field	Bits	Type	Description
RXUNDER SZG	[31:0]	r	RXUNDERSZG This field indicates the number of frames received with length less than 64 bytes and without errors.

32-bit Register - Rx_Oversize_Frames_Good

This register maintains the number of frames received with length greater than the maxsize (1,518 or 1,522 for VLAN tagged frames) and without errors.

ETH_RX_OVERSIZE_FRAMES_GOOD

Register 106 - Receive Frame Count for Oversize Frames (11A8_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
RXOVERSZG	[31:0]	r	RXOVERSZG This field indicates the number of frames received without errors, with length greater than the maxsize (1,518 or 1,522 for VLAN tagged frames; 2,000 bytes if enabled in bit 27 of Register 0 (MAC Configuration Register)).

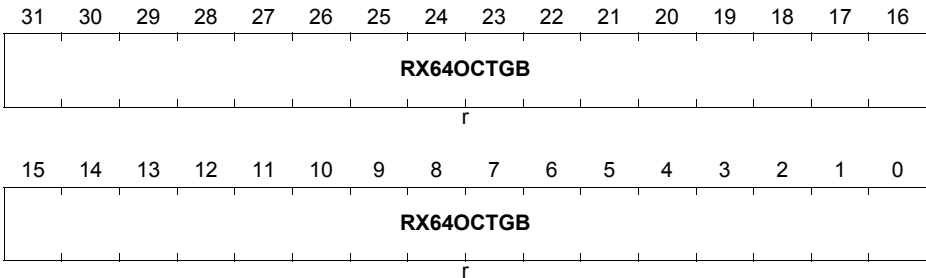
32-bit Register - Rx_64Octets_Frames_Good_Bad

This register maintains the number of received good and bad frames with length 64 bytes, exclusive of preamble.

ETH_RX_64OCTETS_FRAMES_GOOD_BAD

Register 107 - Receive Frame Count for Good and Bad 64 Byte Frames (11AC_H)

Reset Value: 0000 0000_H



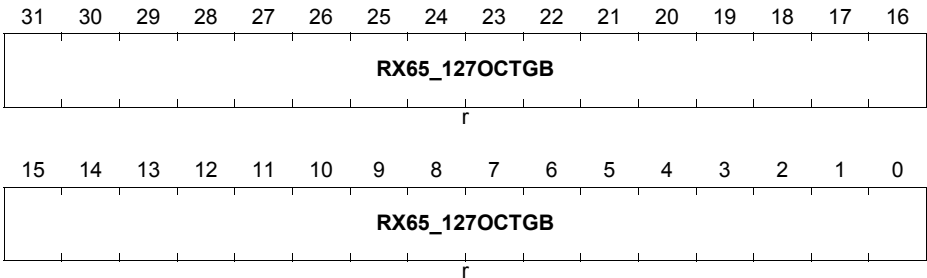
Field	Bits	Type	Description
RX64OCTGB	[31:0]	r	RX64OCTGB This field indicates the number of received good and bad frames with length 64 bytes, exclusive of preamble.

32-bit Register - Rx_65To127Octets_Frames_Good_Bad

This register maintains the number of received good and bad frames received with length between 65 and 127 (inclusive) bytes, exclusive of preamble.

ETH_RX_65TO127OCTETS_FRAMES_GOOD_BAD

Register 108 - Receive Frame Count for Good and Bad 65 to 127 Bytes Frames (11B0_H)
Reset Value: 0000 0000_H



Field	Bits	Type	Description
RX65_127 OCTGB	[31:0]	r	RX65_127OCTGB This field indicates the number of received good and bad frames received with length between 65 and 127 (inclusive) bytes, exclusive of preamble.

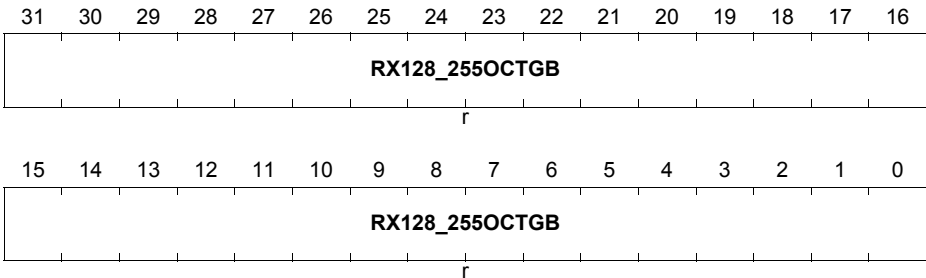
Ethernet MAC (ETH)

32-bit Register - Rx_128To255Octets_Frames_Good_Bad

This register maintains the number of received good and bad frames with length between 128 and 255 (inclusive) bytes, exclusive of preamble.

ETH_RX_128TO255OCTETS_FRAMES_GOOD_BAD

Register 109 - Receive Frame Count for Good and Bad 128 to 255 Bytes Frames (11B4_H)
Reset Value: 0000 0000_H



Field	Bits	Type	Description
RX128_255OCTGB	[31:0]	r	RX128_255OCTGB This field indicates the number of received good and bad frames with length between 128 and 255 (inclusive) bytes, exclusive of preamble.

Ethernet MAC (ETH)

32-bit Register - Rx_256To511Octets_Frames_Good_Bad

This register maintains the number of received good and bad frames with length between 256 and 511 (inclusive) bytes, exclusive of preamble.

ETH_RX_256TO511OCTETS_FRAMES_GOOD_BAD

Register 110 - Receive Frame Count for Good and Bad 256 to 511 Bytes Frames (11B8_H)
Reset Value: 0000 0000_H



Field	Bits	Type	Description
RX256_511OCTGB	[31:0]	r	RX256_511OCTGB This field indicates the number of received good and bad frames with length between 256 and 511 (inclusive) bytes, exclusive of preamble.

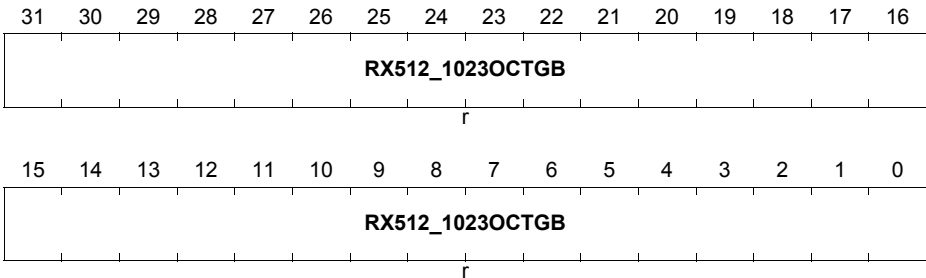
Ethernet MAC (ETH)

32-bit Register - Rx_512To1023Octets_Frames_Good_Bad

This register maintains the number of received good and bad frames with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble.

ETH_RX_512TO1023OCTETS_FRAMES_GOOD_BAD

Register 111 - Receive Frame Count for Good and Bad 512 to 1,023 Bytes Frames (11BC_H)
Reset Value: 0000 0000_H



Field	Bits	Type	Description
RX512_1023OCTGB	[31:0]	r	RX512_1023OCTGB This field indicates the number of received good and bad frames with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble.

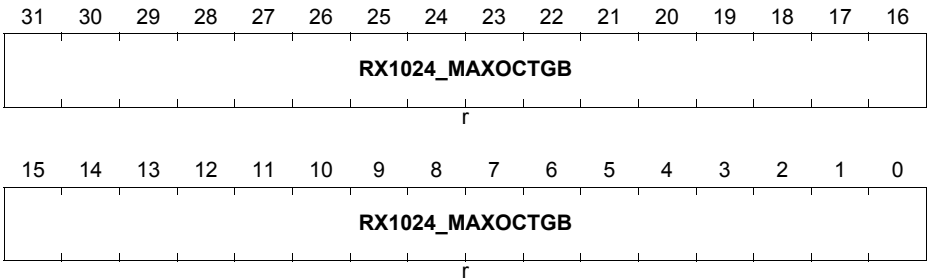
Ethernet MAC (ETH)

32-bit Register - Rx_1024ToMaxOctets_Frames_Good_Bad

This register maintains the number of received good and bad frames with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble.

ETH_RX_1024TOMAXOCTETS_FRAMES_GOOD_BAD

Register 112 - Receive Frame Count for Good and Bad 1,024 to Maxsize Bytes Frames (11C0_H) Reset Value: 0000 0000_H



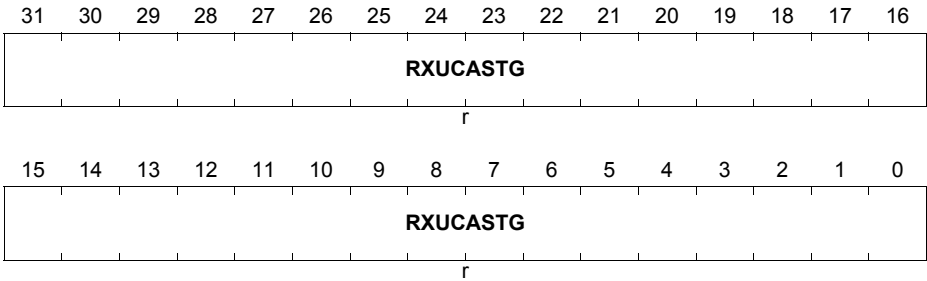
Field	Bits	Type	Description
RX1024_MAXOCTGB	[31:0]	r	RX1024_MAXOCTGB This field indicates the number of received good and bad frames with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble and retried frames.

32-bit Register - Rx_Unicast_Frames_Good

This register maintains the number of received good unicast frames.

ETH_RX_UNICAST_FRAMES_GOOD

Register 113 - Receive Frame Count for Good Unicast Frames (11C4_H) **Reset Value: 0000 0000_H**



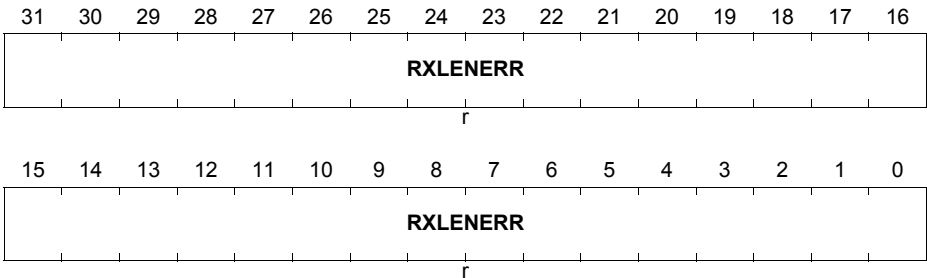
Field	Bits	Type	Description
RXUCASTG	[31:0]	r	RXUCASTG This field indicates the number of received good unicast frames.

32-bit Register - Rx_Length_Error_Frames

This register maintains the number of frames received with length error (Length type field not equal to frame size) for all frames with valid length field.

ETH_RX_LENGTH_ERROR_FRAMES

Register 114 - Receive Frame Count for Length Error Frames (11C8_H) **Reset Value: 0000 0000_H**



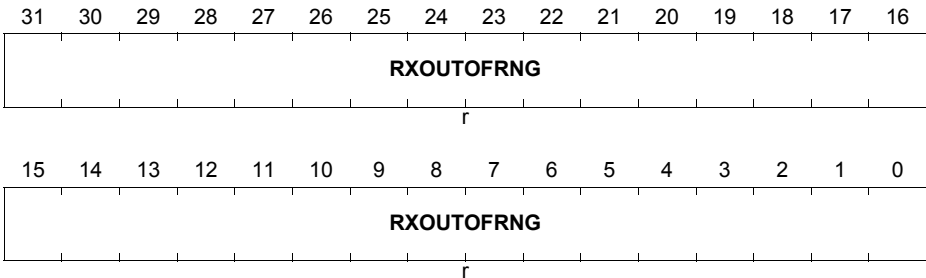
Field	Bits	Type	Description
RXLENER R	[31:0]	r	RXLENERR This field indicates the number of frames received with length error (Length type field not equal to frame size) for all frames with valid length field.

32-bit Register - Rx_Out_Of_Range_Type_Frames

This register maintains the number of received frames with length field not equal to the valid frame size (greater than 1,500 but less than 1,536).

ETH_RX_OUT_OF_RANGE_TYPE_FRAMES

Register 115 - Receive Frame Count for Out of Range Frames (11CC_H) **Reset Value: 0000 0000_H**



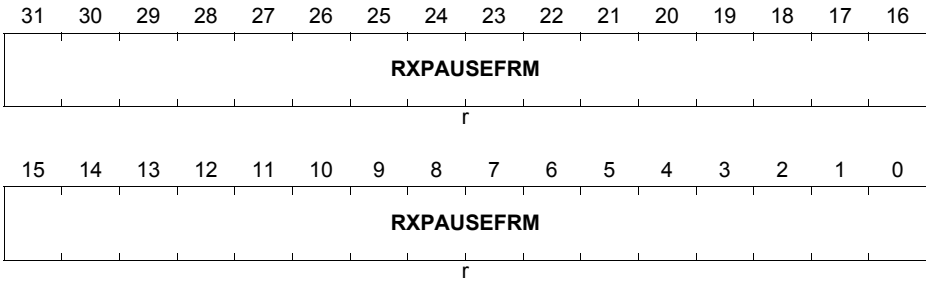
Field	Bits	Type	Description
RXOUTOFRNG	[31:0]	r	RXOUTOFRNG This field indicates the number of received frames with length field not equal to the valid frame size (greater than 1,500 but less than 1,536).

32-bit Register - Rx_Pause_Frames

This register maintains the number of received good and valid PAUSE frames.

ETH_RX_PAUSE_FRAMES

Register 116 - Receive Frame Count for PAUSE Frames (11D0_H) Reset Value: 0000 0000_H



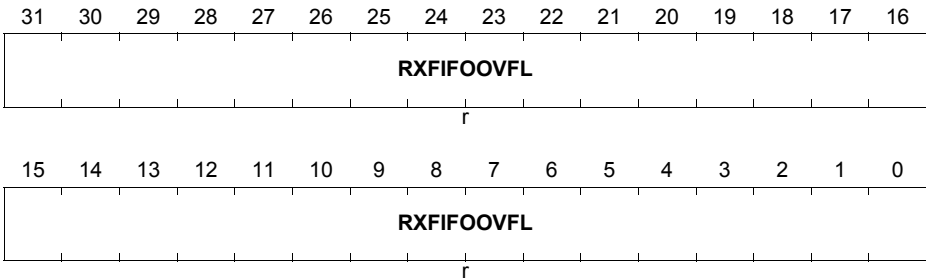
Field	Bits	Type	Description
RXPAUSE FRM	[31:0]	r	RXPAUSEFRM This field indicates the number of received good and valid PAUSE frames.

32-bit Register - Rx_FIFO_Overflow_Frames

This register maintains the number of received frames missed because of FIFO overflow.

ETH_RX_FIFO_OVERFLOW_FRAMES

Register 117 - Receive Frame Count for FIFO Overflow Frames (11D4_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RXFIFOVFL	[31:0]	r	RXFIFOVFL This field indicates the number of received frames missed because of FIFO overflow.

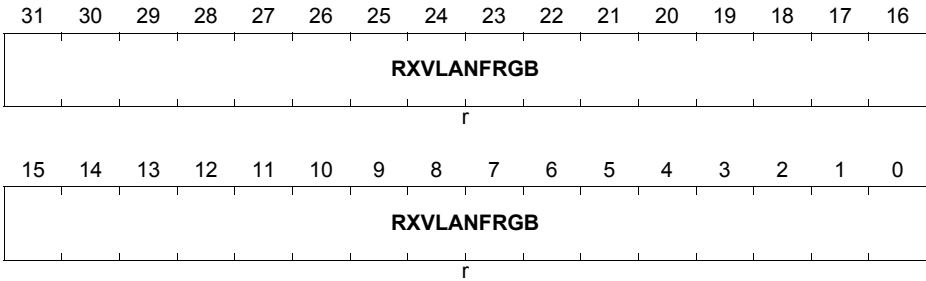
32-bit Register - Rx_VLAN_Frames_Good_Bad

This register maintains the number of received good and bad VLAN frames.

ETH_RX_VLAN_FRAMES_GOOD_BAD

Register 118 - Receive Frame Count for Good and Bad VLAN Frames (11D8_H)

Reset Value: 0000 0000_H



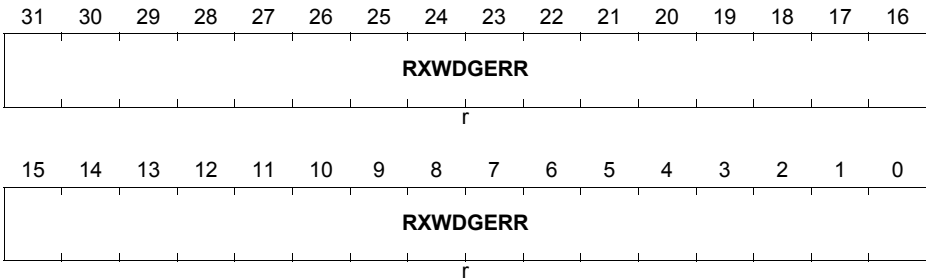
Field	Bits	Type	Description
RXVLANFRGB	[31:0]	r	RXVLANFRGB This field indicates the number of received good and bad VLAN frames.

32-bit Register - Rx_Watchdog_Error_Frames

This register maintains the number of frames received with error because of the watchdog timeout error (frames with more than 2,048 bytes data load).

ETH_RX_WATCHDOG_ERROR_FRAMES

Register 119 - Receive Frame Count for Watchdog Error Frames (11DC_H) Reset Value: 0000 0000_H



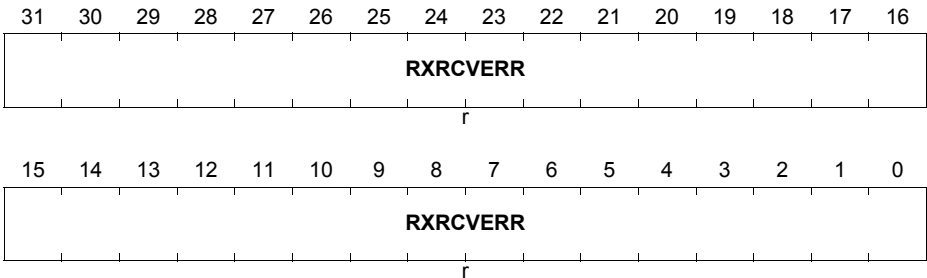
Field	Bits	Type	Description
RXWDGERR	[31:0]	r	RXWDGERR This field indicates the number of frames received with error because of the watchdog timeout error (frames with more than 2,048 bytes data load).

32-bit Register - Rx_Receive_Error_Frames

This register maintains the number of frames received with error because of the GMII/MII RXER error.

ETH_RX_RECEIVE_ERROR_FRAMES

Register 120 - Receive Frame Count for Receive Error Frames (11E0_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RXRCVERR	[31:0]	r	RXRCVERR This field indicates the number of frames received with error because of the watchdog timeout error (frames with more than 2,048 bytes data load).

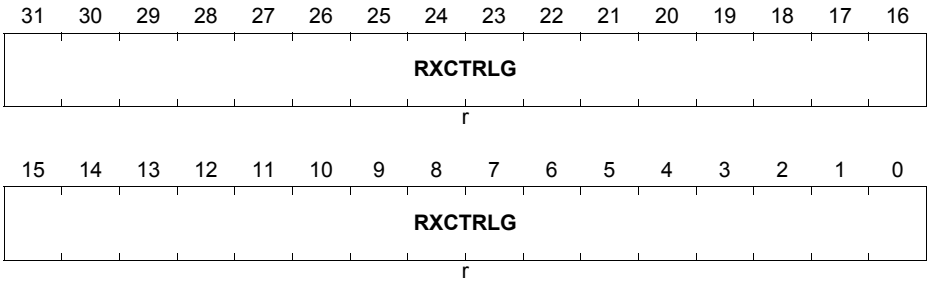
32-bit Register - Rx_Control_Frames_Good

This register maintains the number of good control frames received.

ETH_RX_CONTROL_FRAMES_GOOD

Register 121 - Receive Frame Count for Good Control Frames (11E4_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RXCTRLG	[31:0]	r	RXCTRLG This field indicates the number of frames received with error because of the watchdog timeout error (frames with more than 2,048 bytes data load).

32-bit Register - MMC_IPC_Receive_Interrupt_Mask

This register maintains the mask for the interrupt generated from the receive IPC statistic counters. This register is 32-bits wide. This register is present only when any one of the MMC Receive IPC Counters is selected during core configuration.

ETH_MMC_IPC_RECEIVE_INTERRUPT_MASK
Register 128 - MMC Receive Checksum Offload Interrupt Mask Register (1200_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVE D_31_30	RXIC MPE ROI M	RXIC MPG OIM	RXT CPE ROI M	RXT CPG OIM	RXU DPE ROI M	RXU DPG OIM	RXIP V6N OPA YOI M	RXIP V6H EROI M	RXIP V6G OIM	RXIP V4U DSB LOI M	RXIP V4F RAG OIM	RXIP V4N OPA YOI M	RXIP V4H EROI M	RXIP V4G OIM	
r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVE D_15_14	RXIC MPE RFIM	RXIC MPG FIM	RXT CPE RFIM	RXT CPG FIM	RXU DPE RFIM	RXU DPG FIM	RXIP V6N OPA YFIM	RXIP V6H ERFI M	RXIP V6G FIM	RXIP V4U DSB LFIM	RXIP V4F RAG FIM	RXIP V4N OPA YFIM	RXIP V4H ERFI M	RXIP V4G FIM	
r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
RXIPV4GFI M	0	rw	MMC Receive IPV4 Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_gd_frms counter reaches half of the maximum value or the maximum value.
RXIPV4HE RFIM	1	rw	MMC Receive IPV4 Header Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_hdrerr_frms counter reaches half of the maximum value or the maximum value.
RXIPV4NO PAYFIM	2	rw	MMC Receive IPV4 No Payload Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_nopay_frms counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RXIPV4FRAGFIM	3	rw	MMC Receive IPV4 Fragmented Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_frag_frms counter reaches half of the maximum value or the maximum value.
RXIPV4UDSBLFIM	4	rw	MMC Receive IPV4 UDP Checksum Disabled Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_udtbl_frms counter reaches half of the maximum value or the maximum value.
RXIPV6GFIIM	5	rw	MMC Receive IPV6 Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_gd_frms counter reaches half of the maximum value or the maximum value.
RXIPV6HERFIM	6	rw	MMC Receive IPV6 Header Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_hdrerr_frms counter reaches half of the maximum value or the maximum value.
RXIPV6NOPAYFIM	7	rw	MMC Receive IPV6 No Payload Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_nopay_frms counter reaches half of the maximum value or the maximum value.
RXUDPGFIIM	8	rw	MMC Receive UDP Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_gd_frms counter reaches half of the maximum value or the maximum value.
RXUDPERFIM	9	rw	MMC Receive UDP Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_err_frms counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RXTCPGFI M	10	rw	MMC Receive TCP Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_gd_frms counter reaches half of the maximum value or the maximum value.
RXTCPER FIM	11	rw	MMC Receive TCP Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_err_frms counter reaches half of the maximum value or the maximum value.
RXICMPG FIM	12	rw	MMC Receive ICMP Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_gd_frms counter reaches half of the maximum value or the maximum value.
RXICMPE RFIM	13	rw	MMC Receive ICMP Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_err_frms counter reaches half of the maximum value or the maximum value.
RESERVE D_15_14	[15:14]	r	RESERVED_15_14
RXIPV4G OIM	16	rw	MMC Receive IPV4 Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_gd_octets counter reaches half of the maximum value or the maximum value.
RXIPV4HE ROIM	17	rw	MMC Receive IPV4 Header Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_hdrerr_octets counter reaches half of the maximum value or the maximum value.
RXIPV4NO PAYOIM	18	rw	MMC Receive IPV4 No Payload Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_nopay_octets counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RXIPV4FR AGOIM	19	rw	MMC Receive IPV4 Fragmented Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_frag_octets counter reaches half of the maximum value or the maximum value.
RXIPV4UD SBLOIM	20	rw	MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_udtbl_octets counter reaches half of the maximum value or the maximum value.
RXIPV6G OIM	21	rw	MMC Receive IPV6 Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_gd_octets counter reaches half of the maximum value or the maximum value.
RXIPV6HE ROIM	22	rw	MMC Receive IPV6 Header Error Octet Counter Interrupt Mask Setting this bit masks interrupt when the rxipv6_hdrerr_octets counter reaches half of the maximum value or the maximum value.
RXIPV6NO PAYOIM	23	rw	MMC Receive IPV6 No Payload Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_nopay_octets counter reaches half of the maximum value or the maximum value.
RXUDPGO IM	24	rw	MMC Receive UDP Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_gd_octets counter reaches half of the maximum value or the maximum value.
RXUDPER OIM	25	rw	MMC Receive UDP Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_err_octets counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RXTCPGO IM	26	rw	MMC Receive TCP Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_gd_octets counter reaches half of the maximum value or the maximum value.
RXTCPER OIM	27	rw	MMC Receive TCP Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_err_octets counter reaches half of the maximum value or the maximum value.
RXICMPG OIM	28	rw	MMC Receive ICMP Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_gd_octets counter reaches half of the maximum value or the maximum value.
RXICMPE ROIM	29	rw	MMC Receive ICMP Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_err_octets counter reaches half of the maximum value or the maximum value.
RESERVE D_31_30	[31:30]	r	RESERVED_31_30

32-bit Register - MMC_IPC_Receive_Interrupt

This register maintains the interrupt that the receive IPC statistic counters generate.

ETH_MMC_IPC_RECEIVE_INTERRUPT

Register 130 - MMC Receive Checksum Offload Interrupt Register (1208_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVE D_31_30	RXIC MPE ROIS	RXIC MPG OIS	RXT CPE ROIS	RXT CPG OIS	RXU DPE ROIS	RXU DPG OIS	RXIP V6N OPA YOIS	RXIP V6H EROI S	RXIP V6G OIS	RXIP V4U DSB LOIS	RXIP V4F RAG OIS	RXIP V4N OPA YOIS	RXIP V4H EROI S	RXIP V4G OIS	
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVE D_15_14	RXIC MPE RFIS	RXIC MPG FIS	RXT CPE RFIS	RXT CPG FIS	RXU DPE RFIS	RXU DPG FIS	RXIP V6N OPA YFIS	RXIP V6H ERFI S	RXIP V6G FIS	RXIP V4U DSB LFIS	RXIP V4F RAG FIS	RXIP V4N OPA YFIS	RXIP V4H ERFI S	RXIP V4G FIS	
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
RXIPV4GFI IS	0	r	MMC Receive IPV4 Good Frame Counter Interrupt Status This bit is set when the rxipv4_gd_frms counter reaches half of the maximum value or the maximum value.
RXIPV4HE RFIS	1	r	MMC Receive IPV4 Header Error Frame Counter Interrupt Status This bit is set when the rxipv4_hdrerr_frms counter reaches half of the maximum value or the maximum value.
RXIPV4NO PAYFIS	2	r	MMC Receive IPV4 No Payload Frame Counter Interrupt Status This bit is set when the rxipv4_nopay_frms counter reaches half of the maximum value or the maximum value.
RXIPV4FR AGFIS	3	r	MMC Receive IPV4 Fragmented Frame Counter Interrupt Status This bit is set when the rxipv4_frag_frms counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RXIPV4UD SBLFIS	4	r	MMC Receive IPV4 UDP Checksum Disabled Frame Counter Interrupt Status This bit is set when the rxipv4_udsbl_frms counter reaches half of the maximum value or the maximum value.
RXIPV6GF IS	5	r	MMC Receive IPV6 Good Frame Counter Interrupt Status This bit is set when the rxipv6_gd_frms counter reaches half of the maximum value or the maximum value.
RXIPV6HE RFIS	6	r	MMC Receive IPV6 Header Error Frame Counter Interrupt Status This bit is set when the rxipv6_hdrerr_frms counter reaches half of the maximum value or the maximum value.
RXIPV6NO PAYFIS	7	r	MMC Receive IPV6 No Payload Frame Counter Interrupt Status This bit is set when the rxipv6_nopay_frms counter reaches half of the maximum value or the maximum value.
RXUDPGF IS	8	r	MMC Receive UDP Good Frame Counter Interrupt Status This bit is set when the rxudp_gd_frms counter reaches half of the maximum value or the maximum value.
RXUDPER FIS	9	r	MMC Receive UDP Error Frame Counter Interrupt Status This bit is set when the rxudp_err_frms counter reaches half of the maximum value or the maximum value.
RXTCPGFI S	10	r	MMC Receive TCP Good Frame Counter Interrupt Status This bit is set when the rxtcp_gd_frms counter reaches half of the maximum value or the maximum value.
RXTCPER FIS	11	r	MMC Receive TCP Error Frame Counter Interrupt Status This bit is set when the rxtcp_err_frms counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RXICMPG FIS	12	r	MMC Receive ICMP Good Frame Counter Interrupt Status This bit is set when the rxicmp_gd_frms counter reaches half of the maximum value or the maximum value.
RXICMPE RFIS	13	r	MMC Receive ICMP Error Frame Counter Interrupt Status This bit is set when the rxicmp_err_frms counter reaches half of the maximum value or the maximum value.
RESERVE D_15_14	[15:14]	r	RESERVED_15_14
RXIPV4G OIS	16	r	MMC Receive IPV4 Good Octet Counter Interrupt Status This bit is set when the rxipv4_gd_octets counter reaches half of the maximum value or the maximum value.
RXIPV4HE ROIS	17	r	MMC Receive IPV4 Header Error Octet Counter Interrupt Status This bit is set when the rxipv4_hdrerr_octets counter reaches half of the maximum value or the maximum value.
RXIPV4NO PAYOIS	18	r	MMC Receive IPV4 No Payload Octet Counter Interrupt Status This bit is set when the rxipv4_nopay_octets counter reaches half of the maximum value or the maximum value.
RXIPV4FR AGOIS	19	r	MMC Receive IPV4 Fragmented Octet Counter Interrupt Status This bit is set when the rxipv4_frag_octets counter reaches half of the maximum value or the maximum value.
RXIPV4UD SBLOIS	20	r	MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Status This bit is set when the rxipv4_udsbl_octets counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

Field	Bits	Type	Description
RXIPV6G OIS	21	r	MMC Receive IPV6 Good Octet Counter Interrupt Status This bit is set when the rxipv6_gd_octets counter reaches half of the maximum value or the maximum value.
RXIPV6HE ROIS	22	r	MMC Receive IPV6 Header Error Octet Counter Interrupt Status This bit is set when the rxipv6_hdrerr_octets counter reaches half of the maximum value or the maximum value.
RXIPV6NO PAYOIS	23	r	MMC Receive IPV6 No Payload Octet Counter Interrupt Status This bit is set when the rxipv6_nopay_octets counter reaches half of the maximum value or the maximum value.
RXUDPGO IS	24	r	MMC Receive UDP Good Octet Counter Interrupt Status This bit is set when the rxudp_gd_octets counter reaches half of the maximum value or the maximum value.
RXUDPER OIS	25	r	MMC Receive UDP Error Octet Counter Interrupt Status This bit is set when the rxudp_err_octets counter reaches half the maximum value or the maximum value.
RXTCPGO IS	26	r	MMC Receive TCP Good Octet Counter Interrupt Status This bit is set when the rxtcp_gd_octets counter reaches half the maximum value or the maximum value.
RXTCPER OIS	27	r	MMC Receive TCP Error Octet Counter Interrupt Status This bit is set when the rxtcp_err_octets counter reaches half of the maximum value or the maximum value.
RXICMPG OIS	28	r	MMC Receive ICMP Good Octet Counter Interrupt Status This bit is set when the rxicmp_gd_octets counter reaches half of the maximum value or the maximum value.

Ethernet MAC (ETH)

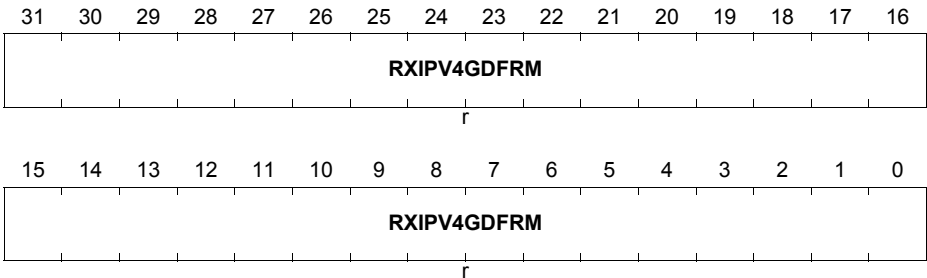
Field	Bits	Type	Description
RXICMPE ROIS	29	r	MMC Receive ICMP Error Octet Counter Interrupt Status This bit is set when the rxicmp_err_octets counter reaches half of the maximum value or the maximum value.
RESERVE D_31_30	[31:30]	r	RESERVED_31_30

32-bit Register - RxIPv4_Good_Frames

This register maintains the number of good IPv4 datagrams received with the TCP, UDP, or ICMP payload.

ETH_RXIPV4_GOOD_FRAMES

Register 132 - Receive IPV4 Good Frame Counter Register (1210_H) Reset Value: 0000 0000_H



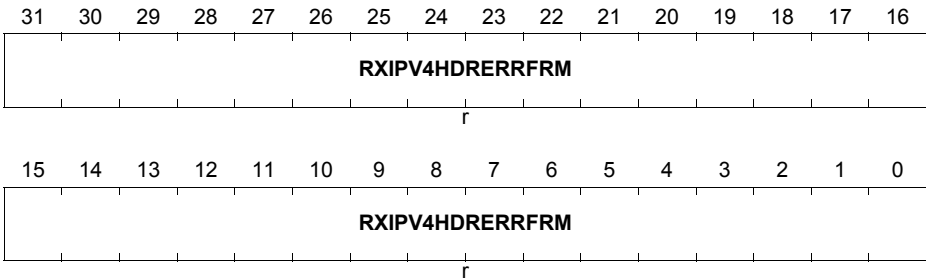
Field	Bits	Type	Description
RXIPV4GDFRM	[31:0]	r	RXIPV4GDFRM This field indicates the number of good IPv4 datagrams received with the TCP, UDP, or ICMP payload.

32-bit Register - RxIPv4_Header_Error_Frames

This register maintains the number of IPv4 datagrams received with header errors (checksum, length, or version mismatch).

ETH_RXIPV4_HEADER_ERROR_FRAMES

Register 133 - Receive IPV4 Header Error Frame Counter Register (1214_H) Reset Value: 0000 0000_H



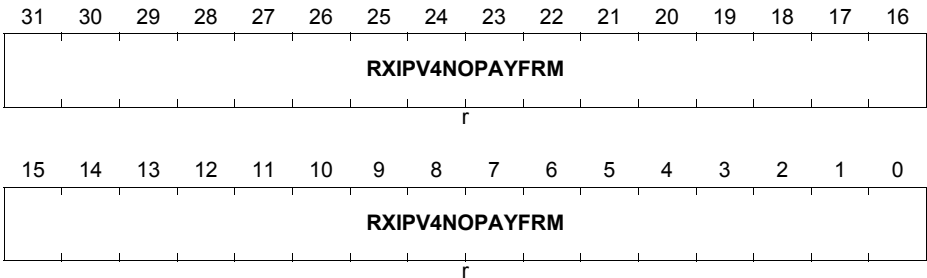
Field	Bits	Type	Description
RXIPV4HDRERRFRM	[31:0]	r	RXIPV4HDRERRFRM This field indicates the number of IPv4 datagrams received with header errors (checksum, length, or version mismatch).

32-bit Register - RxIPv4_No_Payload_Frames

This register maintains the number of received IPv4 datagram frames without a TCP, UDP, or ICMP payload processed by the Checksum engine.

ETH_RXIPV4_NO_PAYLOAD_FRAMES

Register 134 - Receive IPV4 No Payload Frame Counter Register (1218_H) Reset Value: 0000 0000_H



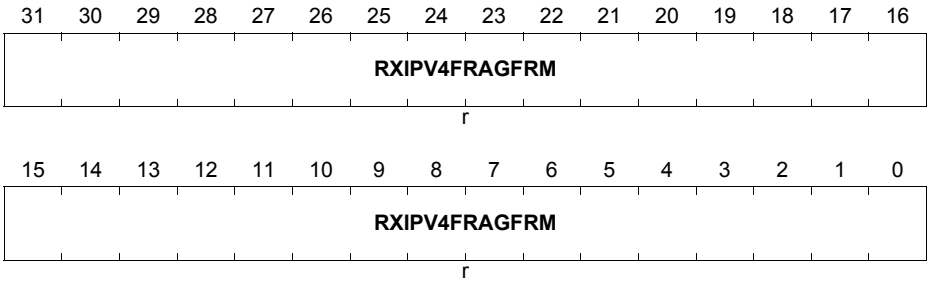
Field	Bits	Type	Description
RXIPV4NO PAYFRM	[31:0]	r	RXIPV4NOPAYFRM This field indicates the number of IPv4 datagram frames received that did not have a TCP, UDP, or ICMP payload processed by the Checksum engine.

32-bit Register - RxIPv4_Fragmented_Frames

This register maintains the number of good IPv4 datagrams received with fragmentation.

ETH_RXIPV4_FRAGMENTED_FRAMES

Register 135 - Receive IPV4 Fragmented Frame Counter Register (121C_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
RXIPV4FRAGFRM	[31:0]	r	RXIPV4FRAGFRM This field indicates the number of good IPv4 datagrams received with fragmentation.

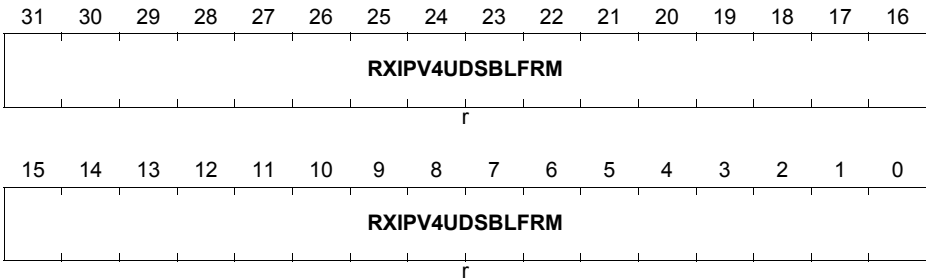
Ethernet MAC (ETH)

32-bit Register - RxIPv4_UDP_Checksum_Disabled_Frames

This register maintains the number of received good IPv4 datagrams which have the UDP payload with checksum disabled.

ETH_RXIPV4_UDP_CHECKSUM_DISABLED_FRAMES

Register 136 - Receive IPV4 UDP Checksum Disabled Frame Counter Register (1220_H)
Reset Value: 0000 0000_H



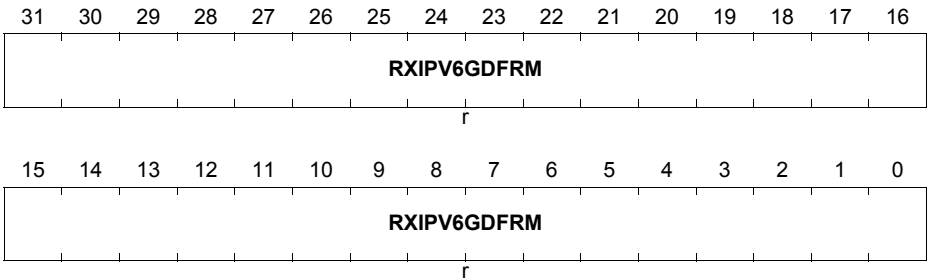
Field	Bits	Type	Description
RXIPV4UDSBLFRM	[31:0]	r	RXIPV4UDSBLFRM This field indicates the number of received good IPv4 datagrams which have the UDP payload with checksum disabled.

32-bit Register - RxIPv6_Good_Frames

This register maintains the number of good IPv6 datagrams received with TCP, UDP, or ICMP payloads.

ETH_RXIPV6_GOOD_FRAMES

Register 137 - Receive IPV6 Good Frame Counter Register (1224_H) Reset Value: 0000 0000_H



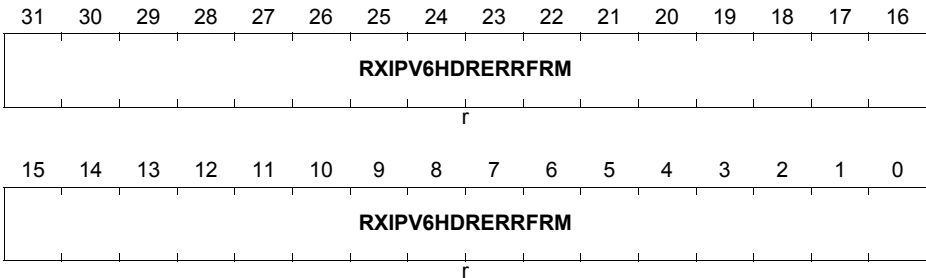
Field	Bits	Type	Description
RXIPV6GDFRM	[31:0]	r	RXIPV6GDFRM This field indicates the number of good IPv6 datagrams received with TCP, UDP, or ICMP payloads.

32-bit Register - RxIPv6_Header_Error_Frames

This register maintains the number of IPv6 datagrams received with header errors (length or version mismatch).

ETH_RXIPV6_HEADER_ERROR_FRAMES

Register 138 - Receive IPV6 Header Error Frame Counter Register (1228_H) Reset Value: 0000 0000_H



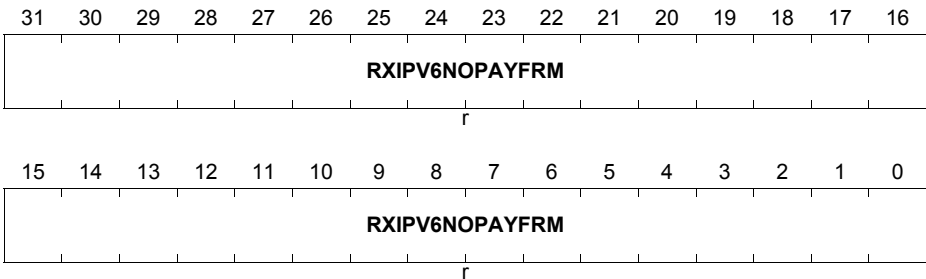
Field	Bits	Type	Description
RXIPV6HDRERRFRM	[31:0]	r	RXIPV6HDRERRFRM This field indicates the number of IPv6 datagrams received with header errors (length or version mismatch).

32-bit Register - RxIPv6_No_Payload_Frames

This register maintains the number of received IPv6 datagram frames without a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers.

ETH_RXIPV6_NO_PAYLOAD_FRAMES

Register 139 - Receive IPV6 No Payload Frame Counter Register (122C_H) **Reset Value: 0000 0000_H**



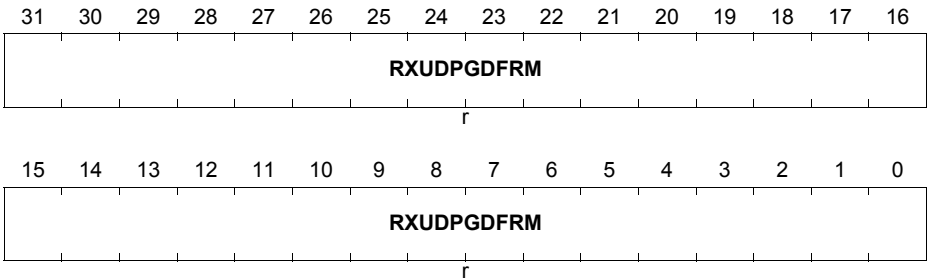
Field	Bits	Type	Description
RXIPV6NO PAYFRM	[31:0]	r	RXIPV6NOPAYFRM This field indicates the number of IPv6 datagram frames received that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers.

32-bit Register - RxUDP_Good_Frames

This register maintains the number of good IP datagrams with a good UDP payload. This counter is not updated when the counter is incremented.

ETH_RXUDP_GOOD_FRAMES

Register 140 - Receive UDP Good Frame Counter Register (1230_H) Reset Value: 0000 0000_H



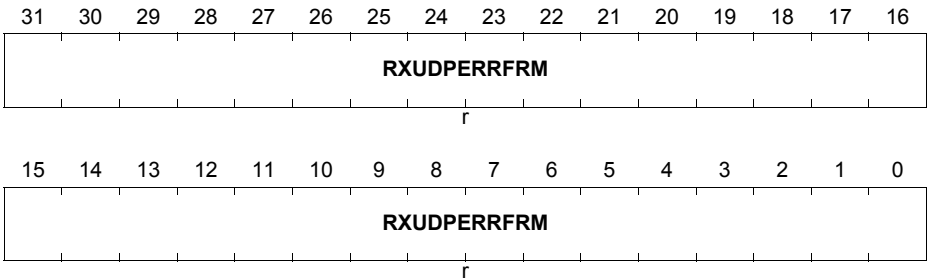
Field	Bits	Type	Description
RXUDPGDFRM	[31:0]	r	RXUDPGDFRM This field indicates the number of good IP datagrams with a good UDP payload. This counter is not updated when the counter is incremented.

32-bit Register - RxUDP_Error_Frames

This register maintains the number of good IP datagrams whose UDP payload has a checksum error.

ETH_RXUDP_ERROR_FRAMES

Register 141 - Receive UDP Error Frame Counter Register (1234_H) Reset Value: 0000 0000_H



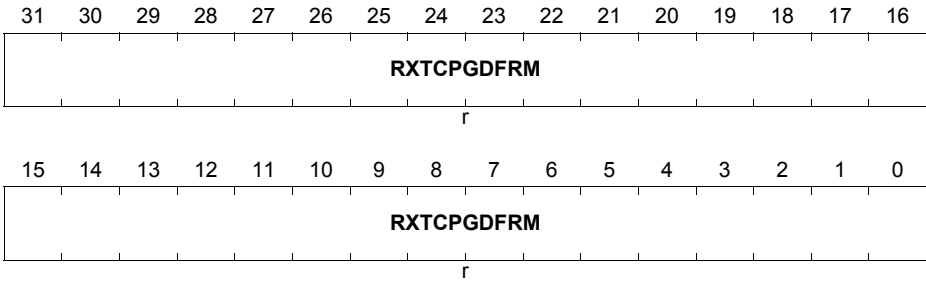
Field	Bits	Type	Description
RXUDPER RFRM	[31:0]	r	RXUDPERFRM This field indicates the number of good IP datagrams whose UDP payload has a checksum error.

32-bit Register - RxTCP_Good_Frames

This register maintains the number of good IP datagrams with a good TCP payload.

ETH_RXTCP_GOOD_FRAMES

Register 142 - Receive TCP Good Frame Counter Register (1238_H) Reset Value: 0000 0000_H



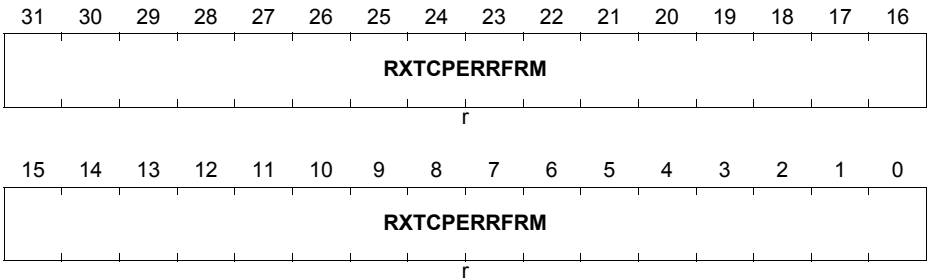
Field	Bits	Type	Description
RXTCPGD FRM	[31:0]	r	RXTCPGDFRM This field indicates the number of good IP datagrams with a good TCP payload.

32-bit Register - RxTCP_Error_Frames

This register maintains the number of good IP datagrams whose TCP payload has a checksum error.

ETH_RXTCP_ERROR_FRAMES

Register 143 - Receive TCP Error Frame Counter Register (123C_H) Reset Value: 0000 0000_H



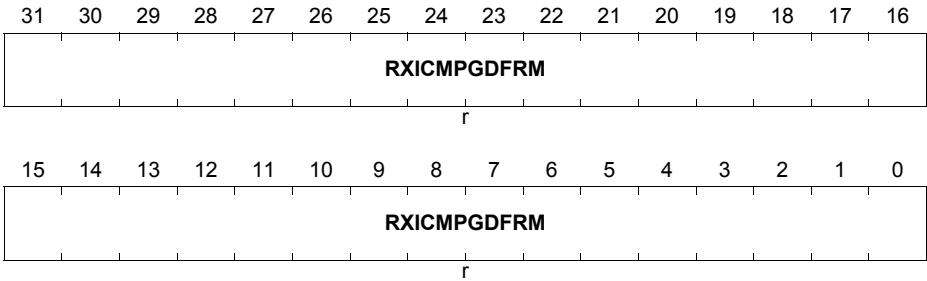
Field	Bits	Type	Description
RXTCPERRFRM	[31:0]	r	RXTCPERRFRM This field indicates the number of good IP datagrams whose TCP payload has a checksum error.

32-bit Register - RxICMP_Good_Frames

This register maintains the number of good IP datagrams with a good ICMP payload.

ETH_RXICMP_GOOD_FRAMES

Register 144 - Receive ICMP Good Frame Counter Register (1240_H) Reset Value: 0000 0000_H



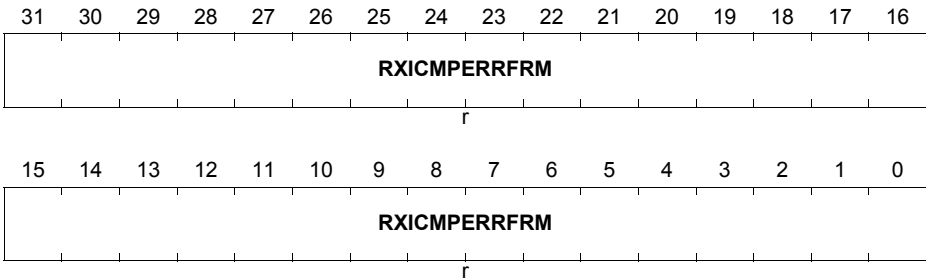
Field	Bits	Type	Description
RXICMPGDFRM	[31:0]	r	RXICMPGDFRM This field indicates the number of good IP datagrams with a good ICMP payload.

32-bit Register - RxICMP_Error_Frames

This register maintains the number of good IP datagrams whose ICMP payload has a checksum error.

ETH_RXICMP_ERROR_FRAMES

Register 145 - Receive ICMP Error Frame Counter Register (1244_H) Reset Value: 0000 0000_H



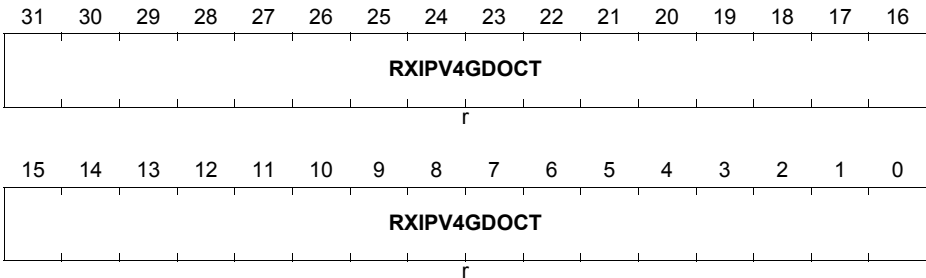
Field	Bits	Type	Description
RXICMPERRFRM	[31:0]	r	RXICMPERRFRM This field indicates the number of good IP datagrams whose ICMP payload has a checksum error.

32-bit Register - RxIPv4_Good_Octets

This register maintains the number of bytes received in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data.

ETH_RXIPV4_GOOD_OCTETS

Register 148 - Receive IPV4 Good Octet Counter Register (1250_H) Reset Value: 0000 0000_H



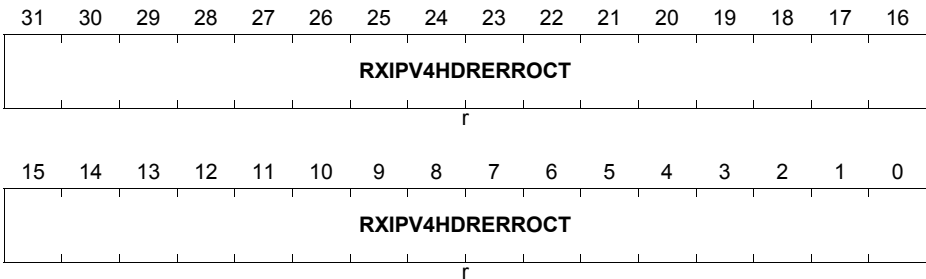
Field	Bits	Type	Description
RXIPV4GD OCT	[31:0]	r	This field indicates the number of bytes received in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

32-bit Register - RxIPv4_Header_Error_Octets

This register maintains the number of bytes received in IPv4 datagrams with header errors (checksum, length, or version mismatch). The value in the Length field of the IPv4 header is used to update this counter.

ETH_RXIPV4_HEADER_ERROR_OCTETS

Register 149 - Receive IPV4 Header Error Octet Counter Register (1254_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RXIPV4HDRERROCT	[31:0]	r	This field indicates the number of bytes received in the IPv4 datagrams with header errors (checksum, length, or version mismatch). The value in the Length field of IPv4 header is used to update this counter. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

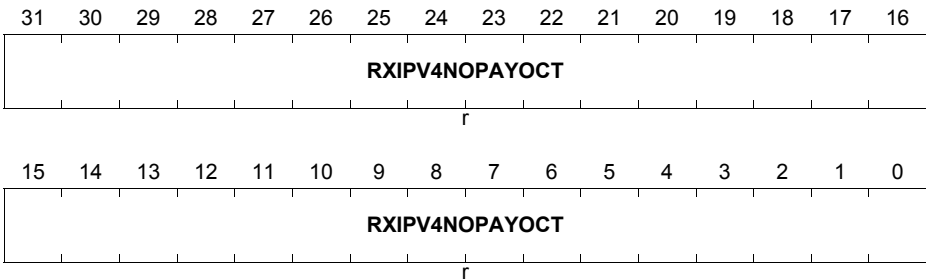
Ethernet MAC (ETH)

32-bit Register - RxIPv4_No_Payload_Octets

This register maintains the number of bytes received in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the IPv4 headers Length field is used to update this counter.

ETH_RXIPV4_NO_PAYLOAD_OCTETS

Register 150 - Receive IPV4 No Payload Octet Counter Register (1258_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RXIPV4NO PAYOCT	[31:0]	r	This field indicates the number of bytes received in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the IPv4 headers Length field is used to update this counter. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

Ethernet MAC (ETH)

32-bit Register - RxIPv4_Fragmented_Octets

This register maintains the number of bytes received in fragmented IPv4 datagrams. The value in the IPv4 headers Length field is used to update this counter.

ETH_RXIPV4_FRAGMENTED_OCTETS

Register 151 - Receive IPV4 Fragmented Octet Counter Register (125C_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
RXIPV4FRAGOCT	[31:0]	r	This field indicates the number of bytes received in fragmented IPv4 datagrams. The value in the IPv4 headers Length field is used to update this counter. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

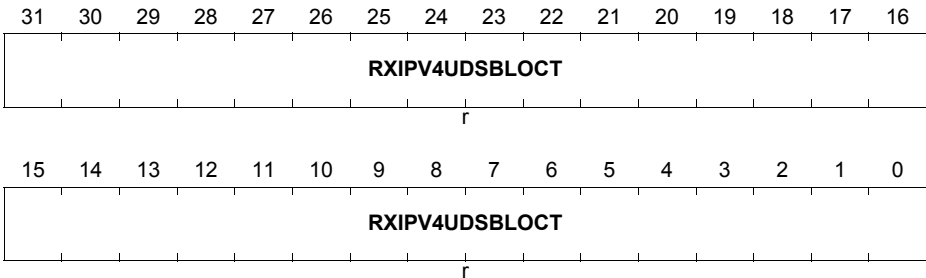
Ethernet MAC (ETH)

32-bit Register - RxIPv4_UDP_Checksum_Disable_Octets

This register maintains the number of bytes received in a UDP segment that had the UDP checksum disabled.

ETH_RXIPV4_UDP_CHECKSUM_DISABLE_OCTETS

Register 152 - Receive IPV4 Fragmented Octet Counter Register (1260_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RXIPV4UDSBLOCT	[31:0]	r	RXIPV4UDSBLOCT This field indicates the number of bytes received in a UDP segment that had the UDP checksum disabled. This counter does not count the IP Header bytes. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

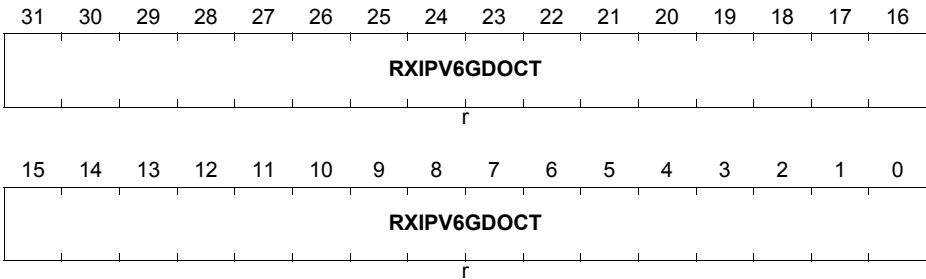
Ethernet MAC (ETH)

32-bit Register - RxIPv6_Good_Octets

This register maintains the number of bytes received in good IPv6 datagrams encapsulating TCP, UDP or ICMPv6 data.

ETH_RXIPV6_GOOD_OCTETS

Register 153 - Receive IPV6 Good Octet Counter Register (1264_H) Reset Value: 0000 0000_H



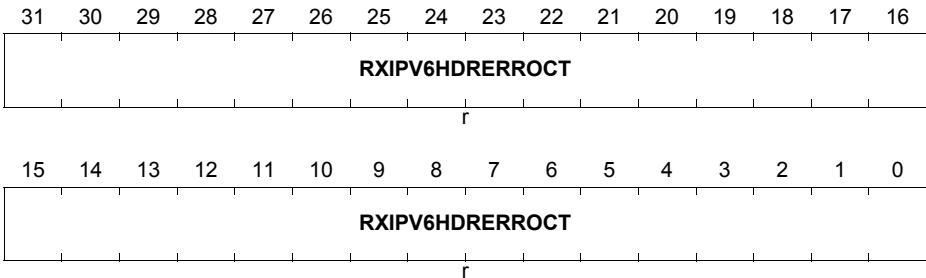
Field	Bits	Type	Description
RXIPV6GD OCT	[31:0]	r	<p>This field indicates the number of bytes received in good IPv6 datagrams encapsulating TCP, UDP or ICMPv6 data.</p> <p>This counter does not count the IP Header bytes. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.</p>

32-bit Register - RxIPv6_Header_Error_Octets

This register maintains the number of bytes received in IPv6 datagrams with header errors (length or version mismatch).

ETH_RXIPV6_HEADER_ERROR_OCTETS

Register 154 - Receive IPv6 Header Error Octet Counter Register (1268_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
RXIPV6HDRERROCT	[31:0]	r	This field indicates the number of bytes received in IPv6 datagrams with header errors (length or version mismatch). The value in the IPv6 headers Length field is used to update this counter. This counter does not count the IP Header bytes. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

32-bit Register - RxIPv6_No_Payload_Octets

This register maintains the number of bytes received in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload.

ETH_RXIPV6_NO_PAYLOAD_OCTETS

Register 155 - Receive IPv6 No Payload Octet Counter Register (126C_H) Reset Value: 0000 0000_H



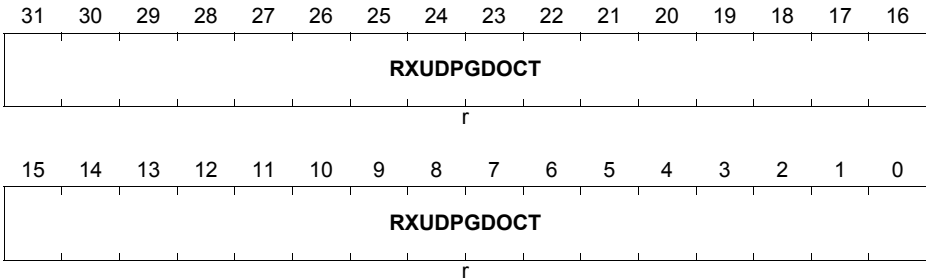
Field	Bits	Type	Description
RXIPV6NO PAYOCT	[31:0]	r	This field indicates the number of bytes received in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the IPv6 headers Length field is used to update this counter. This counter does not count the IP Header bytes. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

32-bit Register - RxUDP_Good_Octets

This register maintains the number of bytes received in a good UDP segment.

ETH_RXUDP_GOOD_OCTETS

Register 156 - Receive UDP Good Octet Counter Register (1270_H) Reset Value: 0000 0000_H



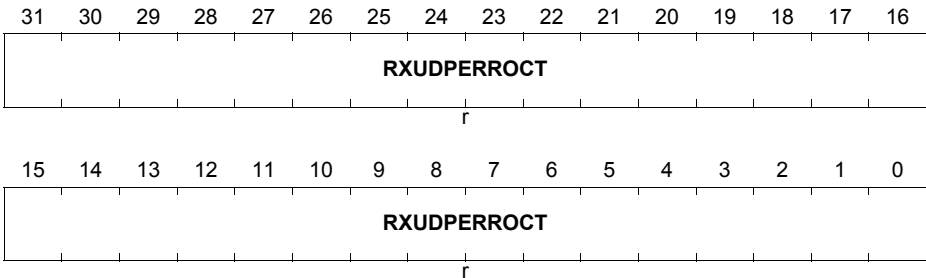
Field	Bits	Type	Description
RXUDPGD OCT	[31:0]	r	RXUDPGDOCT This field indicates the number of bytes received in a good UDP segment. This counter does not count IP header bytes. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

32-bit Register - RxUDP_Error_Octets

This register maintains the number of bytes received in a UDP segment with checksum errors.

ETH_RXUDP_ERROR_OCTETS

Register 157 - Receive UDP Error Octet Counter Register (1274_H) **Reset Value: 0000 0000_H**



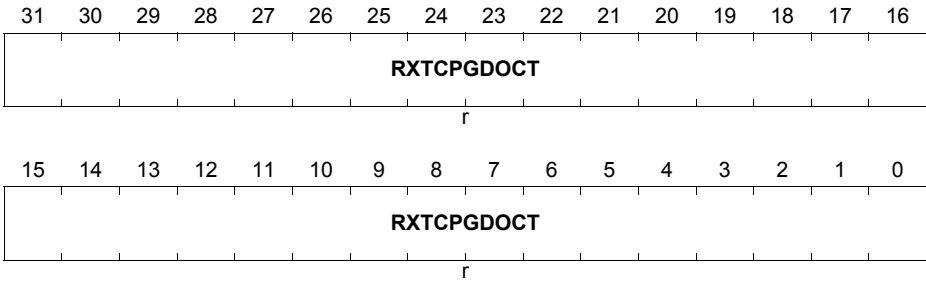
Field	Bits	Type	Description
RXUDPERROCT	[31:0]	r	RXUDPERROCT This field indicates the number of bytes received in a UDP segment with checksum errors. This counter does not count the IP Header bytes. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

32-bit Register - RxTCP_Good_Octets

This register maintains the number of bytes received in a good TCP segment.

ETH_RXTCP_GOOD_OCTETS

Register 158 - Receive TCP Good Octet Counter Register (1278_H) **Reset Value: 0000 0000_H**



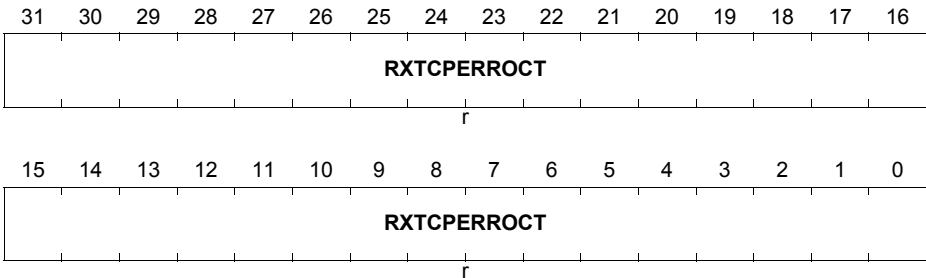
Field	Bits	Type	Description
RXTCPGD OCT	[31:0]	r	RXTCPGDOCT This field indicates the number of bytes received in a good TCP segment. This counter does not count the IP Header bytes. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

32-bit Register - RxTCP_Error_Octets

This register maintains the number of bytes received in a TCP segment with checksum errors.

ETH_RXTCP_ERROR_OCTETS

Register 159 - Receive TCP Error Octet Counter Register (127C_H) **Reset Value: 0000 0000_H**



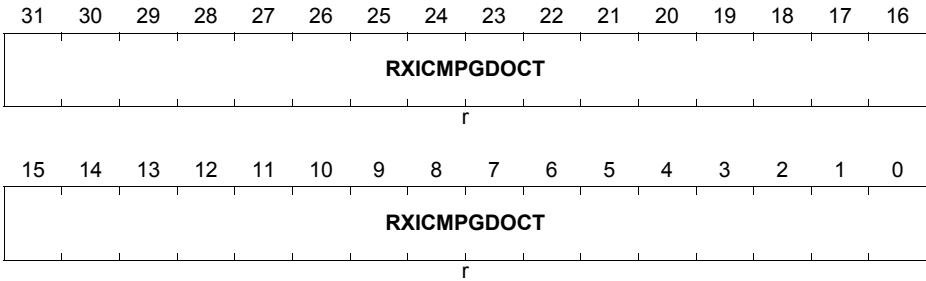
Field	Bits	Type	Description
RXTCPERROCT	[31:0]	r	RXTCPERROCT This field indicates the number of bytes received in a TCP segment with checksum errors. This counter does not count the IP Header bytes. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

32-bit Register - RxICMP_Good_Octets

This register maintains the number of bytes received in a good ICMP segment.

ETH_RXICMP_GOOD_OCTETS

Register 160 - Receive ICMP Good Octet Counter Register (1280_H) **Reset Value: 0000 0000_H**



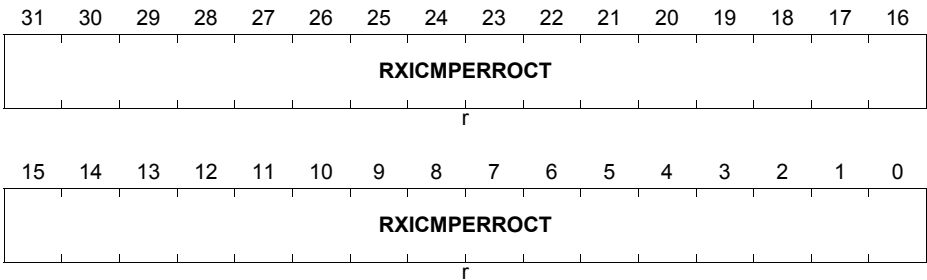
Field	Bits	Type	Description
RXICMPG DOCT	[31:0]	r	RXICMPGDOCT This field indicates the number of bytes received in a good ICMP segment. This counter does not count the IP Header bytes. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

32-bit Register - RxICMP_Error_Octets

This register maintains the number of bytes received in a ICMP segment with checksum errors. This counter does not count the IP Header bytes. The Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.

ETH_RXICMP_ERROR_OCTETS

Register 161 - Receive ICMP Error Octet Counter Register (1284_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
RXICMPERROCT	[31:0]	r	RXICMPERROCT Number of bytes received in an ICMP segment with checksum errors

Ethernet MAC (ETH)
32-bit Register - Timestamp_Control

This register controls the operation of the System Time generator and the processing of PTP packets for timestamping in the Receiver. Note: * Bits[5:1] are reserved when External Timestamp Input feature is enabled. * Bits[19:8] are reserved and read-only when Advanced Timestamp feature is not enabled. * Bits[28:24] are reserved and read-only when Auxiliary Snapshot feature is not enabled. * Release 3.60a onwards, the functions of Bits 17 and 16 (SNAPTYPSEL) have changed. These functions are not backward compatible with the functions described in release 3.50a.

ETH_TIMESTAMP_CONTROL
Register 448 - Timestamp Control Register (1700_H) **Reset Value: 0000 2000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED_31_29			ATS EN3	ATS EN2	ATS EN1	ATS EN0	ATS FC	RESERVED_23_19				TSE NMA CAD DR	SNAPTYP SEL		
r			r	r	r	r	r				rw	rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSM STR ENA	TSE VNT ENA	TSIP V4E NA	TSIP V6E NA	TSIP ENA	TSV ER2 ENA	TSC TRL SSR	TSE NAL L	RESERVE D_7_6	TSA DDR EG	TST RIG	TSU PDT	TSIN IT	TSC FUP DT	TSE NA	
rw	rw	rw	rw	rw	rw	rw	rw	r	rw	rw	rw	rw	rw	rw	

Field	Bits	Type	Description
TSENA	0	rw	Timestamp Enable When set, the timestamp is added for the transmit and receive frames. When disabled, timestamp is not added for the transmit and receive frames and the Timestamp Generator is also suspended. You need to initialize the Timestamp (system time) after enabling this mode. On the receive side, the MAC processes the 1588 frames only if this bit is set.
TSCFUPDT	1	rw	Timestamp Fine or Coarse Update When set, this bit indicates that the system times update should be done using the fine update method. When reset, it indicates the system timestamp update should be done using the Coarse method.

Ethernet MAC (ETH)

Field	Bits	Type	Description
TSINIT	2	rw	<p>Timestamp Initialize</p> <p>When set, the system time is initialized (overwritten) with the value specified in the Register 452 (System Time - Seconds Update Register) and Register 453 (System Time - Nanoseconds Update Register). This bit should be read zero before updating it. This bit is reset when the initialization is complete. The Timestamp Higher Word register (if enabled during core configuration) can only be initialized.</p>
TSUPDT	3	rw	<p>Timestamp Update</p> <p>When set, the system time is updated (added or subtracted) with the value specified in Register 452 (System Time - Seconds Update Register) and Register 453 (System Time - Nanoseconds Update Register). This bit should be read zero before updating it. This bit is reset when the update is completed in hardware. The Timestamp Higher Word register (if enabled during core configuration) is not updated.</p>
TSTRIG	4	rw	<p>Timestamp Interrupt Trigger Enable</p> <p>When set, the timestamp interrupt is generated when the System Time becomes greater than the value written in the Target Time register. This bit is reset after the generation of the Timestamp Trigger Interrupt.</p>
TSADDRE G	5	rw	<p>Addend Reg Update</p> <p>When set, the content of the Timestamp Addend register is updated in the PTP block for fine correction. This is cleared when the update is completed. This register bit should be zero before setting it.</p>
RESERVE D_7_6	[7:6]	r	RESERVED_7_6
TSENALL	8	rw	<p>Enable Timestamp for All Frames</p> <p>When set, the timestamp snapshot is enabled for all frames received by the MAC.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
TSCTRLS SR	9	rw	Timestamp Digital or Binary Rollover Control When set, the Timestamp Low register rolls over after 0x3B9A_C9FF value (that is, 1 nanosecond accuracy) and increments the timestamp (High) seconds. When reset, the rollover value of sub-second register is 0x7FFF_FFFF. The sub-second increment has to be programmed correctly depending on the PTP reference clock frequency and the value of this bit.
TSVER2E NA	10	rw	Enable PTP packet Processing for Version 2 Format When set, the PTP packets are processed using the 1588 version 2 format. Otherwise, the PTP packets are processed using the version 1 format.
TSIPENA	11	rw	Enable Processing of PTP over Ethernet Frames When set, the MAC receiver processes the PTP packets encapsulated directly in the Ethernet frames. When this bit is clear, the MAC ignores the PTP over Ethernet packets.
TSIPV6EN A	12	rw	Enable Processing of PTP Frames Sent Over IPv6-UDP When set, the MAC receiver processes PTP packets encapsulated in UDP over IPv6 packets. When this bit is clear, the MAC ignores the PTP transported over UDP-IPv6 packets.
TSIPV4EN A	13	rw	Enable Processing of PTP Frames Sent over IPv4-UDP When set, the MAC receiver processes the PTP packets encapsulated in UDP over IPv4 packets. When this bit is clear, the MAC ignores the PTP transported over UDP-IPv4 packets. This bit is set by default.
TSEVNT NA	14	rw	Enable Timestamp Snapshot for Event Messages When set, the timestamp snapshot is taken only for event messages (SYNC, Delay_Req, Pdelay_Req, or Pdelay_Resp). When reset, the snapshot is taken for all messages except Announce, Management, and Signaling.
TSMSTRE NA	15	rw	Enable Snapshot for Messages Relevant to Master When set, the snapshot is taken only for the messages relevant to the master node. Otherwise, the snapshot is taken for the messages relevant to the slave node.

Ethernet MAC (ETH)

Field	Bits	Type	Description
SNAPTYP SEL	[17:16]	rw	Select PTP packets for Taking Snapshots These bits along with Bits 15 and 14 decide the set of PTP packet types for which snapshot needs to be taken.
TSENMAC ADDR	18	rw	Enable MAC address for PTP Frame Filtering When set, the DA MAC address (that matches any MAC Address register) is used to filter the PTP frames when PTP is directly sent over Ethernet.
RESERVE D_23_19	[23:19]	r	RESERVED_23_19
ATSFC	24	r	Auxiliary Snapshot FIFO Clear When set, it resets the pointers of the Auxiliary Snapshot FIFO. This bit is cleared when the pointers are reset and the FIFO is empty. When this bit is high, auxiliary snapshots get stored in the FIFO. This bit is reserved when the Add IEEE 1588 Auxiliary Snapshot option is not selected during core configuration.
ATSEN0	25	r	Auxiliary Snapshot 0 Enable This field controls capturing the Auxiliary Snapshot Trigger 0. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[0] input is enabled. When this bit is reset, the events on this input are ignored. This bit is reserved when the IEEE 1588 Auxiliary Snapshot option is not selected during core configuration.
ATSEN1	26	r	Auxiliary Snapshot 1 Enable This field controls capturing the Auxiliary Snapshot Trigger 1. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[1] input is enabled. When this bit is reset, the events on this input are ignored. This bit is reserved when the IEEE 1588 Auxiliary Snapshot option is not selected during core configuration or the selected number in the Number of IEEE 1588 Auxiliary Snapshot Inputs option is less than two.

Ethernet MAC (ETH)

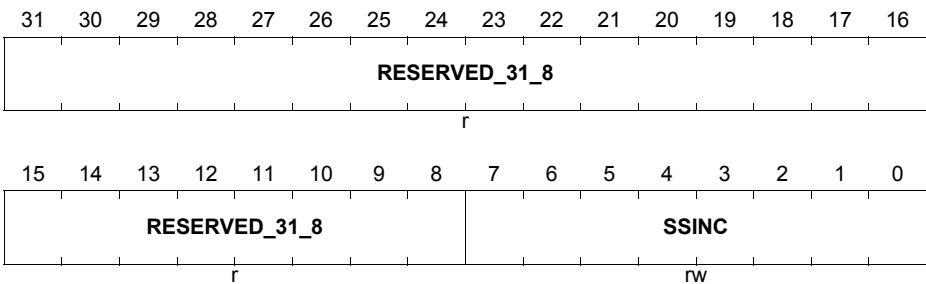
Field	Bits	Type	Description
ATSEN2	27	r	<p>Auxiliary Snapshot 2 Enable</p> <p>This field controls capturing the Auxiliary Snapshot Trigger 2. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[2] input is enabled. When this bit is reset, the events on this input are ignored. This bit is reserved when the IEEE 1588 Auxiliary Snapshot option is not selected during core configuration or the selected number in the Number of IEEE 1588 Auxiliary Snapshot Inputs option is less than three.</p>
ATSEN3	28	r	<p>Auxiliary Snapshot 3 Enable</p> <p>This field controls capturing the Auxiliary Snapshot Trigger 3. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[3] input is enabled. When this bit is reset, the events on this input are ignored. This bit is reserved when the IEEE 1588 Auxiliary Snapshot option is not selected during core configuration or the selected number in the Number of IEEE 1588 Auxiliary Snapshot Inputs option is less than four.</p>
RESERVE D_31_29	[31:29]	r	RESERVED_31_29

32-bit Register - Sub_Second_Increment

This register is present only when the IEEE 1588 timestamp feature is selected without an external timestamp input. In the Coarse Update mode (TSCFUPDT bit in Register 448), the value in this register is added to the system time every clock cycle of `clk_ptp_ref_i`. In the Fine Update mode, the value in this register is added to the system time whenever the Accumulator gets an overflow.

ETH_SUB_SECOND_INCREMENT

Register 449 - Sub-Second Increment Register (1704_H) Reset Value: 0000 0000_H



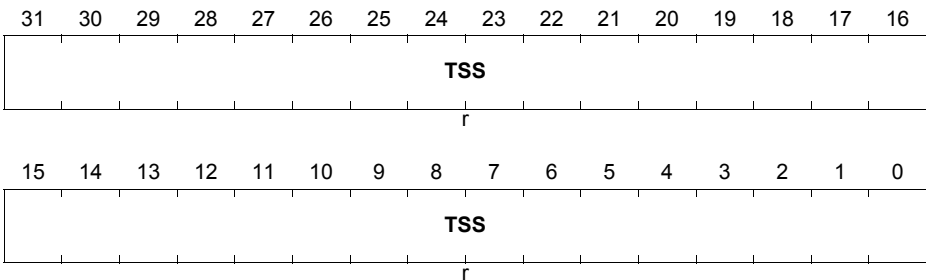
Field	Bits	Type	Description
SSINC	[7:0]	rw	Sub-second Increment Value The value programmed in this field is accumulated every clock cycle (of <code>clk_ptp_i</code>) with the contents of the sub-second register. For example, when PTP clock is 50 MHz (period is 20 ns), you should program 20 (0x14) when the System Time-Nanoseconds register has an accuracy of 1 ns (TSCTRLSSR bit is set). When TSCTRLSSR is clear, the Nanoseconds register has a resolution of ~0.465ns. In this case, you should program a value of 43 (0x2B) that is derived by 20ns/0.465.
RESERVE D_31_8	[31:8]	r	RESERVED_31_8

32-bit Register - System_Time_Seconds

The System Time -Seconds register, along with System-TimeNanoseconds register, indicates the current value of the system time maintained by the MAC. Though it is updated on a continuous basis, there is some delay from the actual time because of clock domain transfer latencies (from clk_ptp_ref_i to clk_csr_i). These registers (450 and 451) are present only when the IEEE 1588 Timestamp feature is selected without external timestamp input.

ETH_SYSTEM_TIME_SECONDS

Register 450 - System Time - Seconds Register (1708_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
TSS	[31:0]	r	Timestamp Second The value in this field indicates the current value in seconds of the System Time maintained by the MAC.

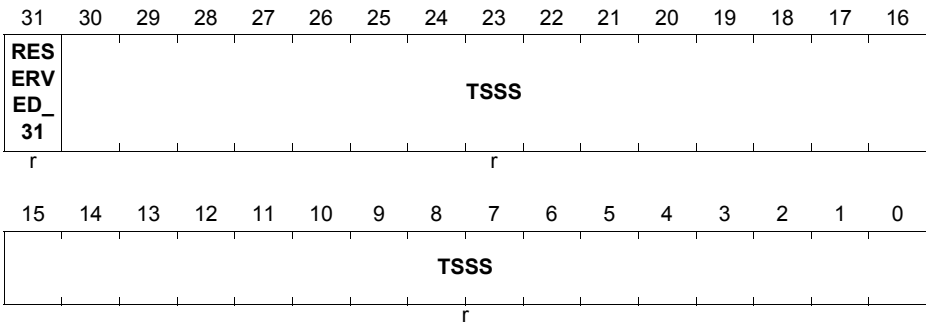
Ethernet MAC (ETH)

32-bit Register - System_Time_Nanoseconds

The value in this field has the sub second representation of time, with an accuracy of 0.46 ns. When TSCTRLSSR is set, each bit represents 1 ns and the maximum value is 0x3B9A_C9FF, after which it rolls-over to zero.

ETH_SYSTEM_TIME_NANOSECONDS

Register 451 - System Time - Nanoseconds Register (170C_H) Reset Value: 0000 0000_H



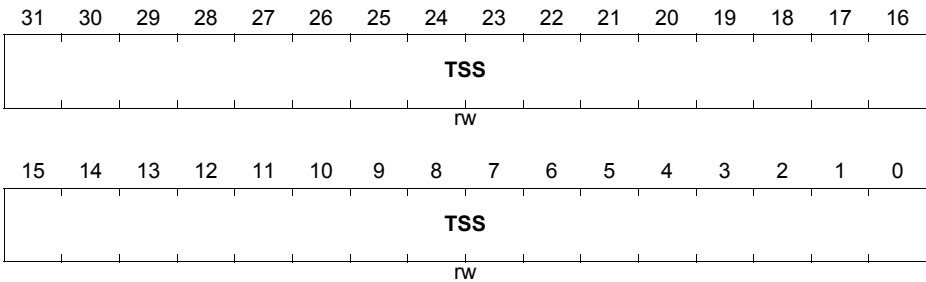
Field	Bits	Type	Description
TSSS	[30:0]	r	Timestamp Sub Seconds The value in this field has the sub second representation of time, with an accuracy of 0.46 ns. When bit 9 (TSCTRLSSR) is set in Register 448 (Timestamp Control Register), each bit represents 1 ns and the maximum value is 0x3B9A_C9FF, after which it rolls-over to zero.
RESERVED_31	31	r	RESERVED_31

32-bit Register - System_Time_Seconds_Update

The System Time - Seconds Update register, along with the System Time - Nanoseconds Update register, initializes or updates the system time maintained by the MAC. You must write both of these registers before setting the TSINIT or TSUPDT bits in the Timestamp Control register. This register is present only when the IEEE 1588 Timestamp feature is selected without external timestamp input.

ETH_SYSTEM_TIME_SECONDS_UPDATE

Register 452 - System Time - Seconds Update Register (1710_H) Reset Value: 0000 0000_H



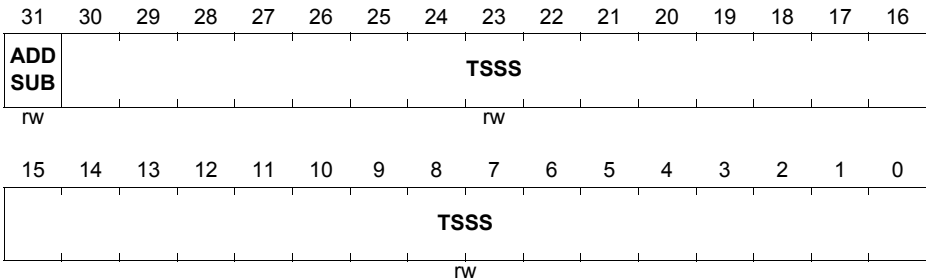
Field	Bits	Type	Description
TSS	[31:0]	rw	Timestamp Second The value in this field indicates the time in seconds to be initialized or added to the system time.

32-bit Register - System_Time_Nanoseconds_Update

This register is present only when IEEE 1588 timestamp feature is selected without external timestamp input.

ETH_SYSTEM_TIME_NANOSECONDS_UPDATE

Register 453 - System Time - Nanoseconds Update Register (1714_H) Reset Value: 0000 0000_H



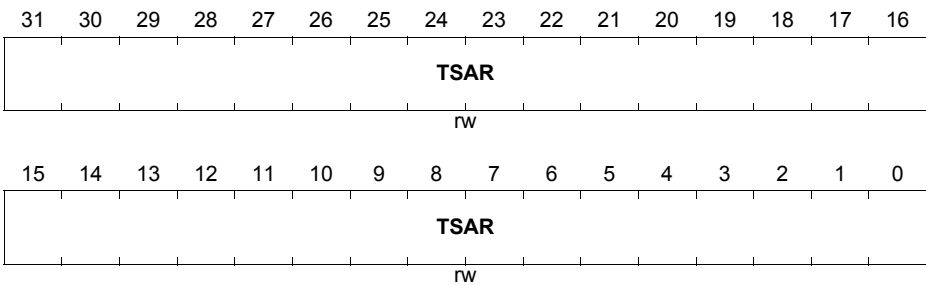
Field	Bits	Type	Description
TSSS	[30:0]	rw	Timestamp Sub Second The value in this field has the sub second representation of time, with an accuracy of 0.46 ns. When bit 9 (TCTRLSSR) is set in Register 448 (Timestamp Control Register), each bit represents 1 ns and the programmed value should not exceed 0x3B9A_C9FF.
ADDSUB	31	rw	Add or subtract time When this bit is set, the time value is subtracted with the contents of the update register. When this bit is reset, the time value is added with the contents of the update register.

32-bit Register - Timestamp_Addend

This register is present only when the IEEE 1588 Timestamp feature is selected without external timestamp input. This register value is used only when the system time is configured for Fine Update mode (TSCFUPDT bit in Register 448). This register content is added to a 32-bit accumulator in every clock cycle (of clk_ptp_ref_i) and the system time is updated whenever the accumulator overflows.

ETH_TIMESTAMP_ADDEND

Register 454 - Timestamp Addend Register (1718_H) Reset Value: 0000 0000_H



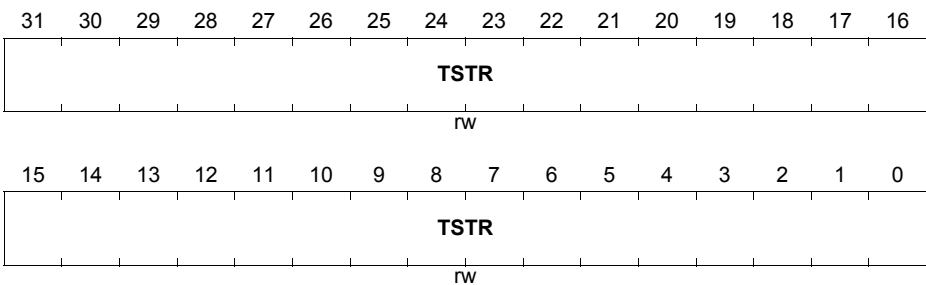
Field	Bits	Type	Description
TSAR	[31:0]	rw	Timestamp Addend Register This field indicates the 32-bit time value to be added to the Accumulator register to achieve time synchronization.

32-bit Register - Target_Time_Seconds

The Target Time Seconds register, along with Target Time Nanoseconds register, is used to schedule an interrupt event (Register 458[1] when Advanced Timestamping is enabled; otherwise, TS interrupt bit in Register14[9]) when the system time exceeds the value programmed in these registers. This register is present only when the IEEE 1588 Timestamp feature is selected without external timestamp input.

ETH_TARGET_TIME_SECONDS

Register 455 - Target Time Seconds Register (171C_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
TSTR	[31:0]	rw	Target Time Seconds Register This register stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, then based on Bits [6:5] of Register 459 (PPS Control Register), the MAC starts or stops the PPS signal output and generates an interrupt (if enabled).

32-bit Register - Target_Time_Nanoseconds

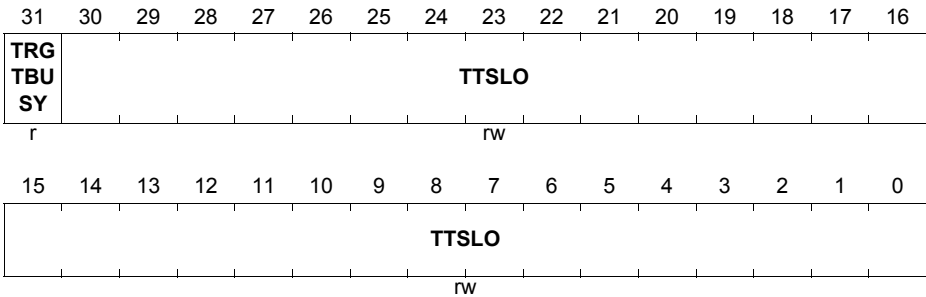
This register is present only when the IEEE 1588 Timestamp feature is selected without external timestamp input.

ETH_TARGET_TIME_NANOSECONDS

Register 456 - Target Time Nanoseconds Register (1720_H)

Reset Value: 0000

0000_H



Field	Bits	Type	Description
TTSLO	[30:0]	rw	<p>Target Timestamp Low Register</p> <p>This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the both Target Timestamp registers, then based on the TRGTMODSEL0 field (Bits [6:5]) in Register 459 (PPS Control Register), the MAC starts or stops the PPS signal output and generates an interrupt (if enabled). This value should not exceed 0x3B9A_C9FF when TSCTRLSSR is set in the Timestamp control register. The actual start or stop time of the PPS signal output may have an error margin up to one unit of sub-second increment value.</p>

Ethernet MAC (ETH)

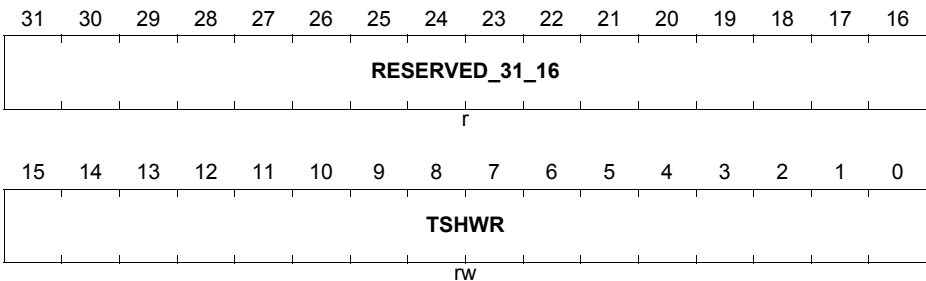
Field	Bits	Type	Description
TRGTBUSY	31	r	<p>Target Time Register Busy</p> <p>The MAC sets this bit when the PPSCMD field (Bits[3:0]) in Register 459 (PPS Control Register) is programmed to 010 or 011. Programming the PPSCMD field to 010 or 011, instructs the MAC to synchronize the Target Time Registers to the PTP clock domain.</p> <p>The MAC clears this bit after synchronizing the Target Time Registers to the PTP clock domain. The application must not update the Target Time Registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted. This bit is reserved when the Enable Flexible Pulse-Per-Second Output feature is not selected.</p>

32-bit Register - System_Time_Higher_Word_Seconds

This register is present only when the IEEE 1588 Advanced Timestamp feature is selected without an external timestamp input.

ETH_SYSTEM_TIME_HIGHER_WORD_SECONDS

Register 457 - System Time - Higher Word Seconds Register (1724_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
TSHWR	[15:0]	rw	Timestamp Higher Word Register This field contains the most significant 16-bits of the timestamp seconds value. This register is optional and can be selected using the Enable IEEE 1588 Higher Word Register option during core configuration. The register is directly written to initialize the value. This register is incremented when there is an overflow from the 32-bits of the System Time - Seconds register.
RESERVE D_31_16	[31:16]	r	RESERVED_31_16

32-bit Register - Timestamp_Status

This register is present only when Advanced IEEE 1588 Timestamp feature is selected. All bits except Bits[27:25] gets cleared when the host reads this register.

ETH_TIMESTAMP_STATUS
Register 458 - Timestamp Status Register (1728_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVE D_31_30		ATSNS					ATS STM	RESERVED_23_20				ATSSTN			
r		r					r	r				r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED_15_10						TST RGT ERR 3	TST ARG T3	TST RGT ERR 2	TST ARG T2	TST RGT ERR 1	TST ARG T1	TST RGT ERR	AUX TST RIG	TST ARG T	TSS OVF
r						r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
TSSOVF	0	r	Timestamp Seconds Overflow When set, this bit indicates that the seconds value of the timestamp (when supporting version 2 format) has overflowed beyond 32'hFFFF_FFFF.
TSTARGET	1	r	Timestamp Target Time Reached When set, this bit indicates that the value of system time is greater or equal to the value specified in the Register 455 (Target Time Seconds Register) and Register 456 (Target Time Nanoseconds Register).
AUXTSTTRIGGER	2	r	Auxiliary Timestamp Trigger Snapshot This bit is set high when the auxiliary snapshot is written to the FIFO. This bit is valid only if the Enable IEEE 1588 Auxiliary Snapshot feature is selected.
TSTRGTERR	3	r	Timestamp Target Time Error This bit is set when the target time, being programmed in Target Time Registers, is already elapsed. This bit is cleared when read by the application.

Ethernet MAC (ETH)

Field	Bits	Type	Description
TSTARGET 1	4	r	Timestamp Target Time Reached for Target Time PPS1 When set, this bit indicates that the value of system time is greater than or equal to the value specified in Register 480 (PPS1 Target Time High Register) and Register 481 (PPS1 Target Time Low Register).
TSTRGTE RR1	5	r	Timestamp Target Time Error This bit is set when the target time, being programmed in Register 480 and Register 481, is already elapsed. This bit is cleared when read by the application.
TSTARGET 2	6	r	Timestamp Target Time Reached for Target Time PPS2 When set, this bit indicates that the value of system time is greater than or equal to the value specified in Register 488 (PPS2 Target Time High Register) and Register 489 (PPS2 Target Time Low Register).
TSTRGTE RR2	7	r	Timestamp Target Time Error This bit is set when the target time, being programmed in Register 488 and Register 489, is already elapsed. This bit is cleared when read by the application.
TSTARGET 3	8	r	Timestamp Target Time Reached for Target Time PPS3 When set, this bit indicates that the value of system time is greater than or equal to the value specified in Register 496 (PPS3 Target Time High Register) and Register 497 (PPS3 Target Time Low Register).
TSTRGTE RR3	9	r	Timestamp Target Time Error This bit is set when the target time, being programmed in Register 496 and Register 497, is already elapsed. This bit is cleared when read by the application.
RESERVE D_15_10	[15:10]	r	RESERVED_15_10

Ethernet MAC (ETH)

Field	Bits	Type	Description
ATSSTN	[19:16]	r	<p>Auxiliary Timestamp Snapshot Trigger Identifier</p> <p>These bits identify the Auxiliary trigger inputs for which the timestamp available in the Auxiliary Snapshot Register is applicable. When more than one bit is set at the same time, it means that corresponding auxiliary triggers were sampled at the same clock. These bits are applicable only if the number of Auxiliary snapshots is more than one. One bit is assigned for each trigger as shown in the following list:</p> <ul style="list-style-type: none"> * Bit 16: Auxiliary trigger 0 * Bit 17: Auxiliary trigger 1 * Bit 18: Auxiliary trigger 2 * Bit 19: Auxiliary trigger 3 <p>The software can read this register to find the triggers that are set when the timestamp is taken.</p>
RESERVE D_23_20	[23:20]	r	RESERVED_23_20
ATSSTM	24	r	<p>Auxiliary Timestamp Snapshot Trigger Missed</p> <p>This bit is set when the Auxiliary timestamp snapshot FIFO is full and external trigger was set. This indicates that the latest snapshot is not stored in the FIFO. This bit is valid only if the Add IEEE 1588 Auxiliary Snapshot option is selected during core configuration.</p>
ATSNS	[29:25]	r	<p>Number of Auxiliary Timestamp Snapshots</p> <p>This field indicates the number of Snapshots available in the FIFO. A value equal to the selected depth of FIFO (4, 8, or 16) indicates that the Auxiliary Snapshot FIFO is full. These bits are cleared (to 00000) when the Auxiliary snapshot FIFO clear bit is set. This bit is valid only if the Add IEEE 1588 Auxiliary Snapshot option is selected during core configuration.</p>
RESERVE D_31_30	[31:30]	r	RESERVED_31_30

32-bit Register - PPS_Control

This register is present only when the Advanced Timestamp feature is selected and External Timestamp is not enabled. Note: * Bits[30:24] are valid only when four Flexible PPS outputs are selected. * Bits[22:16] are valid only when three or more Flexible PPS outputs are selected. * Bits[14:8] are valid only when two or more Flexible PPS outputs are selected. * Bits[6:4] are valid only when Flexible PPS feature is selected.

ETH_PPS_CONTROL

Register 459 - PPS Control Register (172C_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES ERV ED_ 31	TRGTMOD SEL3		RESERVE D_28_27		PPSCMD3			RES ERV ED_ 23	TRGTMOD SEL2		RESERVE D_20_19		PPSCMD2		
r	r		r		r			r	r		r		r		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES ERV ED_ 15	TRGTMOD SEL1		RESERVE D_12_11		PPSCMD1			RES ERV ED_ 7	TRGTMOD SEL0		PPS EN0	PPSCTRL_PPSCMD			
r	r		r		r			r	r		r	rw			

Ethernet MAC (ETH)

Field	Bits	Type	Description
PPSCTRL _PPSCMD	[3:0]	rw	<p>PPSCTRL0 or PPSCMD0</p> <p>PPSCTRL0: PPS0 Output Frequency Control This field controls the frequency of the PPS0 output (ptp_pps_o[0]) signal. The default value of PPSCTRL is 0000, and the PPS output is 1 pulse (of width clk_ptp_i) every second. For other values of PPSCTRL, the PPS output becomes a generated clock of following frequencies:</p> <ul style="list-style-type: none"> -0001: The binary rollover is 2 Hz, and the digital rollover is 1 Hz. -0010: The binary rollover is 4 Hz, and the digital rollover is 2 Hz. -0011: The binary rollover is 8 Hz, and the digital rollover is 4 Hz. -0100: The binary rollover is 16 Hz, and the digital rollover is 8 Hz. -... -1111: The binary rollover is 32.768 KHz, and the digital rollover is 16.384 KHz. <p>Note: In the binary rollover mode, the PPS output (ptp_pps_o) has a duty cycle of 50 percent with these frequencies. In the digital rollover mode, the PPS output frequency is an average number. The actual clock is of different frequency that gets synchronized every second. For example:</p> <ul style="list-style-type: none"> * When PPSCTRL = 0001, the PPS (1 Hz) has a low period of 537 ms and a high period of 463 ms * When PPSCTRL = 0010, the PPS (2 Hz) is a sequence of: <ul style="list-style-type: none"> - One clock of 50 percent duty cycle and 537 ms period - Second clock of 463 ms period (268 ms low and 195 ms high) * When PPSCTRL = 0011, the PPS (4 Hz) is a sequence of: <ul style="list-style-type: none"> - Three clocks of 50 percent duty cycle and 268 ms period - Fourth clock of 195 ms period (134 ms low and 61 ms high) <p>This behavior is because of the non-linear toggling of bits in the digital rollover mode in Register 451 (System Time - Nanoseconds Register).</p> <p>Flexible PPS0 Output (ptp_pps_o[0]) Control Programming these bits with a non-zero value instructs the MAC to initiate an event. Once the command is</p>

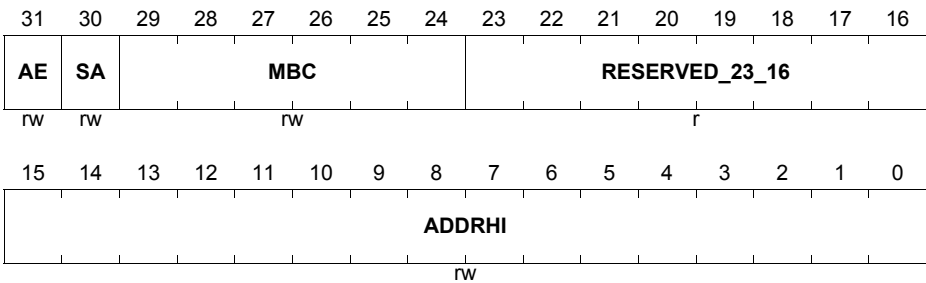
Field	Bits	Type	Description
PPSEN0	4	r	Flexible PPS Output Mode Enable When set low, Bits[3:0] function as PPSCTRL (backward compatible). When set high, Bits[3:0] function as PPSCMD.
TRGTMOD SEL0	[6:5]	r	Target Time Register Mode for PPS0 Output This field indicates the Target Time registers (register 455 and 456) mode for PPS0 output signal: * 00: Indicates that the Target Time registers are programmed only for generating the interrupt event. * 01: Reserved * 10: Indicates that the Target Time registers are programmed for generating the interrupt event and starting or stopping the generation of the PPS0 output signal. * 11: Indicates that the Target Time registers are programmed only for starting or stopping the generation of the PPS0 output signal. No interrupt is asserted.
RESERVE D_7	7	r	RESERVED_7
PPSCMD1	[10:8]	r	Flexible PPS1 Output Control This field controls the flexible PPS1 output (ptp_pps_o[1]) signal. This field is similar to PPSCMD0[2:0] in functionality.
RESERVE D_12_11	[12:11]	r	RESERVED_12_11
TRGTMOD SEL1	[14:13]	r	Target Time Register Mode for PPS1 Output This field indicates the Target Time registers (register 480 and 481) mode for PPS1 output signal. This field is similar to the TRGTMODSEL0 field.
RESERVE D_15	15	r	RESERVED_15
PPSCMD2	[18:16]	r	Flexible PPS2 Output Control This field controls the flexible PPS2 output (ptp_pps_o[2]) signal. This field is similar to PPSCMD0[2:0] in functionality.
RESERVE D_20_19	[20:19]	r	RESERVED_20_19

Ethernet MAC (ETH)

Field	Bits	Type	Description
TRGTMOD SEL2	[22:21]	r	Target Time Register Mode for PPS2 Output This field indicates the Target Time registers (register 488 and 489) mode for PPS2 output signal. This field is similar to the TRGTMODSEL0 field.
RESERVE D_23	23	r	RESERVED_23
PPSCMD3	[26:24]	r	Flexible PPS3 Output Control This field controls the flexible PPS3 output (ptp_pps_o[3]) signal. This field is similar to PPSCMD0[2:0] in functionality.
RESERVE D_28_27	[28:27]	r	RESERVED_28_27
TRGTMOD SEL3	[30:29]	r	Target Time Register Mode for PPS3 Output This field indicates the Target Time registers (register 496 and 497) mode for PPS3 output signal. This field is similar to the TRGTMODSEL0 field.
RESERVE D_31	31	r	RESERVED_31

32-bit Register - MAC_Address16_High

The MAC Address16 High register holds the upper 16 bits of the 17th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address16 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address16 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS16_HIGH
Register 512 - MAC Address16 High Register (1800_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address16 [47:32] This field contains the upper 16 bits (47:32) of the 17th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address16 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

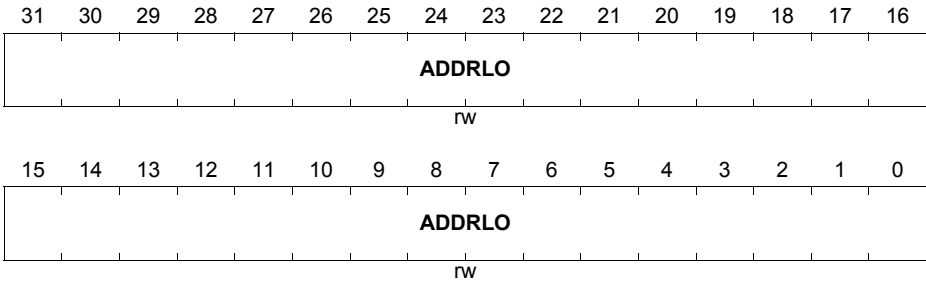
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address16[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address16[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 17th MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address16_Low

The MAC Address16 Low register holds the lower 32 bits of the 17th 6-byte MAC address of the station.

ETH_MAC_ADDRESS16_LOW

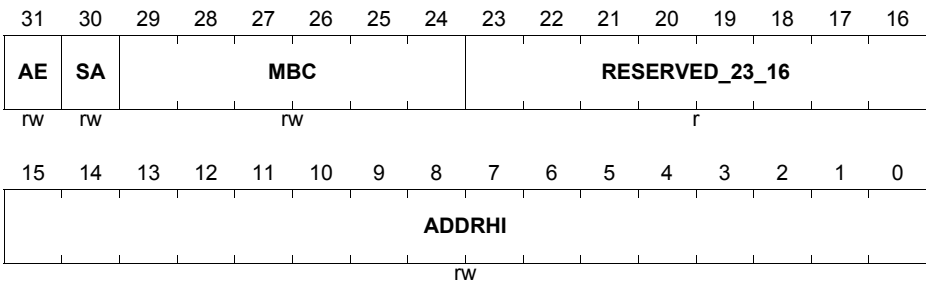
Register 513 - MAC Address16 Low Register (1804_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address16 [31:0] This field contains the lower 32 bits of the 17th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address17_High

The MAC Address17 High register holds the upper 16 bits of the 18th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address17 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address17 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS17_HIGH
Register 514 - MAC Address17 High Register (1808_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address18 [47:32] This field contains the upper 16 bits (47:32) of the 19th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address17 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

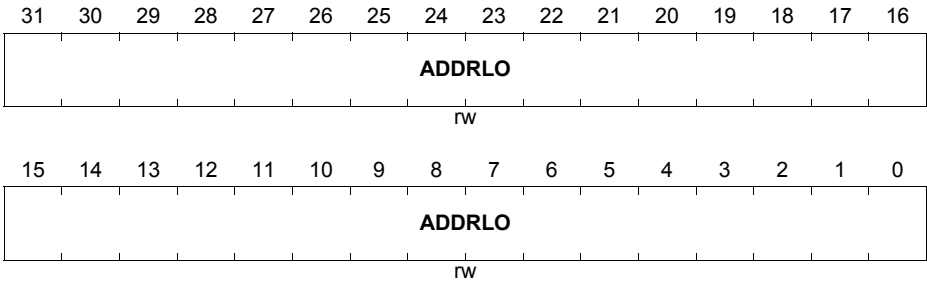
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address17[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address17[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 18th MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address17_Low

The MAC Address17 Low register holds the lower 32 bits of the 18th 6-byte MAC address of the station.

ETH_MAC_ADDRESS17_LOW

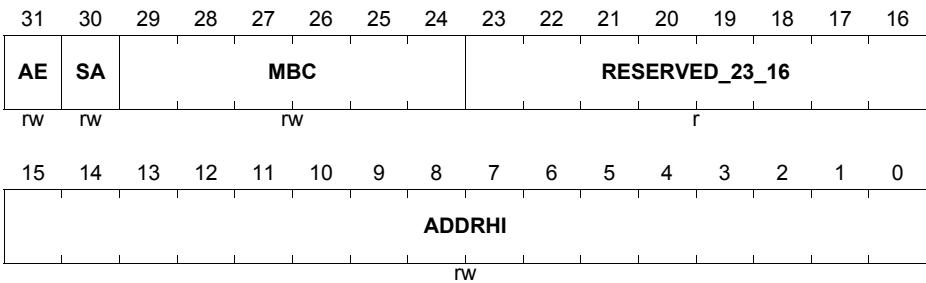
Register 515 - MAC Address17 Low Register (180C_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address17 [31:0] This field contains the lower 32 bits of the 18th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address18_High

The MAC Address18 High register holds the upper 16 bits of the 19th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address18 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address18 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS18_HIGH
Register 516 - MAC Address18 High Register (1810_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address18 [47:32] This field contains the upper 16 bits (47:32) of the 19th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address18 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

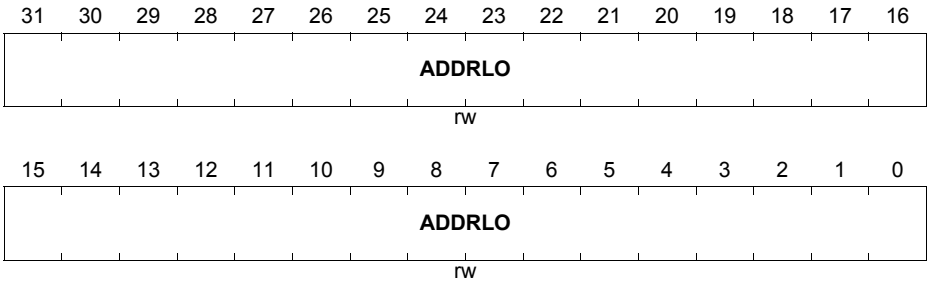
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address18[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address18[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 19th MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address18_Low

The MAC Address18 Low register holds the lower 32 bits of the 19th 6-byte MAC address of the station.

ETH_MAC_ADDRESS18_LOW

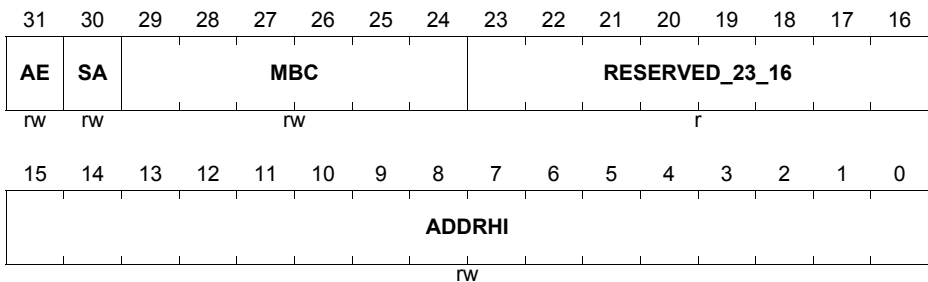
Register 517 - MAC Address18 Low Register (1814_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address18 [31:0] This field contains the lower 32 bits of the 19th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address19_High

The MAC Address19 High register holds the upper 16 bits of the 20th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address19 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address19 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS19_HIGH
Register 518 - MAC Address19 High Register (1818_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address19 [47:32] This field contains the upper 16 bits (47:32) of the 20th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address19 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

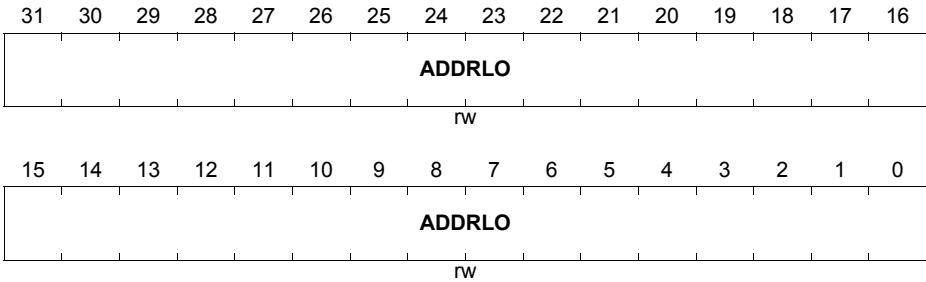
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address19[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address19[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 20th MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address19_Low

The MAC Address19 Low register holds the lower 32 bits of the 20th 6-byte MAC address of the station.

ETH_MAC_ADDRESS19_LOW

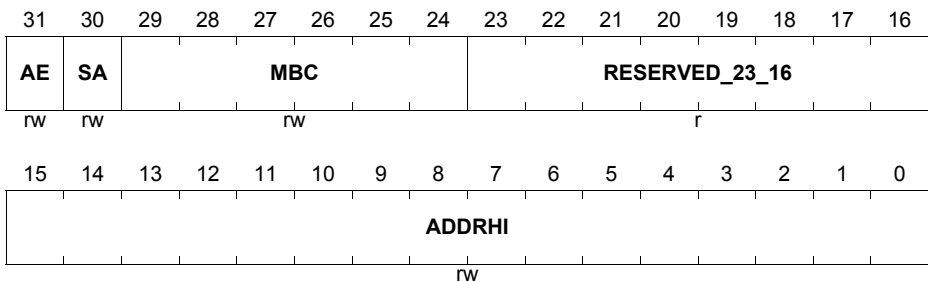
Register 519 - MAC Address19 Low Register (181C_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address19 [31:0] This field contains the lower 32 bits of the 20th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address20_High

The MAC Address20 High register holds the upper 16 bits of the 21st 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address20 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address20 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS20_HIGH
Register 520 - MAC Address20 High Register (1820_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address20 [47:32] This field contains the upper 16 bits (47:32) of the 20th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address20 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

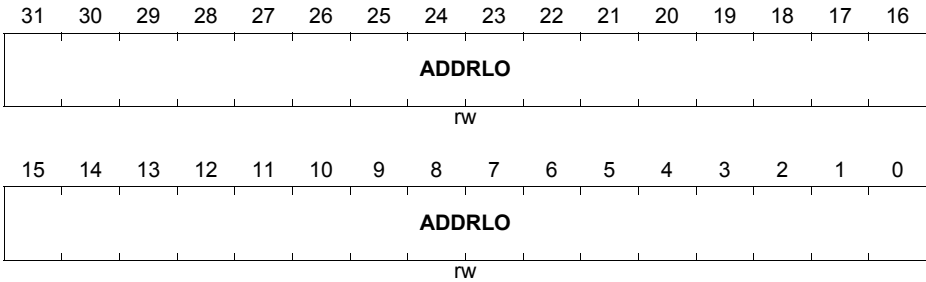
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address20[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address20[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 21st MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address20_Low

The MAC Address20 Low register holds the lower 32 bits of the 21st 6-byte MAC address of the station.

ETH_MAC_ADDRESS20_LOW

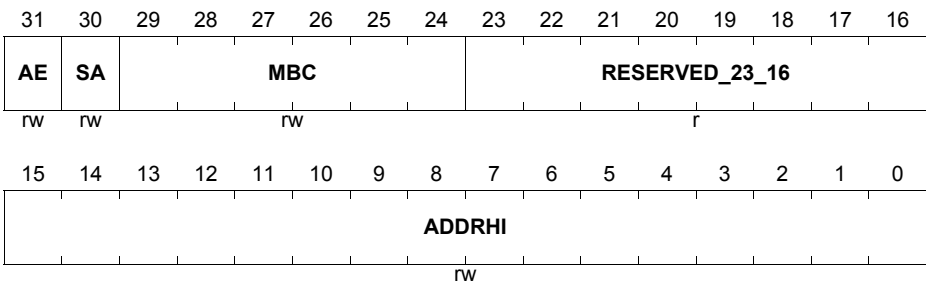
Register 521 - MAC Address20 Low Register (1824_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address20 [31:0] This field contains the lower 32 bits of the 21st 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address21_High

The MAC Address21 High register holds the upper 16 bits of the 22nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address21 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address21 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS21_HIGH
Register 522 - MAC Address21 High Register (1828_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address21 [47:32] This field contains the upper 16 bits (47:32) of the 6-byte 22nd MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address21 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

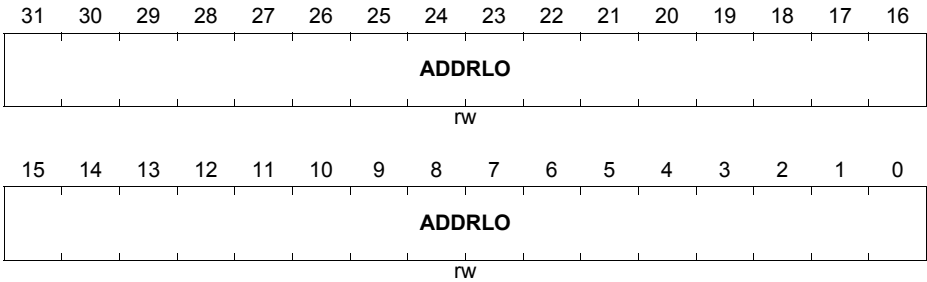
Field	Bits	Type	Description
SA	30	rw	Source Address When this bit is set, the MAC Address21[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address21[47:0] is used to compare with the DA fields of the received frame.
AE	31	rw	Address Enable When this bit is set, the address filter module uses the 22nd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.

32-bit Register - MAC_Address21_Low

The MAC Address21 Low register holds the lower 32 bits of the 22nd 6-byte MAC address of the station.

ETH_MAC_ADDRESS21_LOW

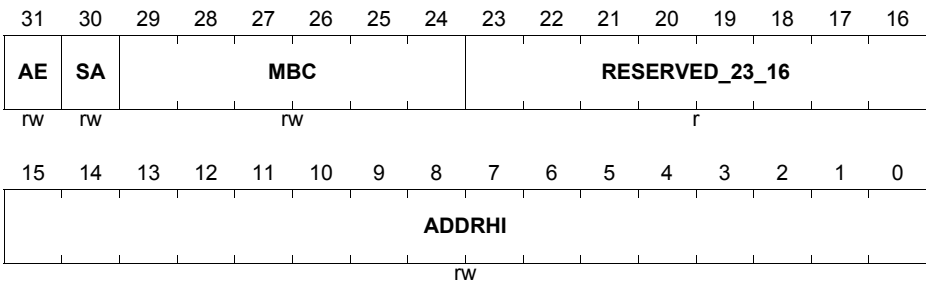
Register 523 - MAC Address21 Low Register (182C_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address21 [31:0] This field contains the lower 32 bits of the 22nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address22_High

The MAC Address22 High register holds the upper 16 bits of the 23rd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address22 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address22 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS22_HIGH
Register 524 - MAC Address22 High Register (1830_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address22 [47:32] This field contains the upper 16 bits (47:32) of the 23rd 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address22 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

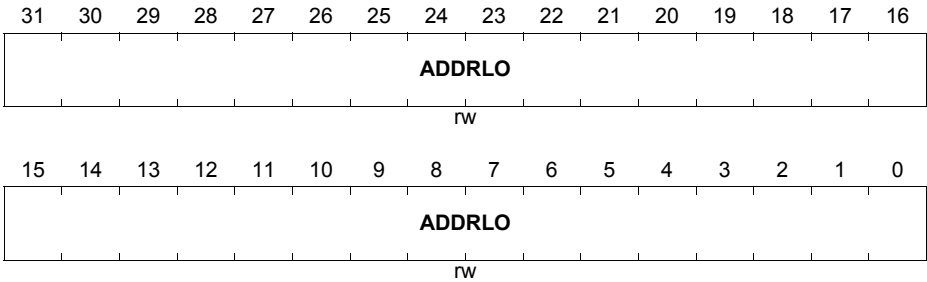
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address22[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address22[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 23rd MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address22_Low

The MAC Address22 Low register holds the lower 32 bits of the 23rd 6-byte MAC address of the station.

ETH_MAC_ADDRESS22_LOW

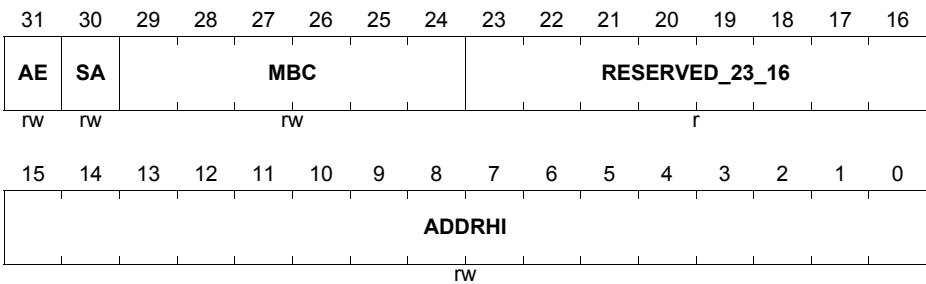
Register 525 - MAC Address22 Low Register (1834_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address22 [31:0] This field contains the lower 32 bits of the 23rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address23_High

The MAC Address23 High register holds the upper 16 bits of the 24th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address23 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address23 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS23_HIGH
Register 526 -
(1838_H)
Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address23 [47:32] This field contains the upper 16 bits (47:32) of the 24th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address23 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

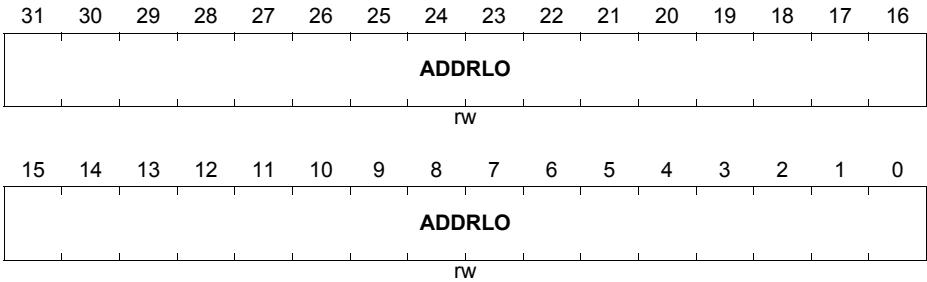
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address23[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address23[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 24th MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address23_Low

The MAC Address23 Low register holds the lower 32 bits of the 24th 6-byte MAC address of the station.

ETH_MAC_ADDRESS23_LOW

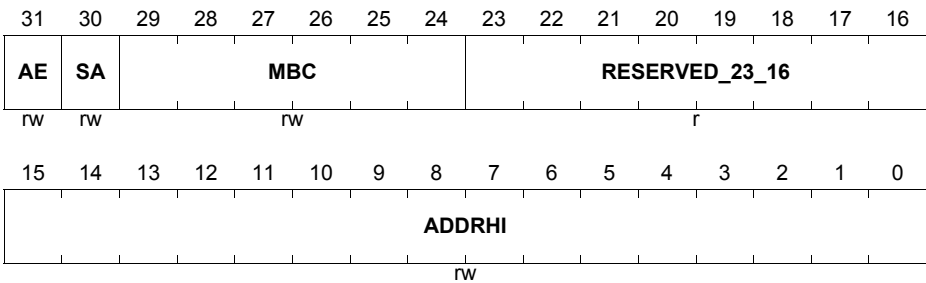
Register 527 - MAC Address23 Low Register (183C_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address23 [31:0] This field contains the lower 32 bits of the 24th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address24_High

The MAC Address24 High register holds the upper 16 bits of the 25th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address24 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address24 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS24_HIGH
Register 528 - MAC Address24 High Register (1840_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address1 [47:32] This field contains the upper 16 bits (47:32) of the 25th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address24 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

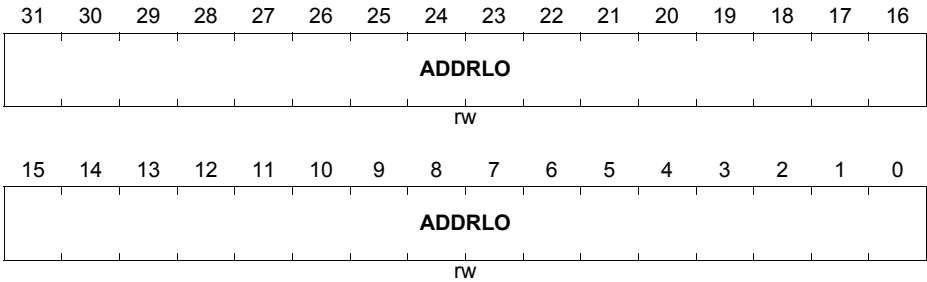
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address₂₄[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address₂₄[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 25th MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address24_Low

The MAC Address24 Low register holds the lower 32 bits of the 25th 6-byte MAC address of the station.

ETH_MAC_ADDRESS24_LOW

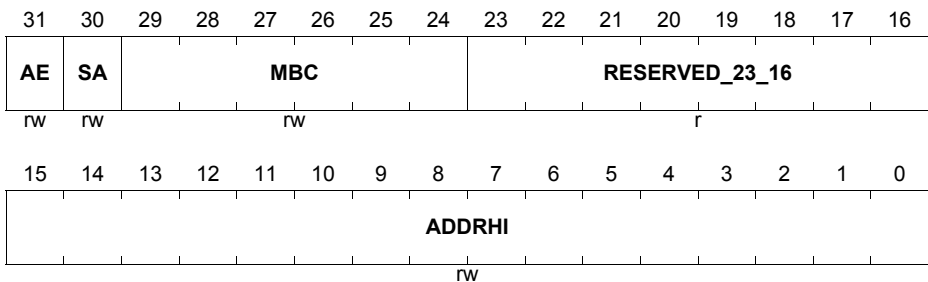
Register 529 - MAC Address24 Low Register (1844_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address24 [31:0] This field contains the lower 32 bits of the 25th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address25_High

The MAC Address25 High register holds the upper 16 bits of the 6-byte 26th MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address25 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address25 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS25_HIGH
Register 530 - MAC Address25 High Register (1848_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address25 [47:32] This field contains the upper 16 bits (47:32) of the 26th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address25 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

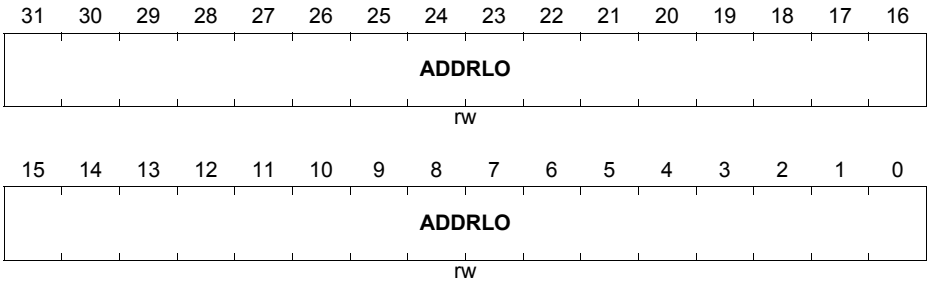
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address25[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address25[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 26th MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address25_Low

The MAC Address25 Low register holds the lower 32 bits of the 26th 6-byte MAC address of the station.

ETH_MAC_ADDRESS25_LOW

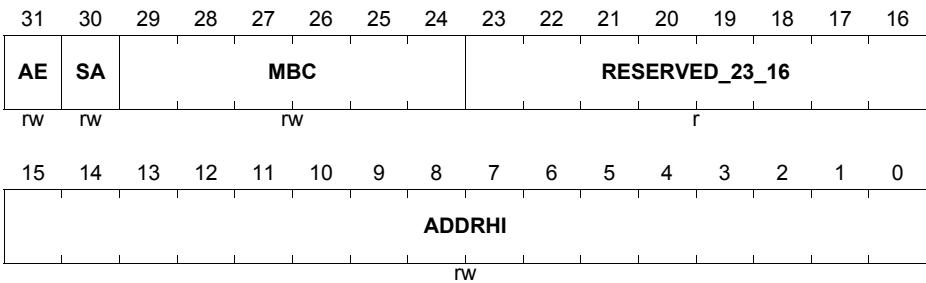
Register 531 - MAC Address25 Low Register (184C_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address25 [31:0] This field contains the lower 32 bits of the 26th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address26_High

The MAC Address26 High register holds the upper 16 bits of the 27th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address26 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address26 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS26_HIGH
Register 532 - MAC Address26 High Register (1850_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address26 [47:32] This field contains the upper 16 bits (47:32) of the 27th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address26 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

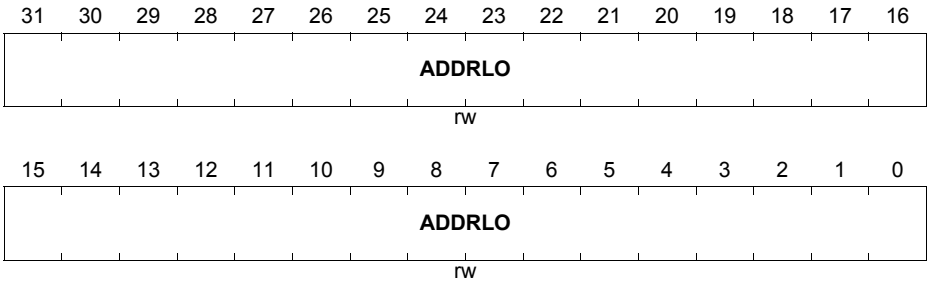
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address26[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address26[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 27th MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address26_Low

The MAC Address26 Low register holds the lower 32 bits of the 27th 6-byte MAC address of the station.

ETH_MAC_ADDRESS26_LOW

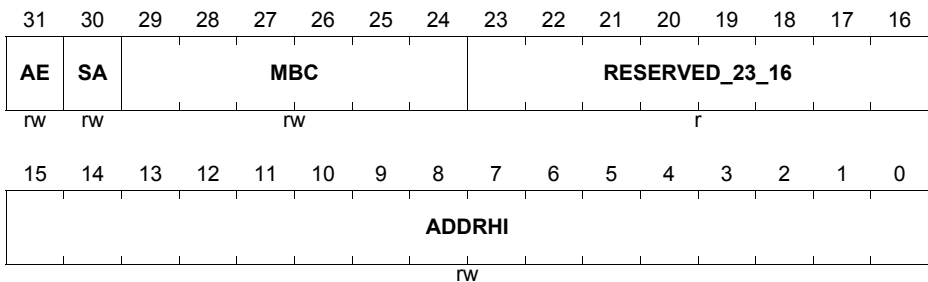
Register 533 - MAC Address26 Low Register (1854_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address26 [31:0] This field contains the lower 32 bits of the 27th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address27_High

The MAC Address27 High register holds the upper 16 bits of the 28th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address27 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address27 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS27_HIGH
Register 534 - MAC Address27 High Register (1858_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address27 [47:32] This field contains the upper 16 bits (47:32) of the 28th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address27 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

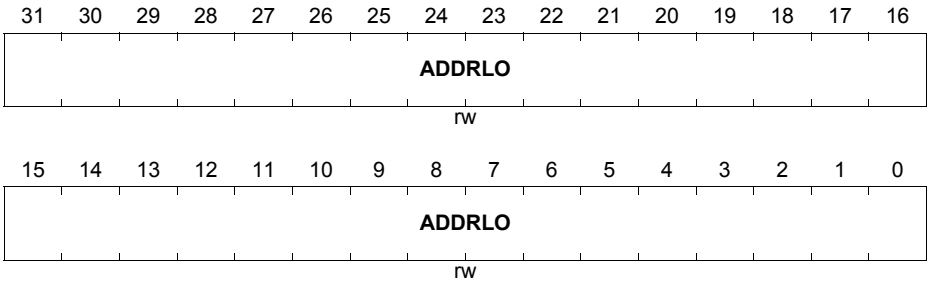
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address₂₇[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address₂₇[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 28th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address27_Low

The MAC Address27 Low register holds the lower 32 bits of the 28th 6-byte MAC address of the station.

ETH_MAC_ADDRESS27_LOW

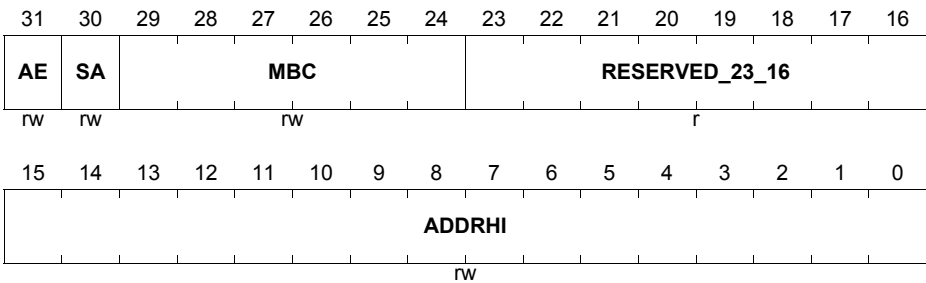
Register 535 - MAC Address27 Low Register (185C_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address27 [31:0] This field contains the lower 32 bits of the 28th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address28_High

The MAC Address28 High register holds the upper 16 bits of the 29th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address28 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address28 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS28_HIGH
Register 536 - MAC Address28 High Register (1860_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address28 [47:32] This field contains the upper 16 bits (47:32) of the 29th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address28 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

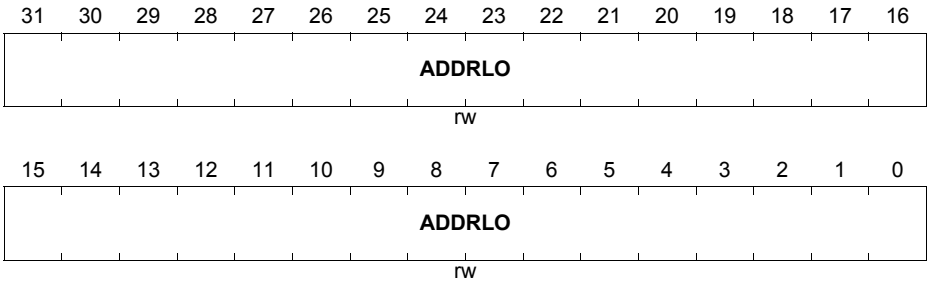
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address28[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address28[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 29th MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address28_Low

The MAC Address28 Low register holds the lower 32 bits of the 29th 6-byte MAC address of the station.

ETH_MAC_ADDRESS28_LOW

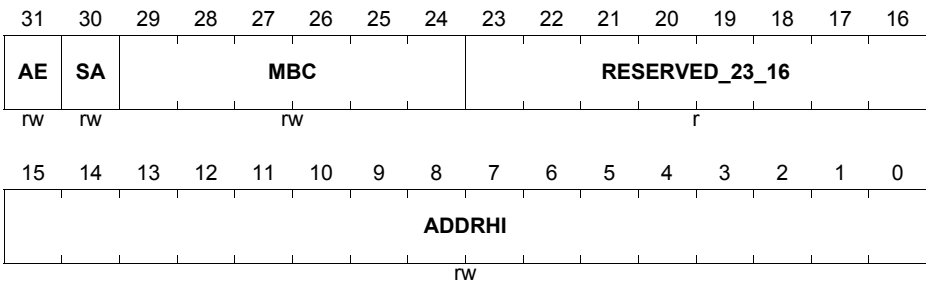
Register 537 - MAC Address28 Low Register (1864_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address28 [31:0] This field contains the lower 32 bits of the 29th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address29_High

The MAC Address29 High register holds the upper 16 bits of the 6-byte 30th MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address29 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address29 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS29_HIGH
Register 538 - MAC Address29 High Register (1868_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address29 [47:32] This field contains the upper 16 bits (47:32) of the 30th 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address29 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

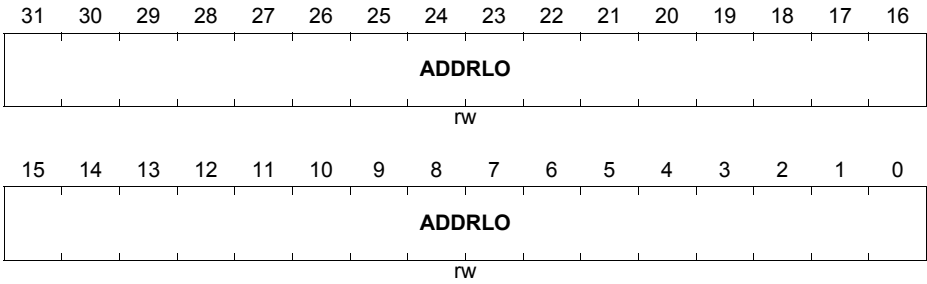
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address29[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address29[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 30th MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address29_Low

The MAC Address29 Low register holds the lower 32 bits of the 30th 6-byte MAC address of the station.

ETH_MAC_ADDRESS29_LOW

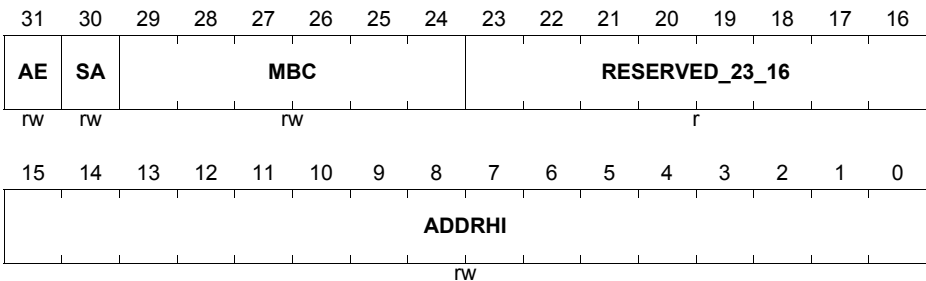
Register 539 - MAC Address29 Low Register (186C_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address29 [31:0] This field contains the lower 32 bits of the 30th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address30_High

The MAC Address30 High register holds the upper 16 bits of the 31st 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address30 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address30 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS30_HIGH
Register 540 - MAC Address30 High Register (1870_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address30 [47:32] This field contains the upper 16 bits (47:32) of the 31st 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address30 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

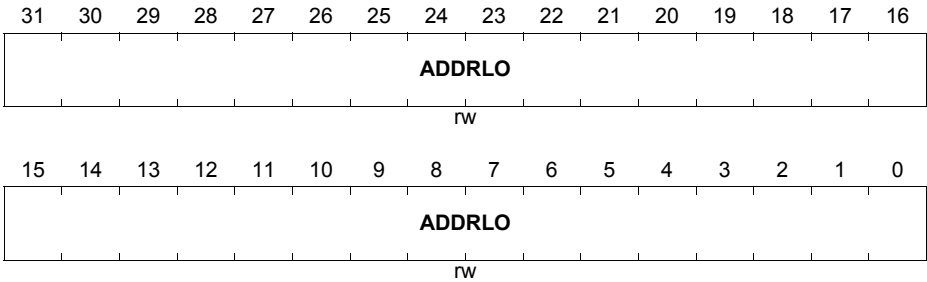
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address30[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address30[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 31st MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address30_Low

The MAC Address30 Low register holds the lower 32 bits of the 31st 6-byte MAC address of the station.

ETH_MAC_ADDRESS30_LOW

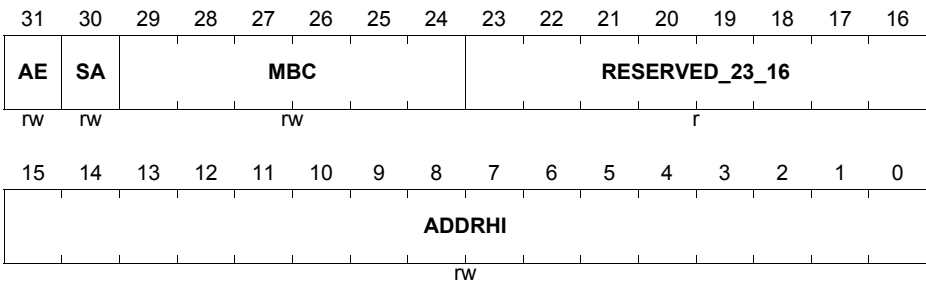
Register 541 - MAC Address30 Low Register (1874_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address30 [31:0] This field contains the lower 32 bits of the 31st 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - MAC_Address31_High

The MAC Address31 High register holds the upper 16 bits of the 32nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address31 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address31 Low Register must be performed after at least four clock cycles in the destination clock domain.

ETH_MAC_ADDRESS31_HIGH
Register 542 - MAC Address31 High Register (1878_H) Reset Value: 0000 FFFF_H


Field	Bits	Type	Description
ADDRHI	[15:0]	rw	MAC Address31 [47:32] This field contains the upper 16 bits (47:32) of the 32nd 6-byte MAC address.
RESERVE D_23_16	[23:16]	r	RESERVED_23_16
MBC	[29:24]	rw	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address31 registers. Each bit controls the masking of the bytes as follows: * Bit 29: Register 18[15:8] * Bit 28: Register 18[7:0] * Bit 27: Register 19[31:24] * ... * Bit 24: Register 19[7:0]

Ethernet MAC (ETH)

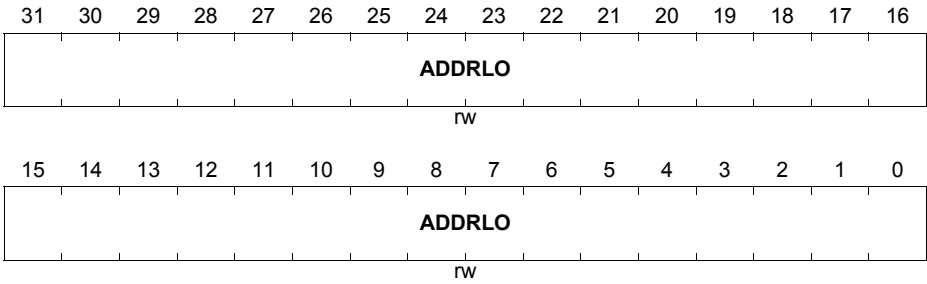
Field	Bits	Type	Description
SA	30	rw	<p>Source Address</p> <p>When this bit is set, the MAC Address31[47:0] is used to compare with the SA fields of the received frame.</p> <p>When this bit is reset, the MAC Address31[47:0] is used to compare with the DA fields of the received frame.</p>
AE	31	rw	<p>Address Enable</p> <p>When this bit is set, the address filter module uses the 32nd MAC address for perfect filtering.</p> <p>When this bit is reset, the address filter module ignores the address for filtering.</p>

32-bit Register - MAC_Address31_Low

The MAC Address31 Low register holds the lower 32 bits of the 32nd 6-byte MAC address of the station.

ETH_MAC_ADDRESS31_LOW

Register 543 - MAC Address31 Low Register (187C_H) Reset Value: FFFF FFFF_H



Field	Bits	Type	Description
ADDRLO	[31:0]	rw	MAC Address31 [31:0] This field contains the lower 32 bits of the 32nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

32-bit Register - Bus_Mode

The Bus Mode register establishes the bus operating modes for the DMA.

ETH_BUS_MODE

Register 0 - Bus Mode Register (2000_H) **Reset Value: 0002 0101_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVE D_31_30		PRWG		TXP R	MB	AAL	PBL x8	USP	RPBL						FB
r		r		rw	rw	rw	rw	rw	rw						rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PR		PBL						ATD S	DSL				DA	SWR	
rw		rw						rw	rw				rw	rw	

Field	Bits	Type	Description
SWR	0	rw	<p>Software Reset</p> <p>When this bit is set, the MAC DMA Controller resets the logic and all internal registers of the MAC. It is cleared automatically after the reset operation has completed in all of the DWC_gmac clock domains. Before reprogramming any register of the DWC_gmac, you should read a zero (0) value in this bit .</p> <p> Note:
</p> <p>* The Software reset function is driven only by this bit. Bit 0 of Register 64 (Channel 1 Bus Mode Register) or Register 128 (Channel 2 Bus Mode Register) has no impact on the Software reset function.</p> <p>* The reset operation is completed only when all resets in all active clock domains are de-asserted. Therefore, it is essential that all the PHY inputs clocks (applicable for the selected PHY interface) are present for the software reset completion.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
DA	1	rw	<p>DMA Arbitration Scheme</p> <p>This bit specifies the arbitration scheme between the transmit and receive paths of Channel 0.</p> <ul style="list-style-type: none"> * 0: Weighted round-robin with Rx:Tx or Tx:Rx. <p>The priority between the paths is according to the priority specified in bits 15:14 (PR) and priority weights specified in Bit 27 (TXPR).</p> <ul style="list-style-type: none"> * 1: Fixed priority. <p>The transmit path has priority over receive path when Bit 27 (TXPR) is set. Otherwise, receive path has priority over the transmit path.</p> <p>In the GMAC-AXI configuration, these bits are reserved and read-only (RO).</p>
DSL	[6:2]	rw	<p>Descriptor Skip Length</p> <p>This bit specifies the number of Word, Dword, or Lword (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of current descriptor to the start of next descriptor. When the DSL value is equal to zero, then the descriptor table is taken as contiguous by the DMA in Ring mode.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
ATDS	7	rw	<p>Alternate Descriptor Size</p> <p>When set, the size of the alternate descriptor increases to 32 bytes (8 DWORDS). This is required when the Advanced Timestamp feature or the IPC Full Offload Engine (Type 2) is enabled in the receiver. The enhanced descriptor is not required if the Advanced Timestamp and IPC Full Checksum Offload (Type 2) features are not enabled. In such cases, you can use the 16 bytes descriptor to save 4 bytes of memory. This bit is present only when you select the Alternate Descriptor feature and any one of the following features during core configuration:</p> <ul style="list-style-type: none"> * Advanced Timestamp feature * IPC Full Checksum Offload Engine (Type 2) feature <p>Otherwise, this bit is reserved and read-only. When reset, the descriptor size reverts back to 4 DWORDs (16 bytes). This bit preserves the backward compatibility for the descriptor size. In versions prior to 3.50a, the descriptor size is 16 bytes for both normal and enhanced descriptor. In version 3.50a, descriptor size is increased to 32 bytes because of the Advanced Timestamp and IPC Full Checksum Offload Engine (Type 2) features.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
PBL	[13:8]	rw	<p>Programmable Burst Length</p> <p>These bits indicate the maximum number of beats to be transferred in one DMA transaction. This is the maximum value that is used in a single block Read or Write. The DMA always attempts to burst as specified in PBL each time it starts a Burst transfer on the host bus. PBL can be programmed with permissible values of 1, 2, 4, 8, 16, and 32. Any other value results in undefined behavior. When USP is set high, this PBL value is applicable only for Tx DMA transactions.</p> <p>If the number of beats to be transferred is more than 32, then perform the following steps:</p> <ol style="list-style-type: none"> 1. Set the PBLx8 mode.
 2. Set the PBL.
 <p>For example, if the maximum number of beats to be transferred is 64, then first set PBLx8 to 1 and then set PBL to 8. The PBL values have the following limitation: The maximum number of possible beats (PBL) is limited by the size of the Tx FIFO and Rx FIFO in the MTL layer and the data bus width on the DMA. The FIFO has a constraint that the maximum beat supported is half the depth of the FIFO, except when specified.</p> <p>For different data bus widths and FIFO sizes, the valid PBL range (including x8 mode) is provided in the following list. If the PBL is common for both transmit and receive DMA, the minimum Rx FIFO and Tx FIFO depths must be considered.</p> <p>Note: In the half-duplex mode, the valid PBL range specified in the following list is applicable only for Tx FIFO.</p> <p>* 32-Bit Data Bus Width</p> <p>- 2 KB and Higher FIFO Depth: All PBL values are supported in the full-duplex mode and half-duplex modes.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
PR	[15:14]	rw	<p>Priority Ratio</p> <p>These bits control the priority ratio in the weighted round-robin arbitration between the Rx DMA and Tx DMA. These bits are valid only when Bit 1 (DA) is reset. The priority ratio is Rx:Tx or Tx:Rx depending on whether Bit 27 (TXPR) is reset or set.</p> <ul style="list-style-type: none"> * 00: The Priority Ratio is 1:1. * 01: The Priority Ratio is 2:1. * 10: The Priority Ratio is 3:1. * 11: The Priority Ratio is 4:1. <p>In the GMAC-AXI configuration, these bits are reserved and read-only (RO).</p>
FB	16	rw	<p>Fixed Burst</p> <p>This bit controls whether the AHB or AXI Master interface performs fixed burst transfers or not. When set, the AHB interface uses only SINGLE, INCR4, INCR8, or INCR16 during start of the normal burst transfers. When reset, the AHB or AXI interface uses SINGLE and INCR burst transfer operations.</p> <p>For more information, see Bit 0 (UNDEF) of the AXI Bus Mode register in the GMAC-AXI configuration.</p>
RPBL	[22:17]	rw	<p>Rx DMA PBL</p> <p>This field indicates the maximum number of beats to be transferred in one Rx DMA transaction. This is the maximum value that is used in a single block Read or Write.</p> <p>The Rx DMA always attempts to burst as specified in the RPBL bit each time it starts a Burst transfer on the host bus. You can program RPBL with values of 1, 2, 4, 8, 16, and 32. Any other value results in undefined behavior. This field is valid and applicable only when USP is set high.</p>
USP	23	rw	<p>Use Separate PBL</p> <p>When set high, this bit configures the Rx DMA to use the value configured in Bits[22:17] as PBL. The PBL value in Bits[13:8] is applicable only to the Tx DMA operations. When reset to low, the PBL value in Bits[13:8] is applicable for both DMA engines.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
PBLx8	24	rw	<p>PBLx8 Mode</p> <p>When set high, this bit multiplies the programmed PBL value (Bits[22:17] and Bits[13:8]) eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value.</p> <p>Note: This bit function is not backward compatible. Before release 3.50a, this bit was 4xPBL.</p>
AAL	25	rw	<p>Address Aligned Beats</p> <p>When this bit is set high and the FB bit is equal to 1, the AHB or AXI interface generates all bursts aligned to the start address LS bits. If the FB bit is equal to 0, the first burst (accessing the data buffer's start address) is not aligned, but subsequent bursts are aligned to the address.</p> <p>This bit is valid only in the GMAC-AHB and GMAC-AXI configuration and is reserved (RO with default value 0) in all other configurations.</p>
MB	26	rw	<p>Mixed Burst</p> <p>When this bit is set high and the FB bit is low, the AHB Master interface starts all bursts of length more than 16 with INCR (undefined burst) whereas it reverts to fixed burst transfers (INCRx and SINGLE) for burst length of 16 and less.</p> <p>This bit is valid only in the GMAC-AHB configuration and reserved in all other configuration.</p>
TXPR	27	rw	<p>Transmit Priority</p> <p>When set, this bit indicates that the transmit DMA has higher priority than the receive DMA during arbitration for the system-side bus. In the GMAC-AXI configuration, this bit is reserved and read-only (RO).</p>

Ethernet MAC (ETH)

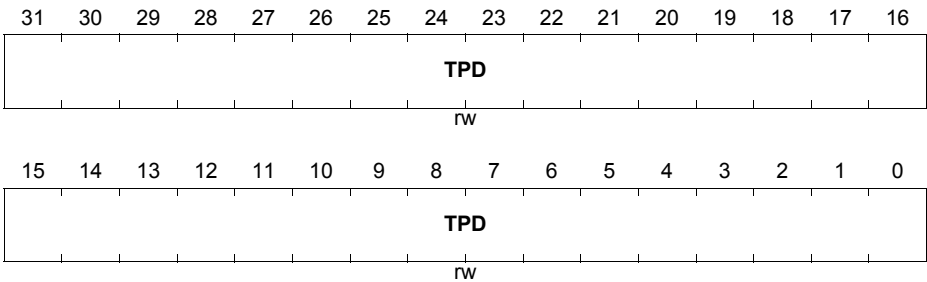
Field	Bits	Type	Description
PRWG	[29:28]	r	Channel Priority Weights This field sets the priority weights for Channel 0 during the round-robin arbitration between the DMA channels for the system bus. * 00: The priority weight is 1. * 01: The priority weight is 2. * 10: The priority weight is 3. * 11: The priority weight is 4. This field is present in all DWC_gmac configurations except GMAC-AXI when you select the AV feature. Otherwise, this field is reserved and read-only (RO).
RESERVE D_31_30	[31:30]	r	RESERVED_31_30

32-bit Register - Transmit_Poll_Demand

The Transmit Poll Demand register enables the Tx DMA to check whether or not the DMA owns the current descriptor. The Transmit Poll Demand command is given to wake up the Tx DMA if it is in the Suspend mode. The Tx DMA can go into the Suspend mode because of an Underflow error in a transmitted frame or the unavailability of descriptors owned by it. You can give this command anytime and the Tx DMA resets this command when it again starts fetching the current descriptor from host memory.

ETH_TRANSMIT_POLL_DEMAND

Register 1 - Transmit Poll Demand Register (2004_H) **Reset Value: 0000 0000_H**



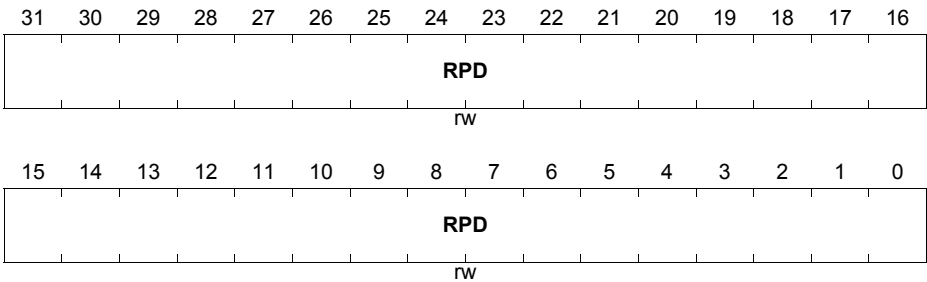
Field	Bits	Type	Description
TPD	[31:0]	rw	Transmit Poll Demand When these bits are written with any value, the DMA reads the current descriptor pointed to by Register 18 (Current Host Transmit Descriptor Register). If that descriptor is not available (owned by the Host), the transmission returns to the Suspend state and the Bit 2 (TU) of Register 5 (Status Register) is asserted. If the descriptor is available, the transmission resumes.

32-bit Register - Receive_Poll_Demand

The Receive Poll Demand register enables the receive DMA to check for new descriptors. This command is used to wake up the Rx DMA from the SUSPEND state. The RxDMA can go into the SUSPEND state only because of the unavailability of descriptors it owns.

ETH_RECEIVE_POLL_DEMAND

Register 2 - Receive Poll Demand Register (2008_H) **Reset Value: 0000 0000_H**



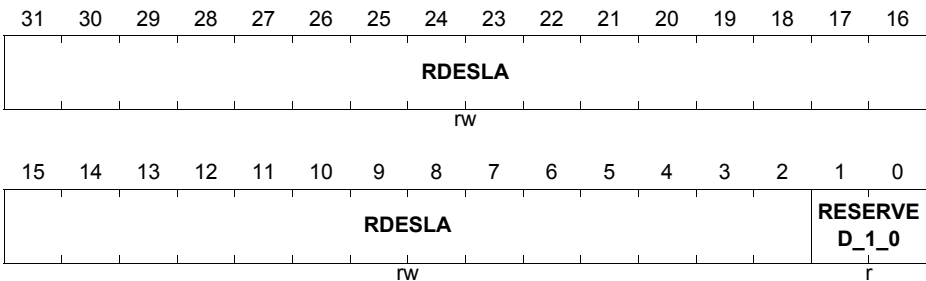
Field	Bits	Type	Description
RPD	[31:0]	rw	<p>Receive Poll Demand</p> <p>When these bits are written with any value, the DMA reads the current descriptor pointed to by Register 19 (Current Host Receive Descriptor Register). If that descriptor is not available (owned by the Host), the reception returns to the Suspended state and the Bit 7 (RU) of Register 5 (Status Register) is not asserted. If the descriptor is available, the Rx DMA returns to the active state.</p>

32-bit Register - Receive_Descriptor_List_Address

The Receive Descriptor List Address register points to the start of the Receive Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, Bit 1 (SR) is set to zero in Register 6 (Operation Mode Register). When stopped, this register can be written with a new descriptor list address. When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address. If this register is not changed when the SR bit is set to 0, then the DMA takes the descriptor address where it was stopped earlier.

ETH_RECEIVE_DESCRIPTOR_LIST_ADDRESS

Register 3 - Receive Descriptor List Address Register (200C_H) Reset Value: 0000 0000_H



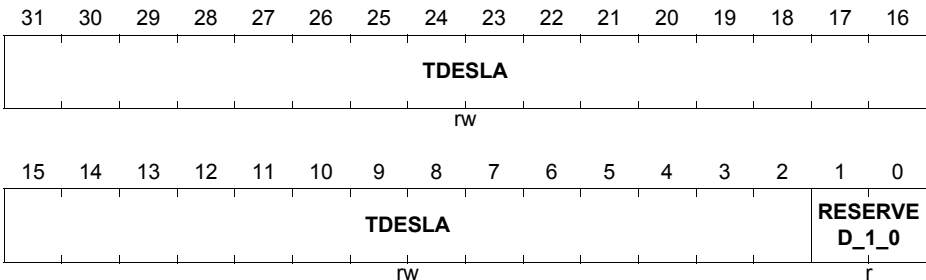
Field	Bits	Type	Description
RESERVE D_1_0	[1:0]	r	RESERVED_1_0
RDESLA	[31:2]	rw	Start of Receive List This field contains the base address of the first descriptor in the Receive Descriptor list. The LSB bits (1:0) for 32-bit bus width are ignored and internally taken as all-zero by the DMA. Therefore, these LSB bits are read-only (RO).

32-bit Register - Transmit_Descriptor_List_Address

The Transmit Descriptor List Address register points to the start of the Transmit Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low. You can write to this register only when the Tx DMA has stopped, that is, Bit 13 (ST) is set to zero in Register 6 (Operation Mode Register). When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly programmed descriptor base address. If this register is not changed when the ST bit is set to 0, then the DMA takes the descriptor address where it was stopped earlier.

ETH_TRANSMIT_DESCRIPTOR_LIST_ADDRESS

Register 4 - Transmit Descriptor List Address Register (2010_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
RESERVE D_1_0	[1:0]	r	RESERVED_1_0
TDESLA	[31:2]	rw	Start of Transmit List This field contains the base address of the first descriptor in the Transmit Descriptor list. The LSB bits (1:0) for 32-bit bus width are ignored and are internally taken as all-zero by the DMA. Therefore, these LSB bits are read-only (RO).

32-bit Register - Status

The Status register contains all status bits that the DMA reports to the host. The Software driver reads this register during an interrupt service routine or polling. Most of the fields in this register cause the host to be interrupted. The bits of this register are not cleared when read. Writing 1'b1 to (unreserved) Bits[16:0] of this register clears these bits and writing 1'b0 has no effect. Each field (Bits[16:0]) can be masked by masking the appropriate bit in Register 7 (Interrupt Enable Register).

ETH_STATUS
Register 5 - Status Register
(2014_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES ERV ED_ 31	GLPI I	TTI	GPI	GMI	GLI	EB			TS			RS			NIS
r	r	r	r	r	r	r			r			r			rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AIS	ERI	FBI	RESERVE D_12_11	ETI	RWT	RPS	RU	RI	UNF	OVF	TJT	TU	TPS	TI	
rw	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
TI	0	rw	Transmit Interrupt This bit indicates that the frame transmission is complete. When transmission is complete, the Bit 31 (Interrupt on Completion) of TDES1 is reset in the first descriptor, and the specific frame status information is updated in the descriptor.
TPS	1	rw	Transmit Process Stopped This bit is set when the transmission is stopped.
TU	2	rw	Transmit Buffer Unavailable This bit indicates that the host owns the Next Descriptor in the Transmit List and the DMA cannot acquire it. Transmission is suspended. Bits[22:20] explain the Transmit Process state transitions. To resume processing Transmit descriptors, the host should change the ownership of the descriptor by setting TDES0[31] and then issue a Transmit Poll Demand command.

Ethernet MAC (ETH)

Field	Bits	Type	Description
TJT	3	rw	<p>Transmit Jabber Timeout</p> <p>This bit indicates that the Transmit Jabber Timer expired, which happens when the frame size exceeds 2,048 (10,240 bytes when the Jumbo frame is enabled). When the Jabber Timeout occurs, the transmission process is aborted and placed in the Stopped state. This causes the Transmit Jabber Timeout TDES0[14] flag to assert.</p>
OVF	4	rw	<p>Receive Overflow</p> <p>This bit indicates that the Receive Buffer had an Overflow during frame reception. If the partial frame is transferred to the application, the overflow status is set in RDES0[11].</p>
UNF	5	rw	<p>Transmit Underflow</p> <p>This bit indicates that the Transmit Buffer had an Underflow during frame transmission. Transmission is suspended and an Underflow Error TDES0[1] is set.</p>
RI	6	rw	<p>Receive Interrupt</p> <p>This bit indicates that the frame reception is complete. When reception is complete, the Bit 31 of RDES1 (Disable Interrupt on Completion) is reset in the last Descriptor, and the specific frame status information is updated in the descriptor. The reception remains in the Running state.</p>
RU	7	rw	<p>Receive Buffer Unavailable</p> <p>This bit indicates that the host owns the Next Descriptor in the Receive List and the DMA cannot acquire it. The Receive Process is suspended. To resume processing Receive descriptors, the host should change the ownership of the descriptor and issue a Receive Poll Demand command. If no Receive Poll Demand is issued, the Receive Process resumes when the next recognized incoming frame is received. This bit is set only when the previous Receive Descriptor is owned by the DMA.</p>
RPS	8	rw	<p>Receive Process Stopped</p> <p>This bit is asserted when the Receive Process enters the Stopped state.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
RWT	9	rw	Receive Watchdog Timeout This bit is asserted when a frame with length greater than 2,048 bytes is received (10, 240 when Jumbo Frame mode is enabled).
ETI	10	rw	Early Transmit Interrupt This bit indicates that the frame to be transmitted is fully transferred to the MTL Transmit FIFO.
RESERVE D_12_11	[12:11]	r	RESERVED_12_11
FBI	13	rw	Fatal Bus Error Interrupt This bit indicates that a bus error occurred, as described in Bits[25:23]. When this bit is set, the corresponding DMA engine disables all of its bus accesses.
ERI	14	rw	Early Receive Interrupt This bit indicates that the DMA had filled the first data buffer of the packet. Bit 6 (RI) of this register automatically clears this bit.
AIS	15	rw	Abnormal Interrupt Summary Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in Register 7 (Interrupt Enable Register): <ul style="list-style-type: none"> * Register 5[1]: Transmit Process Stopped * Register 5[3]: Transmit Jabber Timeout * Register 5[4]: Receive FIFO Overflow * Register 5[5]: Transmit Underflow * Register 5[7]: Receive Buffer Unavailable * Register 5[8]: Receive Process Stopped * Register 5[9]: Receive Watchdog Timeout * Register 5[10]: Early Transmit Interrupt * Register 5[13]: Fatal Bus Error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit and must be cleared each time a corresponding bit, which causes AIS to be set, is cleared.

Ethernet MAC (ETH)

Field	Bits	Type	Description
NIS	16	rw	<p>Normal Interrupt Summary</p> <p>Normal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in Register 7 (Interrupt Enable Register):</p> <ul style="list-style-type: none"> * Register 5[0]: Transmit Interrupt * Register 5[2]: Transmit Buffer Unavailable * Register 5[6]: Receive Interrupt * Register 5[14]: Early Receive Interrupt <p>Only unmasked bits (interrupts for which interrupt enable is set in Register 7) affect the Normal Interrupt Summary bit.</p> <p>This is a sticky bit and must be cleared (by writing 1 to this bit) each time a corresponding bit, which causes NIS to be set, is cleared.</p>
RS	[19:17]	r	<p>Received Process State</p> <p>This field indicates the Receive DMA FSM state. This field does not generate an interrupt.</p> <ul style="list-style-type: none"> * 3'b000: Stopped: Reset or Stop Receive Command issued * 3'b001: Running: Fetching Receive Transfer Descriptor * 3'b010: Reserved for future use * 3'b011: Running: Waiting for receive packet * 3'b100: Suspended: Receive Descriptor Unavailable * 3'b101: Running: Closing Receive Descriptor * 3'b110: TIME_STAMP write state * 3'b111: Running: Transferring the receive packet data from receive buffer to host memory

Ethernet MAC (ETH)

Field	Bits	Type	Description
TS	[22:20]	r	Transmit Process State This field indicates the Transmit DMA FSM state. This field does not generate an interrupt. * 3'b000: Stopped; Reset or Stop Transmit Command issued * 3'b001: Running; Fetching Transmit Transfer Descriptor * 3'b010: Running; Waiting for status * 3'b011: Running; Reading Data from host memory buffer and queuing it to transmit buffer (Tx FIFO) * 3'b100: TIME_STAMP write state * 3'b101: Reserved for future use * 3'b110: Suspended; Transmit Descriptor Unavailable or Transmit Buffer Underflow * 3'b111: Running; Closing Transmit Descriptor
EB	[25:23]	r	Error Bits This field indicates the type of error that caused a Bus Error, for example, error response on the AHB or AXI interface. This field is valid only when Bit 13 (FBI) is set. This field does not generate an interrupt. * Bit 23 - 1'b1: Error during data transfer by the Tx DMA - 1'b0: Error during data transfer by the Rx DMA * Bit 24 - 1'b1: Error during read transfer - 1'b0: Error during write transfer * Bit 25 - 1'b1: Error during descriptor access - 1'b0: Error during data buffer access
GLI	26	r	GMAC Line interface Interrupt This bit reflects an interrupt event in the PCS (link change and AN complete), SMII (link change), or RGMII (link change) interface block of the DWC_gmac. The software must read the corresponding registers (Register 49 for PCS or Register 54 for SMII or RGMII) in the DWC_gmac to get the exact cause of the interrupt and clear the source of interrupt to make this bit as 1'b0. The interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high when this bit is high.

Ethernet MAC (ETH)

Field	Bits	Type	Description
GMI	27	r	<p>GMAC MMC Interrupt</p> <p>This bit reflects an interrupt event in the MMC module of the DWC_gmac. The software must read the corresponding registers in the DWC_gmac to get the exact cause of interrupt and clear the source of interrupt to make this bit as 1'b0. The interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high when this bit is high.</p> <p>This bit is applicable only when the MAC Management Counters (MMC) are enabled. Otherwise, this bit is reserved.</p>
GPI	28	r	<p>GMAC PMT Interrupt</p> <p>This bit indicates an interrupt event in the PMT module of the DWC_gmac. The software must read the PMT Control and Status Register in the MAC to get the exact cause of interrupt and clear its source to reset this bit to 1'b0. The interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high when this bit is high.</p> <p>This bit is applicable only when the Power Management feature is enabled. Otherwise, this bit is reserved.</p> <p>Note: This interrupt is different from the pmt_intr_o interrupt.</p>
TTI	29	r	<p>Timestamp Trigger Interrupt</p> <p>This bit indicates an interrupt event in the Timestamp Generator block of DWC_gmac. The software must read the corresponding registers in the DWC_gmac to get the exact cause of interrupt and clear its source to reset this bit to 1'b0. When this bit is high, the interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high.</p> <p>This bit is applicable only when the IEEE 1588 Timestamp feature is enabled. Otherwise, this bit is reserved.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
GLPII	30	r	GMAC LPI Interrupt (for Channel 0) This bit indicates an interrupt event in the LPI logic of the DWC_gmac. To reset this bit to 1'b0, the software must read the corresponding registers in the DWC_gmac to get the exact cause of the interrupt and clear its source. Note: GLPII status is given only in Channel 0 DMA register and is applicable only when the Energy Efficient Ethernet feature is enabled. Otherwise, this bit is reserved. When this bit is high, the interrupt signal from the MAC (sbd_intr_o) is high.
RESERVE D_31	31	r	RESERVED_31

32-bit Register - Operation_Mode

The Operation Mode register establishes the Transmit and Receive operating modes and commands. This register should be the last CSR to be written as part of the DMA initialization. This register is also present in the GMAC-MTL configuration with unused and reserved bits 24, 13, 2, and 1.

ETH_OPERATION_MODE
Register 6 - Operation Mode Register (2018_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED_31_27				DT	RSF	DFD	RFA_2	RFD_2	TSF	FTF	RESERVED_19_17			TTC	
r				rw	rw	rw	r	r	rw	rw	r			rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTC	ST	RFD	RFA		EFC	FEF	FUF	RESERVED_5	RTC		OSF	SR	RESERVED_0		
rw	rw	rw	rw		rw	rw	rw	r	rw		rw	rw	r		

Field	Bits	Type	Description
RESERVED_0	0	r	RESERVED_0

Ethernet MAC (ETH)

Field	Bits	Type	Description
SR	1	rw	<p>Start or Stop Receive</p> <p>When this bit is set, the Receive process is placed in the Running state. The DMA attempts to acquire the descriptor from the Receive list and processes the incoming frames. The descriptor acquisition is attempted from the current position in the list, which is the address set by Register 3 (Receive Descriptor List Address Register) or the position retained when the Receive process was previously stopped. If the DMA does not own the descriptor, reception is suspended and Bit 7 (Receive Buffer Unavailable) of Register 5 (Status Register) is set. The Start Receive command is effective only when the reception has stopped. If the command is issued before setting Register 3 (Receive Descriptor List Address Register), the DMA behavior is unpredictable. When this bit is cleared, the Rx DMA operation is stopped after the transfer of the current frame. The next descriptor position in the Receive list is saved and becomes the current position after the Receive process is restarted. The Stop Receive command is effective only when the Receive process is in either the Running (waiting for receive packet) or in the Suspended state.</p>
OSF	2	rw	<p>Operate on Second Frame</p> <p>When this bit is set, it instructs the DMA to process the second frame of the Transmit data even before the status for the first frame is obtained.</p>
RTC	[4:3]	rw	<p>Receive Threshold Control</p> <p>These two bits control the threshold level of the MTL Receive FIFO. Transfer (request) to DMA starts when the frame size within the MTL Receive FIFO is larger than the threshold. In addition, full frames with length less than the threshold are transferred automatically. The value of 11 is not applicable if the configured Receive FIFO size is 128 bytes. These bits are valid only when the RSF bit is zero, and are ignored when the RSF bit is set to 1.</p> <ul style="list-style-type: none"> * 00: 64 * 01: 32 * 10 : 96 * 11: 128

Ethernet MAC (ETH)

Field	Bits	Type	Description
RESERVE D_5	5	r	RESERVED_5
FUF	6	rw	<p>Forward Undersized Good Frames</p> <p>When set, the Rx FIFO forwards Undersized frames (frames with no Error and length less than 64 bytes) including pad-bytes and CRC.</p> <p>When reset, the Rx FIFO drops all frames of less than 64 bytes, unless a frame is already transferred because of the lower value of Receive Threshold, for example, RTC = 01.</p>
FEF	7	rw	<p>Forward Error Frames</p> <p>When this bit is reset, the Rx FIFO drops frames with error status (CRC error, collision error, GMII_ER, giant frame, watchdog timeout, or overflow). However, if the start byte (write) pointer of a frame is already transferred to the read controller side (in Threshold mode), then the frame is not dropped.</p> <p>In the GMAC-MTL configuration in which the Frame Length FIFO is also enabled during core configuration, the Rx FIFO drops the error frames if that frame's start byte is not transferred (output) on the ARI bus.</p> <p>When the FEF bit is set, all frames except runt error frames are forwarded to the DMA. If the Bit 25 (RSF) is set and the Rx FIFO overflows when a partial frame is written, then the frame is dropped irrespective of the FEF bit setting. However, if the Bit 25 (RSF) is reset and the Rx FIFO overflows when a partial frame is written, then a partial frame may be forwarded to the DMA.</p>
EFC	8	rw	<p>Enable HW Flow Control</p> <p>When this bit is set, the flow control signal operation based on the fill-level of Rx FIFO is enabled. When reset, the flow control operation is disabled. This bit is not used (reserved and always reset) when the Rx FIFO is less than 4 KB.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
RFA	[10:9]	rw	<p>Threshold for Activating Flow Control (in half-duplex and full-duplex)</p> <p>These bits control the threshold (Fill level of Rx FIFO) at which the flow control is activated.</p> <ul style="list-style-type: none"> - 00: Full minus 1 KB, that is, FULL - 1KB - 01: Full minus 2 KB, that is, FULL - 2KB - 10: Full minus 3 KB, that is, FULL - 3KB - 11: Full minus 4 KB, that is, FULL - 4KB <p>These values only apply to Rx FIFOs of 4 KB or more when the EFC bit is set high. If the Rx FIFO is 8 KB or more, an additional bit (RFA_2) is used for more threshold levels as described in Bit 23. These bits are reserved and read-only when the depth of Rx FIFO is less than 4 KB.</p>
RFD	[12:11]	rw	<p>Threshold for Deactivating Flow Control (in half-duplex and full-duplex)</p> <p>These bits control the threshold (Fill-level of Rx FIFO) at which the flow control is de-asserted after activation.</p> <ul style="list-style-type: none"> - 00: Full minus 1 KB, that is, FULL - 1KB - 01: Full minus 2 KB, that is, FULL - 2KB - 10: Full minus 3 KB, that is, FULL - 3KB - 11: Full minus 4 KB, that is, FULL - 4KB <p>The de-assertion is effective only after flow control is asserted. If the Rx FIFO is 8 KB or more, an additional bit (RFD_2) is used for more threshold levels as described in Bit 22. These bits are reserved and read-only when the Rx FIFO depth is less than 4 KB.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
ST	13	rw	<p>Start or Stop Transmission Command</p> <p>When this bit is set, transmission is placed in the Running state, and the DMA checks the Transmit List at the current position for a frame to be transmitted. Descriptor acquisition is attempted either from the current position in the list, which is the Transmit List Base Address set by Register 4 (Transmit Descriptor List Address Register), or from the position retained when transmission was stopped previously. If the DMA does not own the current descriptor, transmission enters the Suspended state and Bit 2 (Transmit Buffer Unavailable) of Register 5 (Status Register) is set. The Start Transmission command is effective only when transmission is stopped. If the command is issued before setting Register 4 (Transmit Descriptor List Address Register), then the DMA behavior is unpredictable.</p> <p>When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current frame. The Next Descriptor position in the Transmit List is saved, and it becomes the current position when transmission is restarted. To change the list address, you need to program Register 4 (Transmit Descriptor List Address Register) with a new value when this bit is reset. The new value is considered when this bit is set again. The stop transmission command is effective only when the transmission of the current frame is complete or the transmission is in the Suspended state.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
TTC	[16:14]	rw	<p>Transmit Threshold Control</p> <p>These bits control the threshold level of the MTL Transmit FIFO. Transmission starts when the frame size within the MTL Transmit FIFO is larger than the threshold. In addition, full frames with a length less than the threshold are also transmitted. These bits are used only when Bit 21 (TSF) is reset.</p> <ul style="list-style-type: none"> * 000: 64 * 001: 128 * 010: 192 * 011: 256 * 100: 40 * 101: 32 * 110: 24 * 111: 16
RESERVE D_19_17	[19:17]	r	RESERVED_19_17
FTF	20	rw	<p>Flush Transmit FIFO</p> <p>When this bit is set, the transmit FIFO controller logic is reset to its default values and thus all data in the Tx FIFO is lost or flushed. This bit is cleared internally when the flushing operation is completed. The Operation Mode register should not be written to until this bit is cleared. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt frame transmission.</p> <p>Note: The flush operation is complete only when the Tx FIFO is emptied of its contents and all the pending Transmit Status of the transmitted frames are accepted by the host. To complete this flush operation, the PHY transmit clock (clk_tx_i) is required to be active.</p>
TSF	21	rw	<p>Transmit Store and Forward</p> <p>When this bit is set, transmission starts when a full frame resides in the MTL Transmit FIFO. When this bit is set, the TTC values specified in Bits[16:14] are ignored. This bit should be changed only when the transmission is stopped.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
RFD_2	22	r	<p>MSB of Threshold for Deactivating Flow Control</p> <p>If the DWC_gmac is configured for Rx FIFO size of 8 KB or more, this bit (when set) provides additional threshold levels for deactivating the flow control in both half-duplex and full-duplex modes. This bit (as Most Significant Bit) along with the RFD (Bits[12:11]) gives the following thresholds for deactivating flow control:</p> <ul style="list-style-type: none"> * 100: Full minus 5 KB, that is, FULL - 5KB * 101: Full minus 6 KB, that is, FULL - 6KB * 110: Full minus 7 KB, that is, FULL - 7KB * 111: Reserved <p>This bit is reserved (and RO) if the Rx FIFO is 4 KB or less deep.</p>
RFA_2	23	r	<p>MSB of Threshold for Activating Flow Control</p> <p>If the DWC_gmac is configured for an Rx FIFO depth of 8 KB or more, this bit (when set) provides additional threshold levels for activating the flow control in both half-duplex and full-duplex modes. This bit (as Most Significant Bit) along with the RFA (Bits[10:9]) gives the following thresholds for activating flow control:</p> <ul style="list-style-type: none"> * 100: Full minus 5 KB, that is, FULL - 5KB * 101: Full minus 6 KB, that is, FULL - 6KB * 110: Full minus 7 KB, that is, FULL - 7KB * 111: Reserved <p>This bit is reserved (and RO) if the Rx FIFO is 4 KB or less deep.</p>
DFF	24	rw	<p>Disable Flushing of Received Frames</p> <p>When this bit is set, the Rx DMA does not flush any frames because of the unavailability of receive descriptors or buffers as it does normally when this bit is reset.</p> <p>This bit is reserved (and RO) in the GMAC-MTL configuration.</p>

Ethernet MAC (ETH)

Field	Bits	Type	Description
RSF	25	rw	Receive Store and Forward When this bit is set, the MTL reads a frame from the Rx FIFO only after the complete frame has been written to it, ignoring the RTC bits. When this bit is reset, the Rx FIFO operates in the cut-through mode, subject to the threshold specified by the RTC bits.
DT	26	rw	Disable Dropping of TCP/IP Checksum Error Frames When this bit is set, the MAC does not drop the frames which only have errors detected by the Receive Checksum Offload engine. Such frames do not have any errors (including FCS error) in the Ethernet frame received by the MAC but have errors only in the encapsulated payload. When this bit is reset, all error frames are dropped if the FEF bit is reset. If the IPC Full Checksum Offload Engine (Type 2) is disabled, this bit is reserved (RO with value 1'b0).
RESERVE D_31_27	[31:27]	r	RESERVED_31_27

32-bit Register - Interrupt_Enable

The Interrupt Enable register enables the interrupts reported by Register 5 (Status Register). Setting a bit to 1'b1 enables a corresponding interrupt. After a hardware or software reset, all interrupts are disabled.

ETH_INTERRUPT_ENABLE

Register 7 - Interrupt Enable Register (201C_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED_31_17															NIE
r															rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AIE	ERE	FBE	RESERVE D_12_11	ETE	RWE	RSE	RUE	RIE	UNE	OVE	TJE	TUE	TSE	TIE	
rw	rw	rw	r	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Field	Bits	Type	Description
TIE	0	rw	Transmit Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Transmit Interrupt is enabled. When this bit is reset, the Transmit Interrupt is disabled.
TSE	1	rw	Transmit Stopped Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Transmission Stopped Interrupt is enabled. When this bit is reset, the Transmission Stopped Interrupt is disabled.
TUE	2	rw	Transmit Buffer Unavailable Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Transmit Buffer Unavailable Interrupt is enabled. When this bit is reset, the Transmit Buffer Unavailable Interrupt is disabled.
TJE	3	rw	Transmit Jabber Timeout Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Transmit Jabber Timeout Interrupt is enabled. When this bit is reset, the Transmit Jabber Timeout Interrupt is disabled.

Ethernet MAC (ETH)

Field	Bits	Type	Description
OVE	4	rw	Overflow Interrupt Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Overflow Interrupt is enabled. When this bit is reset, the Overflow Interrupt is disabled.
UNE	5	rw	Underflow Interrupt Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Transmit Underflow Interrupt is enabled. When this bit is reset, the Underflow Interrupt is disabled.
RIE	6	rw	Receive Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Receive Interrupt is enabled. When this bit is reset, the Receive Interrupt is disabled.
RUE	7	rw	Receive Buffer Unavailable Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Buffer Unavailable Interrupt is enabled. When this bit is reset, the Receive Buffer Unavailable Interrupt is disabled.
RSE	8	rw	Receive Stopped Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Stopped Interrupt is enabled. When this bit is reset, the Receive Stopped Interrupt is disabled.
RWE	9	rw	Receive Watchdog Timeout Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Watchdog Timeout Interrupt is enabled. When this bit is reset, the Receive Watchdog Timeout Interrupt is disabled.
ETE	10	rw	Early Transmit Interrupt Enable When this bit is set with an Abnormal Interrupt Summary Enable (Bit 15), the Early Transmit Interrupt is enabled. When this bit is reset, the Early Transmit Interrupt is disabled.
RESERVE D_12_11	[12:11]	r	RESERVED_12_11

Ethernet MAC (ETH)

Field	Bits	Type	Description
FBE	13	rw	Fatal Bus Error Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Fatal Bus Error Interrupt is enabled. When this bit is reset, the Fatal Bus Error Enable Interrupt is disabled.
ERE	14	rw	Early Receive Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Early Receive Interrupt is enabled. When this bit is reset, the Early Receive Interrupt is disabled.
AIE	15	rw	Abnormal Interrupt Summary Enable When this bit is set, abnormal interrupt summary is enabled. When this bit is reset, the abnormal interrupt summary is disabled. This bit enables the following interrupts in Register 5 (Status Register): <ul style="list-style-type: none"> * Register 5[1]: Transmit Process Stopped * Register 5[3]: Transmit Jabber Timeout * Register 5[4]: Receive Overflow * Register 5[5]: Transmit Underflow * Register 5[7]: Receive Buffer Unavailable * Register 5[8]: Receive Process Stopped * Register 5[9]: Receive Watchdog Timeout * Register 5[10]: Early Transmit Interrupt * Register 5[13]: Fatal Bus Error
NIE	16	rw	Normal Interrupt Summary Enable When this bit is set, normal interrupt summary is enabled. When this bit is reset, normal interrupt summary is disabled. This bit enables the following interrupts in Register 5 (Status Register): <ul style="list-style-type: none"> * Register 5[0]: Transmit Interrupt * Register 5[2]: Transmit Buffer Unavailable * Register 5[6]: Receive Interrupt * Register 5[14]: Early Receive Interrupt
RESERVE D_31_17	[31:17]	r	RESERVED_31_17

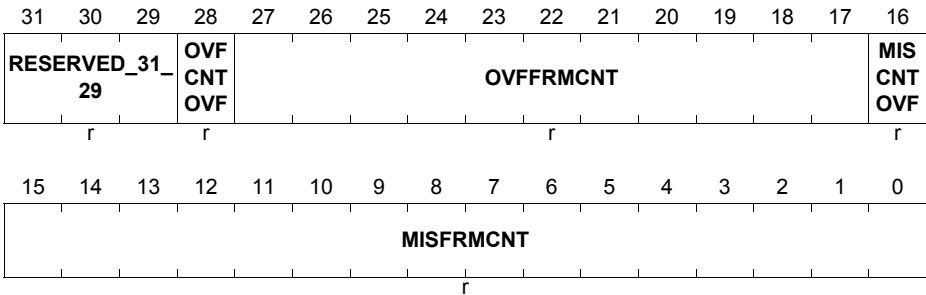
Ethernet MAC (ETH)

32-bit Register - Missed_Frame_And_Buffer_Overflow_Counter

The DMA maintains two counters to track the number of frames missed during reception. This register reports the current value of the counter. The counter is used for diagnostic purposes. Bits[15:0] indicate missed frames because of the host buffer being unavailable. Bits[27:17] indicate missed frames because of buffer overflow conditions (MTL and MAC) and runt frames (good frames of less than 64 bytes) dropped by the MTL.

ETH_MISSED_FRAME_AND_BUFFER_OVERFLOW_COUNTER

Register 8 - Missed Frame and Buffer Overflow Counter Register (2020_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
MISFRMCNT	[15:0]	r	MISFRMCNT This field indicates the number of frames missed by the controller because of the Host Receive Buffer being unavailable. This counter is incremented each time the DMA discards an incoming frame. The counter is cleared when this register is read with mci_be_i[0] at 1'b1.
MISCNTOVF	16	r	MISCNTOVF Overflow bit for Missed Frame Counter
OVFFRMCNT	[27:17]	r	OVFFRMCNT This field indicates the number of frames missed by the application. This counter is incremented each time the MTL asserts the sideband signal mtl_rxoverflow_o. The counter is cleared when this register is read with mci_be_i[2] at 1'b1.

Ethernet MAC (ETH)

Field	Bits	Type	Description
OVFCNTOVF	28	r	OVFCNTOVF Overflow bit for FIFO Overflow Counter
RESERVED_31_29	[31:29]	r	RESERVED_31_29

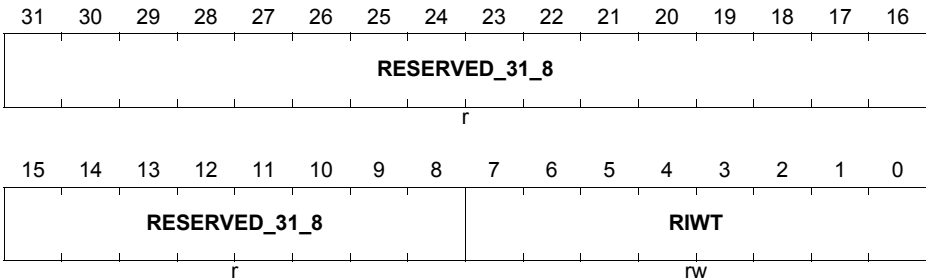
Ethernet MAC (ETH)

32-bit Register - Receive_Interrupt_Watchdog_Timer

This register, when written with non-zero value, enables the watchdog timer for the Receive Interrupt (Bit 6) of Register 5 (Status Register)

ETH_RECEIVE_INTERRUPT_WATCHDOG_TIMER

Register 9 - Receive Interrupt Watchdog Timer Register (2024_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
RIWT	[7:0]	rw	RI Watchdog Timer Count This bit indicates the number of system clock cycles multiplied by 256 for which the watchdog timer is set. The watchdog timer gets triggered with the programmed value after the Rx DMA completes the transfer of a frame for which the RI status bit is not set because of the setting in the corresponding descriptor RDES1[31]. When the watchdog timer runs out, the RI bit is set and the timer is stopped. The watchdog timer is reset when the RI bit is set high because of automatic setting of RI as per RDES1[31] of any received frame.
RESERVE D_31_8	[31:8]	r	RESERVED_31_8

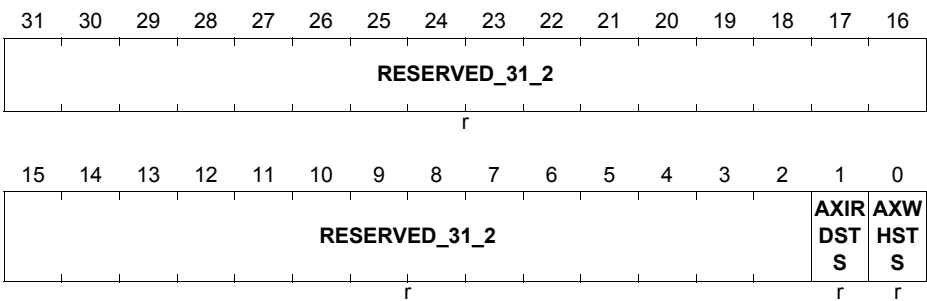
32-bit Register - AHB_or_AXI_Status

This register provides the active status of the AHB master interface or AXI interface's read and write channels. This register is present and valid only in the GMAC-AHB and GMAC-AXI configurations. This register is useful for debugging purposes. In addition, this register is valid only in the Channel 0 DMA when multiple channels are present in the AV mode.

ETH_AHB_OR_AXI_STATUS

Register 11 - AHB or AXI Status Register (202C_H)

Reset Value: 0000 0000_H



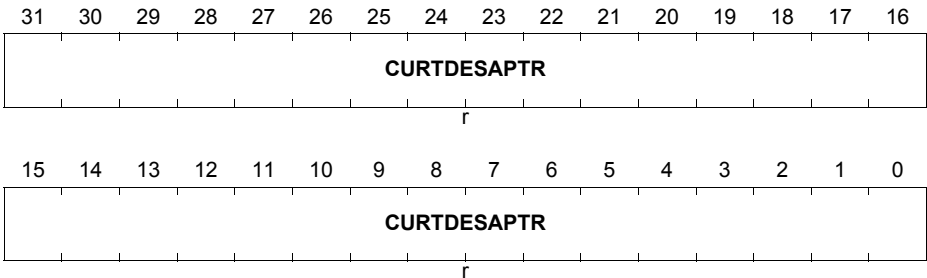
Field	Bits	Type	Description
AXWHSTS	0	r	AXI Master Write Channel or AHB Master Status When high, it indicates that AXI Master's write channel is active and transferring data in the GMAC-AXI configuration. In the GMAC-AHB configuration, it indicates that the AHB master interface FSMs are in the non-idle state.
AXIRDSTS	1	r	AXI Master Read Channel Status When high, it indicates that AXI Master's read channel is active and transferring data.
RESERVE D_31_2	[31:2]	r	RESERVED_31_2

32-bit Register - Current_Host_Transmit_Descriptor

The Current Host Transmit Descriptor register points to the start address of the current Transmit Descriptor read by the DMA.

ETH_CURRENT_HOST_TRANSMIT_DESCRIPTOR

Register 18 - Current Host Transmit Descriptor Register (2048_H) Reset Value: 0000 0000_H



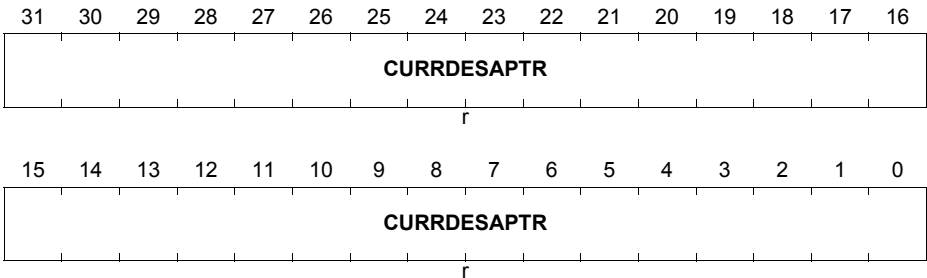
Field	Bits	Type	Description
CURTDES APTR	[31:0]	r	Host Transmit Descriptor Address Pointer Cleared on Reset. Pointer updated by the DMA during operation.

32-bit Register - Current_Host_Receive_Descriptor

The Current Host Receive Descriptor register points to the start address of the current Receive Descriptor read by the DMA.

ETH_CURRENT_HOST_RECEIVE_DESCRIPTOR

Register 19 - Current Host Receive Descriptor Register (204C_H) Reset Value: 0000 0000_H



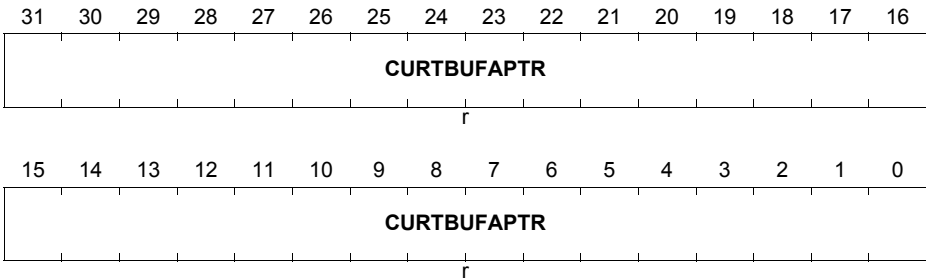
Field	Bits	Type	Description
CURRDES APTR	[31:0]	r	Host Receive Descriptor Address Pointer Cleared on Reset. Pointer updated by the DMA during operation.

32-bit Register - Current_Host_Transmit_Buffer_Address

The Current Host Transmit Buffer Address register points to the current Transmit Buffer Address being read by the DMA.

ETH_CURRENT_HOST_TRANSMIT_BUFFER_ADDRESS

Register 20 - Current Host Transmit Buffer Address Register (2050_H) Reset Value: 0000 0000_H



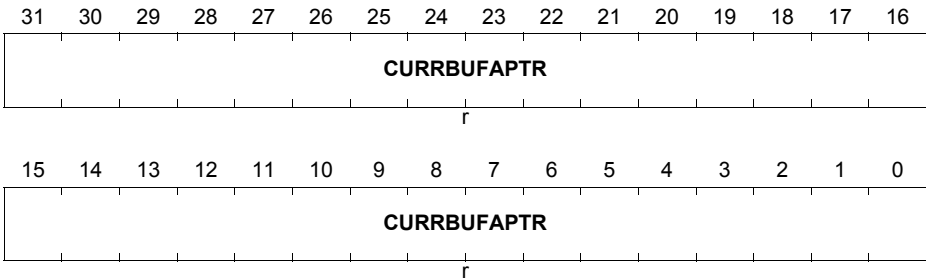
Field	Bits	Type	Description
CURTBUF APTR	[31:0]	r	Host Transmit Buffer Address Pointer Cleared on Reset. Pointer updated by the DMA during operation.

32-bit Register - Current_Host_Receive_Buffer_Address

The Current Host Receive Buffer Address register points to the current Receive Buffer address being read by the DMA.

ETH_CURRENT_HOST_RECEIVE_BUFFER_ADDRESS

Register 21 - Current Host Receive Buffer Address Register (2054_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
CURRBUF APTR	[31:0]	r	Host Receive Buffer Address Pointer Cleared on Reset. Pointer updated by the DMA during operation.

32-bit Register - HW_Feature

This register indicates the presence of the optional features or functions of the DWC_gmac. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks. Note: All bits are set or reset as per the selection of features during the DWC_gmac configuration.

ETH_HW_FEATURE
Register 22 - HW Feature Register (2058_H)
Reset Value:030D 2F35_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RES ERV ED_ 31	ACTPHYIF			SAV LANI NS	FLE XIPP SEN	INTT SEN	ENH DES SEL	TXCHC CNT	RXCHC CNT		RXFI FOSI ZE	RXT YP2 COE	RXT YP1 COE	TXC OES EL	
r	r			r	r	r	r	r		r		r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AVS EL	EEE SEL	TSV ER2 SEL	TSV ER1 SEL	MMC SEL	MGK SEL	RWK SEL	SMA SEL	L3L4 FLT REN	PCS SEL	ADD MAC ADR SEL	HAS HSE L	EXT HAS HEN	HDS EL	GMI SEL	MIIS EL
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
MIISEL	0	r	MIISEL 10 or 100 Mbps support
GMIISEL	1	r	GMIISEL 1000 Mbps support
HDSEL	2	r	HDSEL Half-Duplex support
EXTHASH EN	3	r	EXTHASHEN Expanded DA Hash Filter
HASHSEL	4	r	HASHSEL HASH Filter
ADDMAC ADRSEL	5	r	ADDMACADRSEL Multiple MAC Address Registers
PCSSEL	6	r	PCSSEL PCS registers (TBI, SGMII, or RTBI PHY interface)

Ethernet MAC (ETH)

Field	Bits	Type	Description
L3L4FLTR EN	7	r	L3L4FLTR EN Layer 3 and Layer 4 Filter Feature
SMASEL	8	r	SMASEL SMA (MDIO) Interface
RWKSEL	9	r	RWKSEL PMT Remote Wakeup
MGKSEL	10	r	MGKSEL PMT Magic Packet
MMCSEL	11	r	MMCSEL RMON Module
TSVER1S EL	12	r	TSVER1SEL Only IEEE 1588-2002 Timestamp
TSVER2S EL	13	r	TSVER2SEL IEEE 1588-2008 Advanced Timestamp
EEESEL	14	r	EEESEL Energy Efficient Ethernet
AVSEL	15	r	AVSEL AV Feature
TXCOESE L	16	r	TXCOESEL Checksum Offload in Tx
RXTYP1C OE	17	r	RXTYP1COE IP Checksum Offload (Type 1) in Rx
RXTYP2C OE	18	r	RXTYP2COE IP Checksum Offload (Type 2) in Rx
RXFIFOSI ZE	19	rw	RXFIFOSIZE Rx FIFO > 2,048 Bytes
RXCHCNT	[21:20]	r	RXCHCNT Number of additional Rx channels
TXCHCNT	[23:22]	r	TXCHCNT Number of additional Tx channels
ENHDESS EL	24	r	ENHDESSEL Alternate (Enhanced Descriptor)
INTTSEN	25	r	INTTSEN Timestamping with Internal System Time

Ethernet MAC (ETH)

Field	Bits	Type	Description
FLEXIPPS EN	26	r	FLEXIPPSEN Flexible Pulse-Per-Second Output
SAVLANI NS	27	r	SAVLANINS Source Address or VLAN Insertion
ACTPHYIF	[30:28]	r	Active or Selected PHY interface When you have multiple PHY interfaces in your configuration, this field indicates the sampled value of phy_intf_sel_i during reset de-assertion * 0000: GMII or MII * 0001: RGMII * 0010: SGMII * 0011: TBI * 0100: RMII * 0101: RTBI * 0110: SMII * 0111: RevMII * All Others: Reserved
RESERVE D_31	31	r	RESERVED_31

35.4 Descriptors

This chapter comprises the following subsections:

- **Normal Descriptor Formats**
- **Alternate or Enhanced Descriptors**

*Note: The Ethernet MAC in TC26x B-Step does not support Normal Descriptor Format as it is configured to support IEEE1588 PTP time stamping! Nevertheless parts of the explanations in the chapter **Chapter 1.7.1** are required for complete understanding.*

35.4.1 Normal Descriptor Formats

The DMA in the Ethernet subsystem transfers data based on a linked list of descriptors, as explained in **DMA Controller**. The default descriptor formats (common for both Receive and Transmit Descriptors) are shown in **Figure 35-24**, and field descriptions are provided in **Chapter 1.7.1.1** to **Chapter 1.7.1.2**.

Note:

3. All descriptions in **Chapter 1.7.1.1** and **Chapter 1.7.1.2** correspond to the default descriptor format. The alternate enhanced descriptor format is discussed in **Chapter 1.7.2** “Alternate or Enhanced Descriptors” on **Page 1-680**.
4. Changes to the default descriptor format when IEEE1588 time stamping is enabled are described in **Chapter 1.7.1.3** “Descriptor Format With IEEE 1588 Time Stamping Enabled” on **Page 1-676**.

Each descriptor contains two buffers, two byte-count buffers, and two address pointers, which enable the adapter port to be compatible with various types of memory management schemes.

The descriptor addresses must be aligned to the bus width used (Word/Dword/Lword for 32-bit buses).

Figure 35-24 shows the descriptor format in a 32-bit data bus.

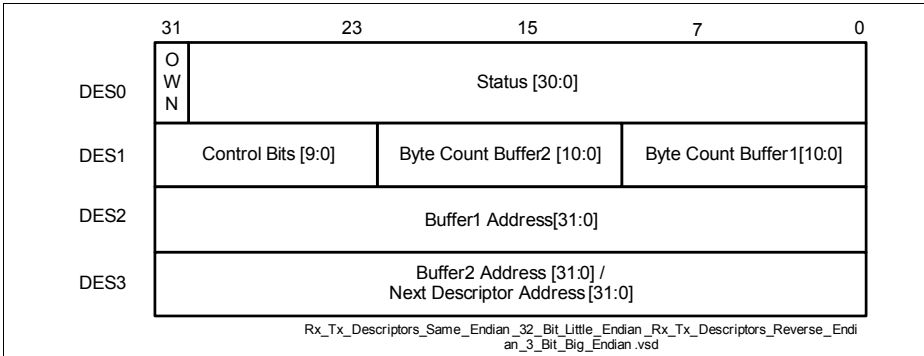


Figure 35-24 Rx/Tx Descriptors for 32-Bit Data Bus

35.4.1.1 Receive Descriptor

The GMAC Subsystem requires at least two descriptors when receiving a frame. The Receive state machine of the DMA (in the GMAC Subsystem) always attempts to acquire an extra descriptor in anticipation of an incoming frame. (The size of the incoming frame is unknown). Before the RxDMA closes a descriptor, it will attempt to acquire the next descriptor even if no frames are received.

In a single descriptor (receive) system, the subsystem will generate a descriptor error if the receive buffer is unable to accommodate the incoming frame and the next descriptor is not owned by the DMA. Thus, the Host is forced to increase either its descriptor pool or the buffer size. Otherwise, the subsystem starts dropping all incoming frames.

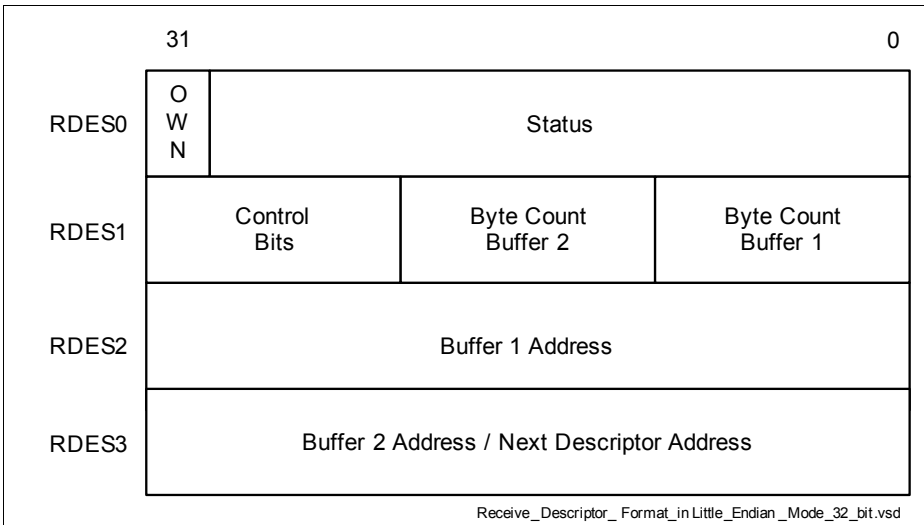


Figure 35-25 Receive Descriptor Format in Little-Endian Mode With a 32-bit Data Bus

Receive Descriptor 0 (RDES0)

RDES0 contains the received frame status, the frame length, and the descriptor ownership information. The descriptions in the following tables (Table 1-83 through Table 1-91) are for the default mode of a Little-Endian 32-bit data bus with Same Endian descriptors, or Big-Endian data bus with Reverse-Endian descriptors. If the DUT is configured for Big-Endian mode, 32-bit data bus with Same Endian descriptors or Little-Endian data bus with Reverse Endian descriptors, then byte lanes 3 and 0 are swapped while byte lanes 1 and 2 are swapped, that is, bits [31:24] will be available on data bus [7:0], and vice-versa.

Table 35-23 Receive Descriptor 0

Bit	Description
31	OWN: Own Bit When set, this bit indicates that the descriptor is owned by the DMA of the GMAC Subsystem. When this bit is reset, this bit indicates that the descriptor is owned by the Host. The DMA clears this bit either when it completes the frame reception or when the buffers that are associated with this descriptor are full.
30	AFM: Destination Address Filter Fail When set, this bit indicates a frame that failed in the DA Filter in the GMAC Core.

Table 35-23 Receive Descriptor 0 (cont'd)

Bit	Description
29:1 6	<p>FL: Frame Length</p> <p>These bits indicate the byte length of the received frame that was transferred to host memory (including CRC). This field is valid when Last Descriptor (RDES0[8]) is set and either the Descriptor Error (RDES0[14]) or Overflow Error bits are reset. The frame length also includes the two bytes appended to the Ethernet frame when IP checksum calculation (Type 1) is enabled and the received frame is not a MAC control frame.</p> <p>This field is valid when Last Descriptor (RDES0[8]) is set. When the Last Descriptor and Error Summary bits are not set, this field indicates the accumulated number of bytes that have been transferred for the current frame.</p>
15	<p>ES: Error Summary</p> <p>Indicates the logical OR of the following bits:</p> <ul style="list-style-type: none"> • RDES0[0]: Payload Checksum Error • RDES0[1]: CRC Error • RDES0[3]: Receive Error • RDES0[4]: Watchdog Timeout • RDES0[6]: Late Collision • RDES0[7]: IPC Checksum (Type 2) / Giant Frame • RDES0[11]: Overflow Error • RDES0[14]: Descriptor Error <p>This field is valid only when the Last Descriptor (RDES0[8]) is set.</p>
14	<p>DE: Descriptor Error</p> <p>When set, this bit indicates a frame truncation caused by a frame that does not fit within the current descriptor buffers, and that the DMA does not own the Next Descriptor. The frame is truncated. This field is valid only when the Last Descriptor (RDES0[8]) is set.</p>
13	<p>SAF: Source Address Filter Fail</p> <p>When set, this bit indicates that the SA field of frame failed the SA Filter in the GMAC Core.</p>
12	<p>LE: Length Error</p> <p>When set, this bit indicates that the actual length of the frame received and that the Length/ Type field does not match. This bit is valid only when the Frame Type (RDES0[5]) bit is reset. Length error status is not valid when CRC error is present.</p>
11	<p>OE: Overflow Error</p> <p>When set, this bit indicates that the received frame was damaged due to buffer overflow in MTL.</p>

Table 35-23 Receive Descriptor 0 (cont'd)

Bit	Description
10	VLAN: VLAN Tag When set, this bit indicates that the frame pointed to by this descriptor is a VLAN frame tagged by the GMAC Core.
9	FS: First Descriptor When set, this bit indicates that this descriptor contains the first buffer of the frame. If the size of the first buffer is 0, the second buffer contains the beginning of the frame. If the size of the second buffer is also 0, the next Descriptor contains the beginning of the frame.
8	LS: Last Descriptor When set, this bit indicates that the buffers pointed to by this descriptor are the last buffers of the frame
7	IPC Checksum Error/Giant Frame When IP Checksum Engine (Type 1) is enabled, this bit, when set, indicates that the 16-bit IPv4 Header checksum calculated by the core did not match the received checksum bytes. The Error Summary bit[15] is NOT set when this bit is set in this mode. If this bit is set when Full Checksum Offload Engine (Type 2) is enabled, it indicates an error in the IPv4 or IPv6 header. This error can be due to inconsistent Ethernet Type field and IP header Version field values, a header checksum mismatch in IPv4, or an Ethernet frame lacking the expected number of IP header bytes. Refer to Table 1-84 for more details. If you do not select IP Checksum Module during core configuration, this bit, when set, indicates that the received frame was a Giant Frame. Giant frames are larger-than-1.518-byte (or 1.522-byte for VLAN) normal frames and larger-than-9,018-byte (9.022-byte for VLAN) frame when Jumbo Frame processing is enabled.
6	LC: Late Collision When set, this bit indicates that a late collision has occurred while receiving the frame in Half-Duplex mode.
5	FT: Frame Type When set, this bit indicates that the Receive Frame is an Ethernet-type frame (the LT field is greater than or equal to 16'h0600). When this bit is reset, it indicates that the received frame is an IEEE802.3 frame. This bit is not valid for Runt frames less than 14 bytes. Refer to Table 1-84 for more details.
4	RWT: Receive Watchdog Timeout When set, this bit indicates that the Receive Watchdog Timer has expired while receiving the current frame and the current frame is truncated after the Watchdog Timeout.

Table 35-23 Receive Descriptor 0 (cont'd)

Bit	Description
3	RE: Receive Error When set, this bit indicates that the gmii_rxr_i signal is asserted while gmii_rxdv_i is asserted during frame reception. This error also includes carrier extension error in GMII and Half-duplex mode. Error can be of less/no extension, or error (rxd ≠ 0f) during extension.
2	DE: Dribble Bit Error When set, this bit indicates that the received frame has a non-integer multiple of bytes (odd nibbles). This bit is valid only in MII Mode.
1	CE: CRC Error When set, this bit indicates that a Cyclic Redundancy Check (CRC) Error occurred on the received frame. This field is valid only when the Last Descriptor (RDES0[8]) is set.
0	Rx MAC Address/Payload Checksum Error When set, this bit indicates that the Rx MAC Address registers value (1 to 15) matched the frame's DA field. When reset, this bit indicates that the Rx MAC Address Register 0 value matched the DA field. If Full Checksum Offload Engine is enabled, this bit, when set, indicates the TCP, UDP, or ICMP checksum the core calculated does not match the received encapsulated TCP, UDP, or ICMP segment's Checksum field. This bit is also set when the received number of payload bytes does not match the value indicated in the Length field of the encapsulated IPv4 or IPv6 datagram in the received Ethernet frame. Refer to Table 1-84 for more details.

When the Full Checksum Offload Engine (Type 2) is enabled, the permutations of bits 5, 7, and 0 reflect the conditions discussed in [Table 1-84](#).

Table 35-24 Receive Descriptor 0 When COE (Type 2) Is Enabled

Bit 5: Frame Type	Bit 7: IPC Checksum Error	Bit 0: Payload Checksum Error	Frame Status
0	0	0	IEEE 802.3 Type frame (Length field value is less than 0600 _H)
1	0	0	IPv4/IPv6 Type frame, no checksum error detected
1	0	1	IPv4/IPv6 Type frame with a payload checksum error (as described for PCE) detected

Table 35-24 Receive Descriptor 0 When COE (Type 2) Is Enabled (cont'd)

Bit 5: Frame Type	Bit 7: IPC Checks um Error	Bit 0: Payload Checks um Error	Frame Status
1	1	0	IPv4/IPv6 Type frame with an IP header checksum error (as described for IPC CE) detected
1	1	1	IPv4/IPv6 Type frame with both IP header and payload checksum errors detected
0	0	1	IPv4/IPv6 Type frame with no IP header checksum error and the payload check bypassed, due to an unsupported payload
0	1	1	A Type frame that is neither IPv4 or IPv6 (the Checksum Offload engine bypasses checksum completely.)
0	1	0	Reserved

Note: The first five conditions are backward-compatible to versions 3.30a and previous. The last two conditions (001, 011), which had been reserved, are not backward-compatible, because the Frame Type (FT) bit is reset during bypasses, even for valid Type frames.

Receive Descriptor 1 (RDES1)

RDES1 contains the buffer sizes and other bits that control the descriptor chain/ring.

Note: See [Buffer Size Calculations](#) for further detail on calculating buffer sizes.

Table 35-25 Receive Descriptor 1

Bit	Description
31	Disable Interrupt on Completion When set, this bit will prevent the setting of the RI (CSR5[6]) bit of the Status Register for the received frame that ends in the buffer pointed to by this descriptor. This, in turn, will disable the assertion of the interrupt to Host due to RI for that frame.
30:26	Reserved
25	RER: Receive End of Ring When set, this bit indicates that the descriptor list reached its final descriptor. The DMA returns to the base address of the list, creating a Descriptor Ring.

Table 35-25 Receive Descriptor 1 (cont'd)

Bit	Description
24	RCH: Second Address Chained When set, this bit indicates that the second address in the descriptor is the Next Descriptor address rather than the second buffer address. When RDES1[24] is set, RBS2 (RDES1[21-11]) is a “don't care” value. RDES1[25] takes precedence over RDES1[24].
23:22	Reserved
21:11	RBS2: Receive Buffer 2 Size These bits indicate the second data buffer size in bytes. The buffer size must be a multiple of 4/8/16 depending upon the bus widths (32/64/128), even if the value of RDES3 (buffer2 address pointer) is not aligned to bus width. In the case where the buffer size is not a multiple of 4/8/16, the resulting behavior is undefined. This field is not valid if RDES1[24] is set.
10:0	RBS1: Receive Buffer 1 Size Indicates the first data buffer size in bytes. The buffer size must be a multiple of 4/8/16 depending upon the bus widths (32/64/128), even if the value of RDES2 (buffer1 address pointer) is not aligned. In the case where the buffer size is not a multiple of 4/8/16, the resulting behavior is undefined. If this field is 0, the DMA ignores this buffer and uses Buffer 2 or next descriptor depending on the value of RCH (Bit 24).

Receive Descriptor 2 (RDES2)

RDES2 contains the address pointer to the first data buffer in the descriptor.

Note: See [Host Data Buffer Alignment](#) for further detail on buffer address alignment.

Table 35-26 Receive Descriptor 2 (Default Operation)

Bit	Description
31:0	Buffer 1 Address Pointer These bits indicate the physical address of Buffer 1. There are no limitations on the buffer address alignment except for the following condition: The DMA uses the configured value for its address generation when the RDES2 value is used to store the start of frame. Note that the DMA performs a write operation with the RDES2[3/2/1:0] bits as 0 during the transfer of the start of frame but the frame data is shifted as per the actual Buffer address pointer. The DMA ignores RDES2[3/2/1:0] (corresponding to bus width of 128/64/32) if the address pointer is to a buffer where the middle or last part of the frame is stored.

Receive Descriptor 3 (RDES3)

RDES3 contains the address pointer either to the second data buffer in the descriptor or to the next descriptor.

Table 35-27 Receive Descriptor 3

Bit	Description
31:0	<p>Buffer 2 Address Pointer (Next Descriptor Address)</p> <p>These bits indicate the physical address of Buffer 2 when a descriptor ring structure is used. If the Second Address Chained (RDES1[24]) bit is set, this address contains the pointer to the physical memory where the Next Descriptor is present.</p> <p>If RDES1[24] is set, the buffer (Next Descriptor) address pointer must be bus width-aligned (RDES3[3, 2, or 1:0] = 0, corresponding to a bus width of 128, 64, or 32. LSBs are ignored internally.) However, when RDES1[24] is reset, there are no limitations on the RDES3 value, except for the following condition: The DMA uses the configured value for its buffer address generation when the RDES3 value is used to store the start of frame. The DMA ignores RDES3[3, 2, or 1:0] (corresponding to a bus width of 128, 64, or 32) if the address pointer is to a buffer where the middle or last part of the frame is stored.</p>

35.4.1.2 Transmit Descriptor

The descriptor addresses must be aligned to the bus width used (32). [Figure 35-26](#) shows the transmit descriptor format in Little-Endian mode with a 32-bit data bus.

Each descriptor is provided with two buffers, two byte-count buffers, and two address pointers, which enable the adapter port to be compatible with various types of memory-management schemes.

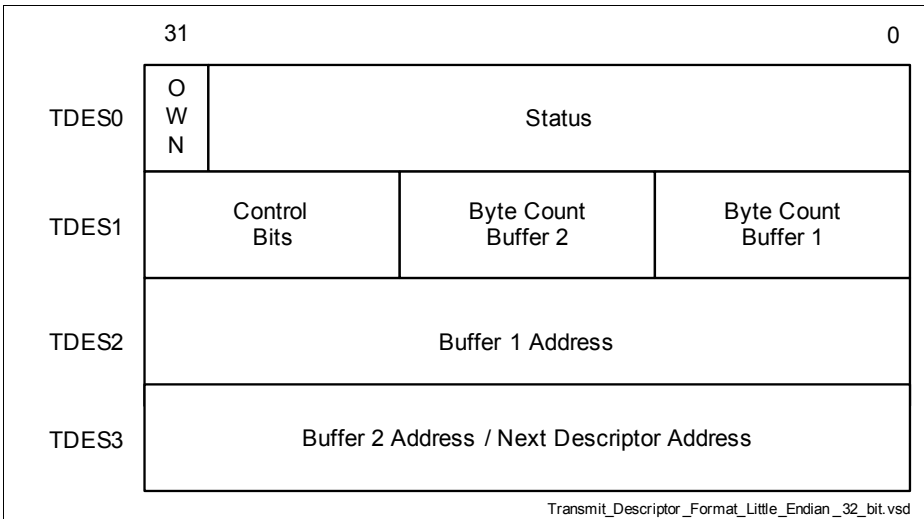


Figure 35-26 Transmit Descriptor Format in Little-Endian Mode With a 32-bit Data Bus

Transmit Descriptor 0 (TDES0)

TDES0 contains the transmitted frame status and the descriptor ownership information.

Table 35-28 Transmit Descriptor 0

Bit	Description
31	<p>OWN: Own Bit</p> <p>When set, this bit indicates that the descriptor is owned by the DMA. When this bit is reset, this bit indicates that the descriptor is owned by the Host. The DMA clears this bit either when it completes the frame transmission or when the buffers allocated in the descriptor are empty. The ownership bit of the First Descriptor of the frame should be set after all subsequent descriptors belonging to the same frame have been set. This avoids a possible race condition between fetching a descriptor and the driver setting an ownership bit.</p>
30:18	Reserved

Table 35-28 Transmit Descriptor 0 (cont'd)

Bit	Description
17	<p>TTSS: Tx Time Stamp Status</p> <p>This status bit indicates that a time stamp has been captured for the corresponding transmit frame. When this bit is set, TDES2 and TDES3 have time stamp values that were captured for the transmit frame. This field is valid only when the Last Segment control bit (TDES1[30]) in a descriptor is set. This bit is valid only when IEEE1588 time stamping feature is enabled; otherwise, it is reserved.</p>
16	<p>IHE: IP Header Error</p> <p>When set, this bit indicates that the Checksum Offload engine detected an IP header error and consequently did not modify the transmitted frame for any checksum insertion. This bit is valid only when Full Checksum Offload is enabled; otherwise, it is reserved.</p>
15	<p>ES: Error Summary</p> <p>Indicates the logical OR of the following bits:</p> <ul style="list-style-type: none"> • TDES0[14]: Jabber Timeout • TDES0[13]: Frame Flush • TDES0[11]: Loss of Carrier • TDES0[10]: No Carrier • TDES0[9]: Late Collision • TDES0[8]: Excessive Collision • TDES0[2]: Excessive Deferral • TDES0[1]: Underflow Error
14	<p>JT: Jabber Timeout</p> <p>When set, this bit indicates the GMAC transmitter has experienced a jabber time-out. This bit is only set when the GMAC configuration register's JD bit is not set.</p>
13	<p>FF: Frame Flushed</p> <p>When set, this bit indicates that the DMA/MTL flushed the frame due to a SW flush command given by the CPU.</p>

Table 35-28 Transmit Descriptor 0 (cont'd)

Bit	Description
12	<p>PCE: Payload Checksum Error</p> <p>This bit, when set, indicates that the Checksum Offload engine had a failure and did not insert any checksum into the encapsulated TCP, UDP, or ICMP payload. This failure can be either due to insufficient bytes, as indicated by the IP Header's Payload Length field, or the MTL starting to forward the frame to the MAC transmitter in Store-and-Forward mode without the checksum having been calculated yet. This second error condition only occurs when the Transmit FIFO depth is less than the length of the Ethernet frame being transmitted: to avoid deadlock, the MTL starts forwarding the frame when the FIFO is full, even in Store-and-Forward mode.</p> <p>When the Full Checksum Offload engine is not enabled during configuration, this bit is reserved.</p>
11	<p>LC: Loss of Carrier</p> <p>When set, this bit indicates that Loss of Carrier occurred during frame transmission (that is, the gmii_crs_i signal was inactive for one or more transmit clock periods during frame transmission). This is valid only for the frames transmitted without collision and when the GMAC operates in Half-Duplex Mode.</p>
10	<p>NC: No Carrier</p> <p>When set, this bit indicates that the carrier sense signal from the PHY was not asserted during transmission.</p>
9	<p>LC: Late Collision</p> <p>When set, this bit indicates that frame transmission was aborted due to a collision occurring after the collision window (64 byte times including Preamble in MII Mode and 512 byte times including Preamble and Carrier Extension in GMII Mode). Not valid if Underflow Error is set.</p>
8	<p>EC: Excessive Collision</p> <p>When set, this bit indicates that the transmission was aborted after 16 successive collisions while attempting to transmit the current frame. If the DR (Disable Retry) bit in the GMAC Configuration Register is set, this bit is set after the first collision and the transmission of the frame is aborted.</p>
7	<p>VF: VLAN Frame</p> <p>When set, this bit indicates that the transmitted frame was a VLAN-type frame.</p>
6:3	<p>CC: Collision Count</p> <p>This 4-bit counter value indicates the number of collisions occurring before the frame was transmitted. The count is not valid when the Excessive Collisions bit (TDES0[8]) is set.</p>

Table 35-28 Transmit Descriptor 0 (cont'd)

Bit	Description
2	<p>ED: Excessive Deferral</p> <p>When set, this bit indicates that the transmission has ended because of excessive deferral of over 24,288 bit times (155,680 bits times in 1000 Mbit/s mode, or in Jumbo Frame enabled mode) if the Deferral Check (DC) bit is set high in the GMAC Control Register.</p>
1	<p>UF: Underflow Error</p> <p>When set, this bit indicates that the GMAC aborted the frame because data arrived late from the Host memory. Underflow Error indicates that the DMA encountered an empty Transmit Buffer while transmitting the frame. The transmission process enters the suspended state and sets both Transmit Underflow (Register 5[5]) and Transmit Interrupt (Register 5[0]).</p>
0	<p>DB: Deferred Bit</p> <p>When set, this bit indicates that the GMAC defers before transmission because of the presence of carrier. This bit is valid only in Half-Duplex mode.</p>

Transmit Descriptor 1 (TDES1)

TDES1 contains the buffer sizes and other bits which control the descriptor chain/ring and the frame being transferred.

Note: See [Buffer Size Calculations](#) for further detail on calculating buffer sizes.

Table 35-29 Transmit Descriptor 1

Bit	Description
31	<p>IC: Interrupt on Completion</p> <p>When set, this bit sets Transmit Interrupt (Register 5[0]) after the present frame has been transmitted.</p>
30	<p>LS: Last Segment</p> <p>When set, this bit indicates that the buffer contains the last segment of the frame.</p>
29	<p>FS: First Segment</p> <p>When set, this bit indicates that the buffer contains the first segment of a frame.</p>

Table 35-29 Transmit Descriptor 1 (cont'd)

Bit	Description
28:27	<p>CIC: Checksum Insertion Control</p> <p>These bits control the insertion of checksums in Ethernet frames that encapsulate TCP, UDP, or ICMP over IPv4 or IPv6 as described below.</p> <ul style="list-style-type: none"> • 2'b00: Do nothing. Checksum Engine is bypassed • 2'b01: Insert IPv4 header checksum. Use this value to insert IPv4 header checksum when the frame encapsulates an IPv4 datagram. • 2'b10: Insert TCP/UDP/ICMP checksum. The checksum is calculated over the TCP, UDP, or ICMP segment only and the TCP, UDP, or ICMP pseudo-header checksum is assumed to be present in the corresponding input frame's Checksum field. An IPv4 header checksum is also inserted if the encapsulated datagram conforms to IPv4. • 2'b11: Insert a TCP/UDP/ICMP checksum that is fully calculated in this engine. In other words, the TCP, UDP, or ICMP pseudo-header is included in the checksum calculation, and the input frame's corresponding Checksum field has an all-zero value. An IPv4 Header checksum is also inserted if the encapsulated datagram conforms to IPv4. <p>The Checksum engine detects whether the TCP, UDP, or ICMP segment is encapsulated in IPv4 or IPv6 and processes its data accordingly.</p>
26	<p>DC: Disable CRC</p> <p>When set, the GMAC does not append the Cyclic Redundancy Check (CRC) to the end of the transmitted frame. This is valid only when the first segment (TDES1[29]).</p>
25	<p>TER: Transmit End of Ring</p> <p>When set, this bit indicates that the descriptor list reached its final descriptor. The returns to the base address of the list, creating a descriptor ring.</p>
24	<p>TCH: Second Address Chained</p> <p>When set, this bit indicates that the second address in the descriptor is the Next Descriptor address rather than the second buffer address. When TDES1[24] is set, TBS2 (TDES1[21–11]) are “don't care” values. TDES1[25] takes precedence over TDES1[24].</p>
23	<p>DP: Disable Padding</p> <p>When set, the GMAC does not automatically add padding to a frame shorter than 64 bytes. When this bit is reset, the DMA automatically adds padding and CRC to a frame shorter than 64 bytes and the CRC field is added despite the state of the DC (TDES1[26]) bit. This is valid only when the first segment (TDES1[29]) is set.</p>

Table 35-29 Transmit Descriptor 1 (cont'd)

Bit	Description
22	TTSE: Transmit Time Stamp Enable When set, this bit enables IEEE1588 hardware time stamping for the transmit frame referenced by the descriptor. This field is valid only when the First Segment control bit (TDES1[29]) is set.
21:11	TBS2: Transmit Buffer 2 Size These bits indicate the Second Data Buffer in bytes. This field is not valid if TDES1[24] is set.
10:0	TBS1: Transmit Buffer 1 Size These bits indicate the First Data Buffer byte size. If this field is 0, the DMA ignores this buffer and uses Buffer 2 or next descriptor depending on the value of TCH (Bit 24).

Transmit Descriptor 2 (TDES2)

TDES2 contains the address pointer to the first buffer of the descriptor.

Table 35-30 Transmit Descriptor 2

Bit	Description
31:0	Buffer 1 Address Pointer These bits indicate the physical address of Buffer 1. There is no limitation on the buffer address alignment. See Host Data Buffer Alignment for further detail on buffer address alignment.

Transmit Descriptor 3 (TDES3)

TDES3 contains the address pointer either to the second buffer of the descriptor or the next descriptor.

Table 35-31 Transmit Descriptor 3

Bit	Description
31:0	Buffer 2 Address Pointer (Next Descriptor Address) Indicates the physical address of Buffer 2 when a descriptor ring structure is used. If the Second Address Chained (TDES1[24]) bit is set, this address contains the pointer to the physical memory where the Next Descriptor is present. The buffer address pointer must be aligned to the bus width only when TDES1[24] is set. (LSBs are ignored internally.)

35.4.1.3 Descriptor Format With IEEE 1588 Time Stamping Enabled

The default descriptor format (as described in **“Receive Descriptor” on Page 1-677** and **“Transmit Descriptor” on Page 1-669**), and field descriptions remain unchanged when created by software (Own bit is set in DES0). However, if the software has enabled IEEE 1588 functionality, the DES2 and DES3 descriptor fields (see **Figure 35-27**) take on a different meaning when the DMA closes the descriptor (own bit in DES0 is cleared). The DMA updates the DES2 and DES3 with the time stamp value before clearing the Own bit in DES0.

When the core is operating in 32-bit mode (**Figure 35-24**), DES2 is updated with the lower 32 time stamp bits (the Sub-Second field, called TSL in subsequent sections) and DES3 is updated with the upper 32 time stamp bits (the Seconds field, called TSH in subsequent sections).

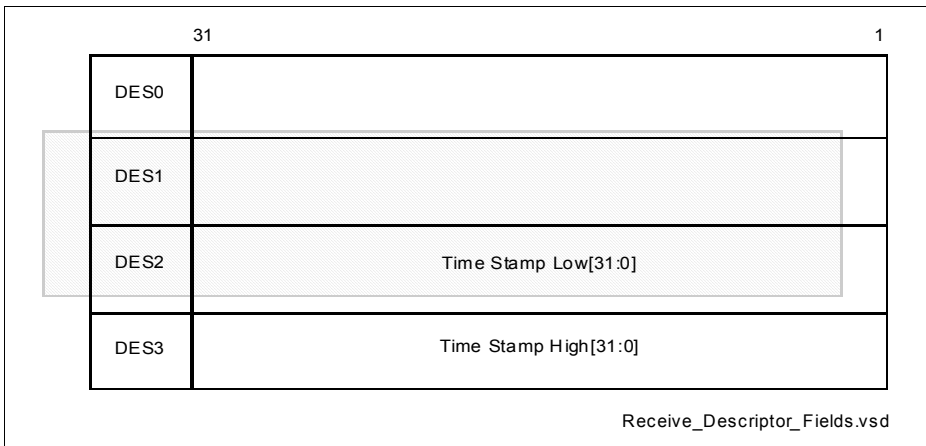


Figure 35-27 Receive Descriptor Fields When DMA Clears the Own Bit

The following sections describe the details specific to receive and transmit descriptors in this mode.

Receive Descriptor

Receive Time Stamp

The tables below describe the fields that have different meaning for RDES2 and RDES3 when the receive descriptor is closed and time stamping is enabled.

Note: When software disables the time stamping feature (the TSENA bit in Register 448 is low), the DMA does not update the descriptor's RDES2/RDES3 fields before closing the RDES0.

Table 35-32 Receive Descriptor Fields (RDES2)

Bit	Description
31:0	<p>RTSL: Receive Frame Time Stamp Low</p> <p>The DMA updates this field with the least significant 32 bits of the time stamp captured for the corresponding receive frame. The DMA updates this field only for the last descriptor of the receive frame indicated by Last Descriptor status bit (RDES0[8]). When this field and the RTSH field in RDES3 show an all-ones value, the time stamp must be treated as corrupt.</p>

Table 35-33 Receive Descriptor Fields (RDES3)

Bit	Description
31:0	<p>RTSH: Receive Frame Time Stamp High</p> <p>The DMA updates this field with the most significant 32 bits of the time stamp captured for the corresponding receive frame. The DMA updates this field only for the last descriptor of the receive frame indicated by Last Descriptor status bit (RDES0[8]).</p> <p>When this field and RDES2's RTSL field show all-ones values, the time stamp must be treated as corrupt.</p>

Transmit Descriptor

In addition to the changes described in [“Descriptor Format With IEEE 1588 Time Stamping Enabled” on Page 1-676](#), the Transmit descriptor has additional control and status bits (TTSE and TTSS, respectively) for time stamping, as shown in [Figure 35-28](#). Software sets the TTSE bit (when the Own bit is set), instructing the core to generate a time stamp for the corresponding Ethernet frame being transmitted. The DMA sets the TTSS bit if the time stamp has been updated in the TDES2 and TDES3 fields when the descriptor is closed (Own bit is cleared).

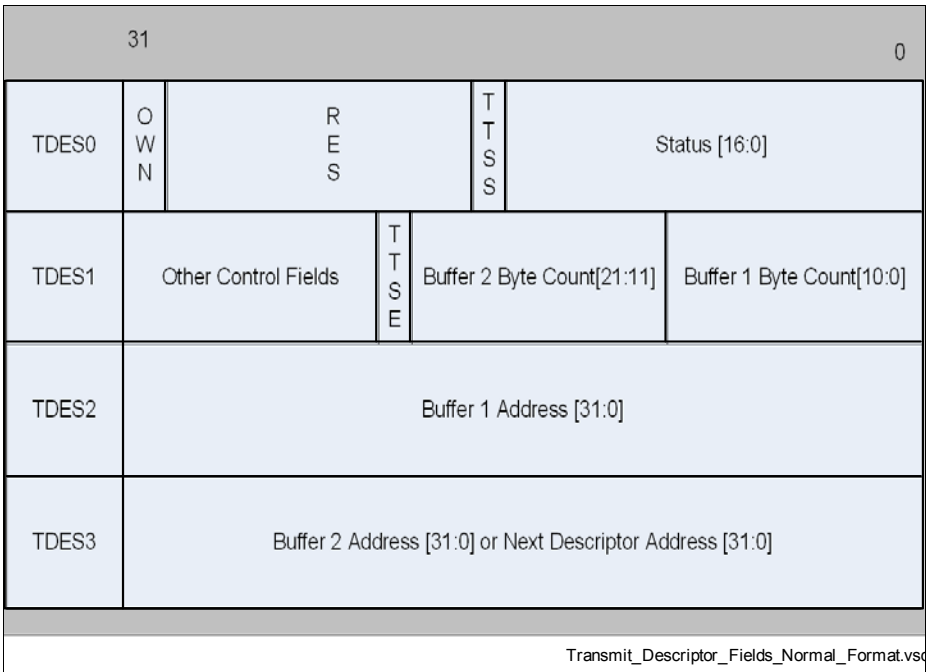


Figure 35-28 Transmit Descriptor Fields– Normal Format

Transmit Time Stamp Control and Status Fields

The position of these fields is different for normal transmit descriptor and enhanced format transmit descriptor. The value of this field in both the cases shall be preserved by the DMA at the time of closing the descriptor.

Updates to [Table 1-87](#) and [Table 1-88](#) for the default (normal) descriptor format are described below.

Table 35-34 Transmit Time Stamp Status – Normal Descriptor Format Case (TDES0)

Bit	Description
17	<p>TTSS: Transmit Time Stamp Status</p> <p>This field is a status bit indicating that a time stamp was captured for the corresponding transmit frame. When this bit is set, both TDES2 and TDES3 have a time stamp value that was captured for the transmit frame. This field is valid only when the Last Segment control bit (TDES1[30] in the descriptor) is set.</p>

Table 35-35 Transmit Time Stamp Control – Normal Descriptor Format Case (TDES1)

Bit	Description
22	<p>TTSE: Transmit Time Stamp Enable</p> <p>When set, this field enables IEEE1588 hardware time stamping for the transmit frame described by the descriptor.</p> <p>This field is valid only when the First Segment control bit (TDES1[29] in the descriptor) is set.</p>

Transmit Time Stamp Field

The transmit descriptor format and field descriptions remain unchanged when they are created by software (when the Own bit is set). However, when the DMA closes the last descriptor (marked, in the alternative descriptor format, by the LS bit in TDES1 or TDES0) and IEEE 1588 functionality is enabled (the Own bit is cleared), the TDES2 and TDES3 descriptor fields are updated with the time stamp, if taken, for that frame.

The fields in TDES2 and TDES3 are swapped when the core operates in 64- or 128-bit and with the descriptor in Reverse-Endian mode

[Table 1-96](#) and [Table 1-97](#) describe the fields that have different meaning when the descriptor is closed.

Table 35-36 Transmit Descriptor Fields (TDES2)

Bit	Description
31:0	<p>TTSL: Transmit Frame Time Stamp Low</p> <p>This field is updated by DMA with the least significant 32 bits of the time stamp captured for the corresponding transmit frame. This field has the time stamp only if the Last Segment control bit (LS) in the descriptor is set.</p>

Table 35-37 Transmit Descriptor Fields (TDES3)

Bit	Description
31:0	<p>TTSH: Transmit Frame Time Stamp High</p> <p>This field is updated by DMA with the most significant 32 bits of the time stamp captured for the corresponding transmit frame. This field has the time stamp only if the Last Segment control bit (LS) in the descriptor is set.</p>

35.4.2 Alternate or Enhanced Descriptors

The alternate (or enhanced) descriptor structure can have 8 DWORDS (32-bytes) instead of the 4 DWORDS as in the case of normal descriptor format. The features of the alternate descriptor structure are

- The normal descriptor structure allows data buffers of up to 2.048 bytes. The alternative descriptor structure has been implemented to support buffers of up to 8 KB (useful for Jumbo frames).
- There is a re-assignment of control and status bits in TDES0, TDES1, RDES0 (Advanced time stamp or IPC full offload configuration), RDES1.
- The transmit descriptor stores the time stamp in TDES6 and TDES7 when advanced time stamp feature is selected.
- This receive descriptor structure is also used for storing the extended status (RDES4) and time stamp (RDES6 and RDES7) when advanced time stamp feature or IPC full offload is selected.
- When alternate descriptor mode is selected, and Time-stamping feature is enabled, the software needs to allocate 32-bytes (8 DWORDS) of memory for every descriptor. When Time-stamping or Receive IPC FullOffload engine are not enabled, the extended descriptors are not required and the SW can use alternate descriptors with the default size of 16 bytes. The core also needs to be configured for this change using the DMA Bus Mode Register[7].
- When alternate descriptor is chosen without Time Stamp or Full IPC Offload feature, the descriptor size is always 4 DWORDS (DES0-DES3).

The description or bit-mapping alternate descriptor structure (in Little Endian mode) is given below.

*Note: The effect of Big Endian mode (byte-swap) explained in **“Normal Descriptor Formats” on Page 1-658** apply to this descriptor structure as well.*

When alternate descriptor with only Full IPC Offload (Type 2) is selected, it is not backward compatible to the previous release 3.4x with respect to status bits[7,5,0] in RDES0. In this mode, you should enable the extended descriptor mode (8 DWORDS) to get the IPC checksum engine status in RDES4.

35.4.2.1 Transmit Descriptor

The transmit descriptor structure is shown in **Figure 35-29**. The application software must program the control bits TDES0[31:20] during descriptor initialization. When the DMA updates the descriptor, it write backs all the control bits except the OWN bit (which it clears) and updates the status bits[19:0]. The contents of the transmitter descriptor word 0 (TDES0) through word 3 (TDES3) are given in **Table 1-98** through **Table 1-101**, respectively.

With the advance time stamp support, the snapshot of the time stamp to be taken can be enabled for a given frame by setting the “TTSE: Transmit Time Stamp Enable” (bit-25 of TDES0). When the descriptor is closed (i.e. when the OWN bit is cleared), the time-

Ethernet MAC (ETH)

stamp is written into TDES6 and TDES7. This is indicated by the status bit “TTSS: Transmit Time Stamp Status” (bit-17 of TDES0). This is shown in **Figure 35-29**. The contents of TDES6 and TDES7 are mentioned in **Table 1-102** and **Table 1-103**.

Note: When either of Advanced Time Stamp or IPC Offload (Type 2) features is enabled, the SW should set the DMA Bus Mode register[7], so that the DMA operates with extended descriptor size. When this control bit is reset, the TDES4-TDES7 descriptor space are not valid.

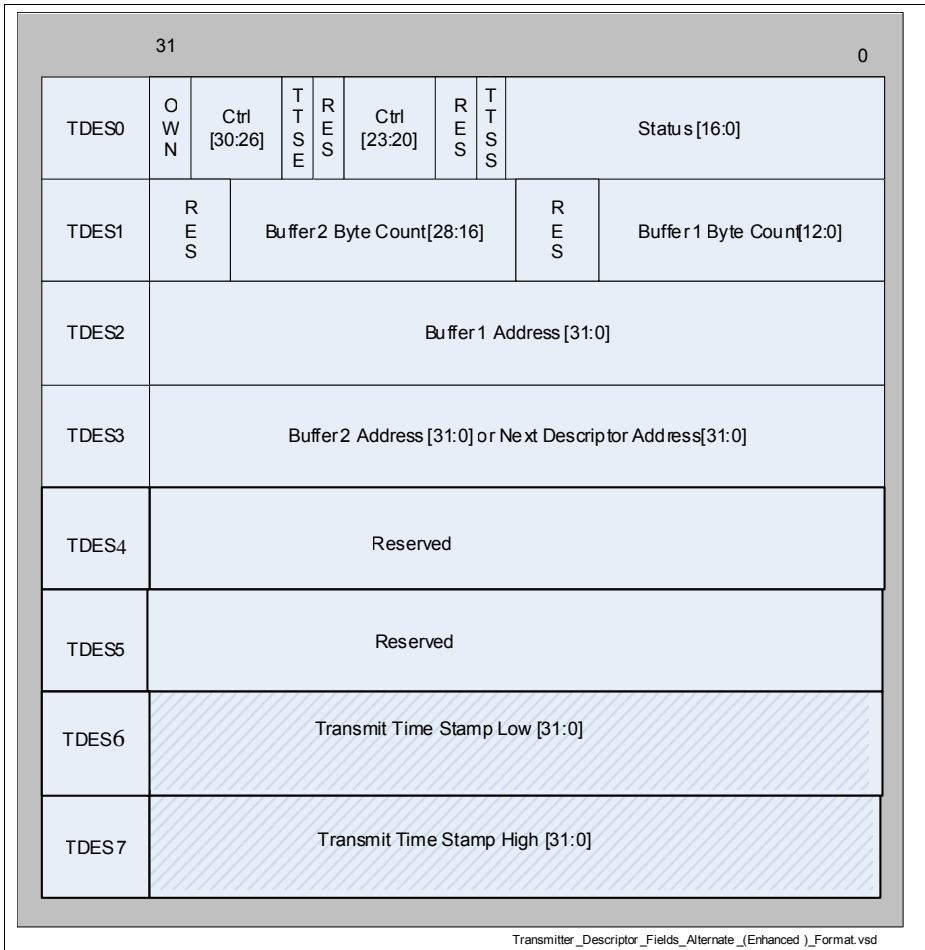


Figure 35-29 Transmitter Descriptor Fields - Alternate (Enhanced) Format

Table 35-38 Transmit Descriptor Word 0 (TDES0)

Bit	Description
31	<p>OWN: Own Bit</p> <p>When set, this bit indicates that the descriptor is owned by the DMA. When this bit is reset, it indicates that the descriptor is owned by the Host. The DMA clears this bit either when it completes the frame transmission or when the buffers allocated in the descriptor are read completely. The ownership bit of the frame's first descriptor must be set after all subsequent descriptors belonging to the same frame have been set. This avoids a possible race condition between fetching a descriptor and the driver setting an ownership bit.</p>
30	<p>IC: Interrupt on Completion</p> <p>When set, this bit sets the Transmit Interrupt (Register 5[0]) after the present frame has been transmitted.</p>
29	<p>LS: Last Segment</p> <p>When set, this bit indicates that the buffer contains the last segment of the frame.</p>
28	<p>FS: First Segment</p> <p>When set, this bit indicates that the buffer contains the first segment of a frame.</p>
27	<p>DC: Disable CRC</p> <p>When this bit is set, the GMAC does not append a cyclic redundancy check (CRC) to the end of the transmitted frame. This is valid only when the first segment (TDES0[28]) is set.</p>
26	<p>DP: Disable Pad</p> <p>When set, the GMAC does not automatically add padding to a frame shorter than 64 bytes. When this bit is reset, the DMA automatically adds padding and CRC to a frame shorter than 64 bytes, and the CRC field is added despite the state of the DC (TDES0[27]) bit. This is valid only when the first segment (TDES0[28]) is set.</p>
25	<p>TTSE: Transmit Time Stamp Enable</p> <p>When set, this bit enables IEEE1588 hardware time stamping for the transmit frame referenced by the descriptor. This field is valid only when the First Segment control bit (TDES0[28]) is set.</p>
24	<p>Reserved</p>

Table 35-38 Transmit Descriptor Word 0 (TDES0) (cont'd)

Bit	Description
23:22	<p>CIC: Checksum Insertion Control</p> <p>These bits control the checksum calculation and insertion. Bit encodings are as shown below.</p> <ul style="list-style-type: none"> • 2'b00: Checksum Insertion Disabled. • 2'b01: Only IP header checksum calculation and insertion are enabled. • 2'b10: IP header checksum and payload checksum calculation and insertion are enabled, but pseudo-header checksum is not calculated in hardware. • 2'b11: IP Header checksum and payload checksum calculation and insertion are enabled, and pseudo-header checksum is calculated in hardware. <p>This field is reserved when the IPC_FULL_OFFLOAD configuration parameter is not selected.</p>
21	<p>TER: Transmit End of Ring</p> <p>When set, this bit indicates that the descriptor list reached its final descriptor. The DMA returns to the base address of the list, creating a descriptor ring.</p>
20	<p>TCH: Second Address Chained</p> <p>When set, this bit indicates that the second address in the descriptor is the Next Descriptor address rather than the second buffer address. When TDES0[20] is set, TBS2 (TDES1[28:16]) is a “don't care” value. TDES0[21] takes precedence over TDES0[20].</p>
19:18	<p>Reserved</p>
17	<p>TTSS: Transmit Time Stamp Status</p> <p>This field is used as a status bit to indicate that a time stamp was captured for the described transmit frame. When this bit is set, TDES2 and TDES3 have a time stamp value captured for the transmit frame. This field is only valid when the descriptor's Last Segment control bit (TDES0[29]) is set.</p>
16	<p>IHE: IP Header Error</p> <p>When set, this bit indicates that the GMAC transmitter detected an error in the IP datagram header. The transmitter checks the header length in the IPv4 packet against the number of header bytes received from the application and indicates an error status if there is a mismatch. For IPv6 frames, a header error is reported if the main header length is not 40 bytes. Furthermore, the Ethernet Length/Type field value for an IPv4 or IPv6 frame must match the IP header version received with the packet. For IPv4 frames, an error status is also indicated if the Header Length field has a value less than 0x5.</p>

Table 35-38 Transmit Descriptor Word 0 (TDES0) (cont'd)

Bit	Description
15	<p>ES: Error Summary Indicates the logical OR of the following bits:</p> <ul style="list-style-type: none"> • TDES0[14]: Jabber Timeout • TDES0[13]: Frame Flush • TDES0[11]: Loss of Carrier • TDES0[10]: No Carrier • TDES0[9]: Late Collision • TDES0[8]: Excessive Collision • TDES0[2]: Excessive Deferral • TDES0[1]: Underflow Error • TDES0[16]: IP Header Error • TDES0[12]: IP Payload Error
14	<p>JT: Jabber Timeout When set, this bit indicates the GMAC transmitter has experienced a jabber time-out. This bit is only set when the GMAC configuration register's JD bit is not set.</p>
13	<p>FF: Frame Flushed When set, this bit indicates that the DMA/MTL flushed the frame due to a software Flush command given by the CPU.</p>
12	<p>IPE: IP Payload Error When set, this bit indicates that GMAC transmitter detected an error in the TCP, UDP, or ICMP IP datagram payload. The transmitter checks the payload length received in the IPv4 or IPv6 header against the actual number of TCP, UDP, or ICMP packet bytes received from the application and issues an error status in case of a mismatch.</p>
11	<p>LC: Loss of Carrier When set, this bit indicates that a loss of carrier occurred during frame transmission (that is, the gmii_crs_i signal was inactive for one or more transmit clock periods during frame transmission). This is valid only for the frames transmitted without collision when the GMAC operates in Half-Duplex mode.</p>
10	<p>NC: No Carrier When set, this bit indicates that the Carrier Sense signal from the PHY was not asserted during transmission.</p>
9	<p>LC: Late Collision When set, this bit indicates that frame transmission was aborted due to a collision occurring after the collision window (64 byte-times, including preamble, in MII mode and 512 byte-times, including preamble and carrier extension, in GMII mode). This bit is not valid if the Underflow Error bit is set.</p>

Table 35-38 Transmit Descriptor Word 0 (TDES0) (cont'd)

Bit	Description
8	EC: Excessive Collision When set, this bit indicates that the transmission was aborted after 16 successive collisions while attempting to transmit the current frame. If the DR (Disable Retry) bit in the GMAC Configuration register is set, this bit is set after the first collision, and the transmission of the frame is aborted.
7	VF: VLAN Frame When set, this bit indicates that the transmitted frame was a VLAN-type frame.
6:3	CC: Collision Count This 4-bit counter value indicates the number of collisions occurring before the frame was transmitted. The count is not valid when the Excessive Collisions bit (TDES0[8]) is set.
2	ED: Excessive Deferral When set, this bit indicates that the transmission has ended because of excessive deferral of over 24,288 bit times (155,680 bits times in 1,000-Mbit/s mode or if Jumbo Frame is enabled) if the Deferral Check (DC) bit in the GMAC Control register is set high.
1	UF: Underflow Error When set, this bit indicates that the GMAC aborted the frame because data arrived late from the Host memory. Underflow Error indicates that the DMA encountered an empty transmit buffer while transmitting the frame. The transmission process enters the Suspended state and sets both Transmit Underflow (Register 5[5]) and Transmit Interrupt (Register 5[0]).
0	DB: Deferred Bit When set, this bit indicates that the GMAC defers before transmission because of the presence of carrier. This bit is valid only in Half-Duplex mode.

Table 35-39 Transmit Descriptor Word 1 (TDES1)

Bit	Description
31:29	Reserved
28:16	TBS2: Transmit Buffer 2 Size These bits indicate the second data buffer size in bytes. This field is not valid if TDES0[20] is set. See Buffer Size Calculations for further detail on calculating buffer sizes.

Table 35-39 Transmit Descriptor Word 1 (TDES1) (cont'd)

Bit	Description
15:13	Reserved
12:0	TBS1: Transmit Buffer 1 Size These bits indicate the first data buffer byte size, in bytes. If this field is 0, the DMA ignores this buffer and uses Buffer 2 or the next descriptor, depending on the value of TCH (TDES0[20]).

Table 35-40 Transmit Descriptor 2 (TDES2)

Bit	Description
31:0	Buffer 1 Address Pointer These bits indicate the physical address of Buffer 1. There is no limitation on the buffer address alignment. See Host Data Buffer Alignment for further detail on buffer address alignment.

Table 35-41 Transmit Descriptor 3 (TDES3)

Bit	Description
31:0	Buffer 2 Address Pointer (Next Descriptor Address) Indicates the physical address of Buffer 2 when a descriptor ring structure is used. If the Second Address Chained (TDES1[24]) bit is set, this address contains the pointer to the physical memory where the Next Descriptor is present. The buffer address pointer must be aligned to the bus width only when TDES1[24] is set. (LSBs are ignored internally.)

Table 35-42 Transmit Descriptor 6 (TDES6)

Bit	Description
31:0	TTSL: Transmit Frame Time Stamp Low This field is updated by DMA with the least significant 32 bits of the time stamp captured for the corresponding transmit frame. This field has the time stamp only if the Last Segment bit (LS) in the descriptor is set and Time stamp status (TTSS) bit is set.

Table 35-43 Transmit Descriptor 7 (TDES7)

Bit	Description
31:0	<p>TTSH: Transmit Frame Time Stamp High</p> <p>This field is updated by DMA with the most significant 32 bits of the time stamp captured for the corresponding receive frame. This field has the time stamp only if the Last Segment bit (LS) in the descriptor is set and Time stamp status (TTSS) bit is set.</p>

35.4.2.2 Receive Descriptor

The structure of the received descriptor is shown in [Figure 35-30](#). This can have 32 bytes of descriptor data (8 DWORDs) when Advanced Time Stamping or IPC Full Offload feature is selected.

Note: When either of these features is enabled, the SW should set the DMA Bus Mode register[7] so that the DMA operates with extended descriptor size. When this control bit is reset, RDES0[7] and RDES0[0] will be always cleared and the RDES4-RDES7 descriptor space are not valid

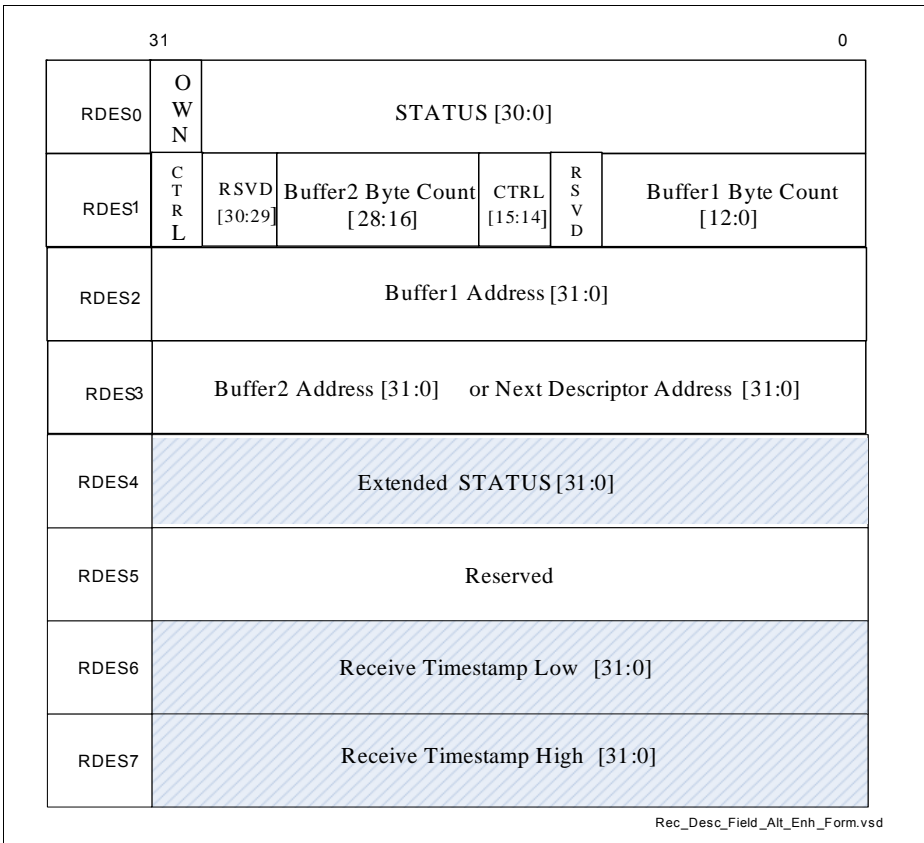


Figure 35-30 Receive Descriptor Fields - Alternate (Enhanced) Format

The contents of RDES0 are identified in [Table 1-104](#). The contents of RDES1 through RDES3 are identified in [Table 1-105](#) through [Table 1-107](#), respectively.

Note: Some of the bit functions of RDES0 are not backward compatible to Release 3.41a and previous versions. These bits are Bit[7], Bit[0] and Bit[5]. The function of Bit[5] is backward compatible to Release 3.30a and previous versions.

Table 35-44 Receive Descriptor Fields (RDES0)

Bit	Description
31	<p>OWN: Own Bit</p> <p>When set, this bit indicates that the descriptor is owned by the DMA of the GMAC Subsystem. When this bit is reset, this bit indicates that the descriptor is owned by the Host. The DMA clears this bit either when it completes the frame reception or when the buffers that are associated with this descriptor are full.</p>
30	<p>AFM: Destination Address Filter Fail</p> <p>When set, this bit indicates a frame that failed in the DA Filter in the GMAC Core.</p>
29:16	<p>FL: Frame Length</p> <p>These bits indicate the byte length of the received frame that was transferred to host memory (including CRC). This field is valid when Last Descriptor (RDES0[8]) is set and either the Descriptor Error (RDES0[14]) or Overflow Error bits are reset. The frame length also includes the two bytes appended to the Ethernet frame when IP checksum calculation (Type 1) is enabled and the received frame is not a MAC control frame.</p> <p>This field is valid when Last Descriptor (RDES0[8]) is set. When the Last Descriptor and Error Summary bits are not set, this field indicates the accumulated number of bytes that have been transferred for the current frame.</p>
15	<p>ES: Error Summary</p> <p>Indicates the logical OR of the following bits:</p> <ul style="list-style-type: none"> • RDES0[1]: CRC Error • RDES0[3]: Receive Error • RDES0[4]: Watchdog Timeout • RDES0[6]: Late Collision • RDES0[7]: Giant Frame • RDES4[4:3]: IP Header/Payload Error • RDES0[11]: Overflow Error • RDES0[14]: Descriptor Error <p>This field is valid only when the Last Descriptor (RDES0[8]) is set.</p>
14	<p>DE: Descriptor Error</p> <p>When set, this bit indicates a frame truncation caused by a frame that does not fit within the current descriptor buffers, and that the DMA does not own the Next Descriptor. The frame is truncated. This field is valid only when the Last Descriptor (RDES0[8]) is set.</p>
13	<p>SAF: Source Address Filter Fail</p> <p>When set, this bit indicates that the SA field of frame failed the SA Filter in the GMAC Core.</p>

Table 35-44 Receive Descriptor Fields (RDES0) (cont'd)

Bit	Description
12	<p>LE: Length Error</p> <p>When set, this bit indicates that the actual length of the frame received and that the Length/ Type field does not match. This bit is valid only when the Frame Type (RDES0[5]) bit is reset.</p>
11	<p>OE: Overflow Error</p> <p>When set, this bit indicates that the received frame was damaged due to buffer overflow in MTL.</p>
10	<p>VLAN: VLAN Tag</p> <p>When set, this bit indicates that the frame pointed to by this descriptor is a VLAN frame tagged by the GMAC Core.</p>
9	<p>FS: First Descriptor</p> <p>When set, this bit indicates that this descriptor contains the first buffer of the frame. If the size of the first buffer is 0, the second buffer contains the beginning of the frame. If the size of the second buffer is also 0, the next Descriptor contains the beginning of the frame.</p>
8	<p>LS: Last Descriptor</p> <p>When set, this bit indicates that the buffers pointed to by this descriptor are the last buffers of the frame</p>
7	<p>Time Stamp Available/IP Checksum Error (Type1) /Giant Frame</p> <p>When Advanced Time Stamp feature is present, this bit, when set, indicates that a snapshot of the timestamp is written in descriptor words 6 (RDES6) and 7 (RDES7). This is valid only when the Last Descriptor bit (RDES0[8]) is set. When IP Checksum Engine (Type 1) is selected, this bit, when set, indicates that the 16-bit IPv4 Header checksum calculated by the core did not match the received checksum bytes.</p> <p>Otherwise, this bit, when set, indicates the Giant Frame Status. Giant frames are larger-than-1,518-byte (or 1,522-byte for VLAN) normal frames and larger-than-9,018-byte (9,022-byte for VLAN) frame when Jumbo Frame processing is enabled.</p>
6	<p>LC: Late Collision</p> <p>When set, this bit indicates that a late collision has occurred while receiving the frame in Half-Duplex mode.</p>
5	<p>FT: Frame Type When set, this bit indicates that the Receive Frame is an Ethernet-type frame (the LT field is greater than or equal to 16'h0600). When this bit is reset, it indicates that the received frame is an IEEE802.3 frame. This bit is not valid for Runt frames less than 14 bytes.</p>

Table 35-44 Receive Descriptor Fields (RDES0) (cont'd)

Bit	Description
4	RWT: Receive Watchdog Timeout When set, this bit indicates that the Receive Watchdog Timer has expired while receiving the current frame and the current frame is truncated after the Watchdog Timeout.
3	RE: Receive Error When set, this bit indicates that the gmii_rxr_i signal is asserted while gmii_rxdv_i is asserted during frame reception. This error also includes carrier extension error in GMII and Half-duplex mode. Error can be of less/no extension, or error (rxd ≠ 0f) during extension.
2	DE: Dribble Bit Error When set, this bit indicates that the received frame has a non-integer multiple of bytes (odd nibbles). This bit is valid only in MII Mode.
1	CE: CRC Error When set, this bit indicates that a Cyclic Redundancy Check (CRC) Error occurred on the received frame. This field is valid only when the Last Descriptor (RDES0[8]) is set.
0	Extended Status Available/Rx MAC Address When either Advanced Time Stamp or IP Checksum Offload (Type 2) is present, this bit, when set, indicates that the extended status is available in descriptor word 4 (RDES4). This is valid only when the Last Descriptor bit (RDES0[8]) is set. When Advance Time Stamp Feature or IPC Full Offload is not selected, this bit indicates Rx MAC Address status. When set, this bit indicates that the Rx MAC Address registers value (1 to 31) matched the frame's DA field. When reset, this bit indicates that the Rx MAC Address Register 0 value matched the DA field.

Table 35-45 Receive Descriptor Fields 1 (RDES1)

Bit	Description
31	DIC: Disable Interrupt on Completion When set, this bit prevents setting the Status Register's RI bit (CSR5[6]) for the received frame ending in the buffer indicated by this descriptor. This, in turn, disables the assertion of the interrupt to Host due to RI for that frame.
30:29	Reserved

Table 35-45 Receive Descriptor Fields 1 (RDES1) (cont'd)

Bit	Description
28:16	<p>RBS2: Receive Buffer 2 Size</p> <p>These bits indicate the second data buffer size, in bytes. The buffer size must be a multiple of 4, 8, or 16, depending on the bus widths (32, 64, or 128, respectively), even if the value of RDES3 (buffer2 address pointer) is not aligned to bus width. If the buffer size is not an appropriate multiple of 4, 8, or 16, the resulting behavior is undefined. This field is not valid if RDES1[14] is set. See Buffer Size Calculations for further details on calculating buffer sizes.</p>
15	<p>RER: Receive End of Ring</p> <p>When set, this bit indicates that the descriptor list reached its final descriptor. The DMA returns to the base address of the list, creating a descriptor ring.</p>
14	<p>RCH: Second Address Chained</p> <p>When set, this bit indicates that the second address in the descriptor is the Next Descriptor address rather than the second buffer address. When this bit is cleared, RBS2 (RDES1[28:16]) is a “don’t care” value. RDES1[15] takes precedence over RDES1[14].</p>
13	<p>Reserved</p>
12:0	<p>RBS1: Receive Buffer 1 Size</p> <p>Indicates the first data buffer size in bytes. The buffer size must be a multiple of 4, 8, or 16, depending upon the bus widths (32, 64, or 128), even if the value of RDES2 (buffer1 address pointer) is not aligned. When the buffer size is not a multiple of 4, 8, or 16, the resulting behavior is undefined. If this field is 0, the DMA ignores this buffer and uses Buffer 2 or next descriptor depending on the value of RCH (Bit 14). See Buffer Size Calculations for further details on calculating buffer sizes.</p>

Table 35-46 Receive Descriptor Fields 2 (RDES2)

Bit	Description
31:0	<p>Buffer 1 Address Pointer</p> <p>These bits indicate the physical address of Buffer 1. There are no limitations on the buffer address alignment except for the following condition: The DMA uses the configured value for its address generation when the RDES2 value is used to store the start of frame. Note that the DMA performs a write operation with the RDES2[3/2/1:0] bits as 0 during the transfer of the start of frame but the frame data is shifted as per the actual Buffer address pointer. The DMA ignores RDES2[3/2/1:0] (corresponding to bus width of 128/64/32) if the address pointer is to a buffer where the middle or last part of the frame is stored. See Host Data Buffer Alignment for further details on buffer address alignment.</p>

Table 35-47 Receive Descriptor Fields 3 (RDES3)

Bit	Description
31:0	<p>Buffer 2 Address Pointer (Next Descriptor Address)</p> <p>These bits indicate the physical address of Buffer 2 when a descriptor ring structure is used. If the Second Address Chained (RDES1[24]) bit is set, this address contains the pointer to the physical memory where the Next Descriptor is present.</p> <p>If RDES1[24] is set, the buffer (Next Descriptor) address pointer must be bus width-aligned (RDES3[3, 2, or 1:0] = 0, corresponding to a bus width of 128, 64, or 32. LSBs are ignored internally.) However, when RDES1[24] is reset, there are no limitations on the RDES3 value, except for the following condition: The DMA uses the configured value for its buffer address generation when the RDES3 value is used to store the start of frame. The DMA ignores RDES3[3, 2, or 1:0] (corresponding to a bus width of 128, 64, or 32) if the address pointer is to a buffer where the middle or last part of the frame is stored.</p>

The extended status written is as shown in [Table 1-108](#). The extended status is written only when there is status related to IPC or Time Stamp available. The availability of extended status is indicated by bit-0 of RDES0. This status is available only when Advance Time Stamp or IPC Full Offload feature is selected.

Table 35-48 Receive Descriptor Fields 4 (RDES4)

Bit	Description
31:14	Reserved
13	<p>PTP Version</p> <p>When set, indicates that the received PTP message is having the IEEE 1588 version 2 format. When reset, it has the version 1 format. This is valid only if the message type is non-zero. This bit is available only if Advance Time Stamp feature is selected else it is reserved</p>
12	<p>PTP Frame Type</p> <p>When set, this bit that the PTP message is sent directly over Ethernet. When this bit is not set and the message type is non-zero, it indicates that the PTP message is sent over UDP-IPv4 or UDP-IPv6. The information on IPv4 or IPv6 can be obtained from bits 6 and 7. This bit is available only if Advanced Time Stamp feature is selected</p>

Table 35-48 Receive Descriptor Fields 4 (RDES4) (cont'd)

Bit	Description
11:8	<p>Message Type</p> <p>These bits are encoded to give the type of the message received.</p> <ul style="list-style-type: none"> • 0000: No PTP message received • 0001: SYNC (all clock types) • 0010: Follow_Up (all clock types) • 0011: Delay_Req (all clock types) • 0100: Delay_Resp (all clock types) • 0101: Pdelay_Req (in peer-to-peer transparent clock) • 0110: Pdelay_Resp (in peer-to-peer transparent clock) • 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock) • 1000: Announce • 1001: Management • 1010: Signaling • 1011-1111: Reserved <p>These bits are valid only when you select the Advance Time Stamp feature.</p>
7	<p>IPv6 Packet Received</p> <p>When set, this bit indicates that the received packet is an IPv6 packet.</p>
6	<p>IPv4 Packet Received</p> <p>When set, this bit indicates that the received packet is an IPv4 packet.</p>
5	<p>IP Checksum Bypassed</p> <p>When set, this bit indicates that the checksum offload engine is bypassed.</p>
4	<p>IP Payload Error</p> <p>When set, this bit indicates that the 16-bit IP payload checksum (that is, the TCP, UDP, or ICMP checksum) that the core calculated does not match the corresponding checksum field in the received segment. It is also set when the TCP, UDP, or ICMP segment length does not match the payload length value in the IP Header field.</p>

Table 35-48 Receive Descriptor Fields 4 (RDES4) (cont'd)

Bit	Description
3	<p>IP Header Error</p> <p>When set, this bit indicates either that the 16-bit IPv4 header checksum calculated by the core does not match the received checksum bytes, or that the IP datagram version is not consistent with the Ethernet Type value.</p>
2:0	<p>IP Payload Type</p> <p>These bits indicate the type of payload encapsulated in the IP datagram processed by the Receive Checksum Offload Engine (COE). The COE also sets these bits to 2'b00 if it does not process the IP datagram's payload due to an IP header error or fragmented IP.</p> <ul style="list-style-type: none"> • 3'b000: Unknown or did not process IP payload • 3'b001: UDP • 3'b010: TCP • 3'b011: ICMP • 3'b1xx: Reserved

RDES6 and RDES7 contain the snapshot of the time-stamp. The availability of the snapshot of the time-stamp in RDES6 and RDES7 is indicated by bit-7 in the RDES0 descriptor. The contents of RDES6 and RDES7 are identified in [Table 1-109](#) and [Table 1-110](#), respectively.

Table 35-49 Receive Descriptor Fields 6 (RDES6)

Bit	Description
31:0	<p>RTSL: Receive Frame Time Stamp Low</p> <p>This field is updated by DMA with the least significant 32 bits of the time stamp captured for the corresponding receive frame. This field is updated by DMA only for the last descriptor of the receive frame which is indicated by Last Descriptor status bit (RDES0[8]).</p>

Table 35-50 Receive Descriptor Fields 7 (RDES7)

Bit	Description
31:0	<p>RTSH: Receive Frame Time Stamp High</p> <p>This field is updated by DMA with the most significant 32 bits of the time stamp captured for the corresponding receive frame. This field is updated by DMA only for the last descriptor of the receive frame which is indicated by Last Descriptor status bit (RDES0[8]).</p>

35.5 Ethernet MAC Module Implementation

This section describes the Ethernet MAC module interface as it is implemented in the TC26x B-Step. It especially covers clock control, port and on-chip connections, interrupt control, and address decoding.

35.5.1 Interface Connections of the Ethernet MAC Module

The Ethernet MAC module is supplied with a separate clock control, address decoding, and interrupt control logic. The modules' service request outputs are connected with one interrupt node.

Ethernet MAC (ETH)

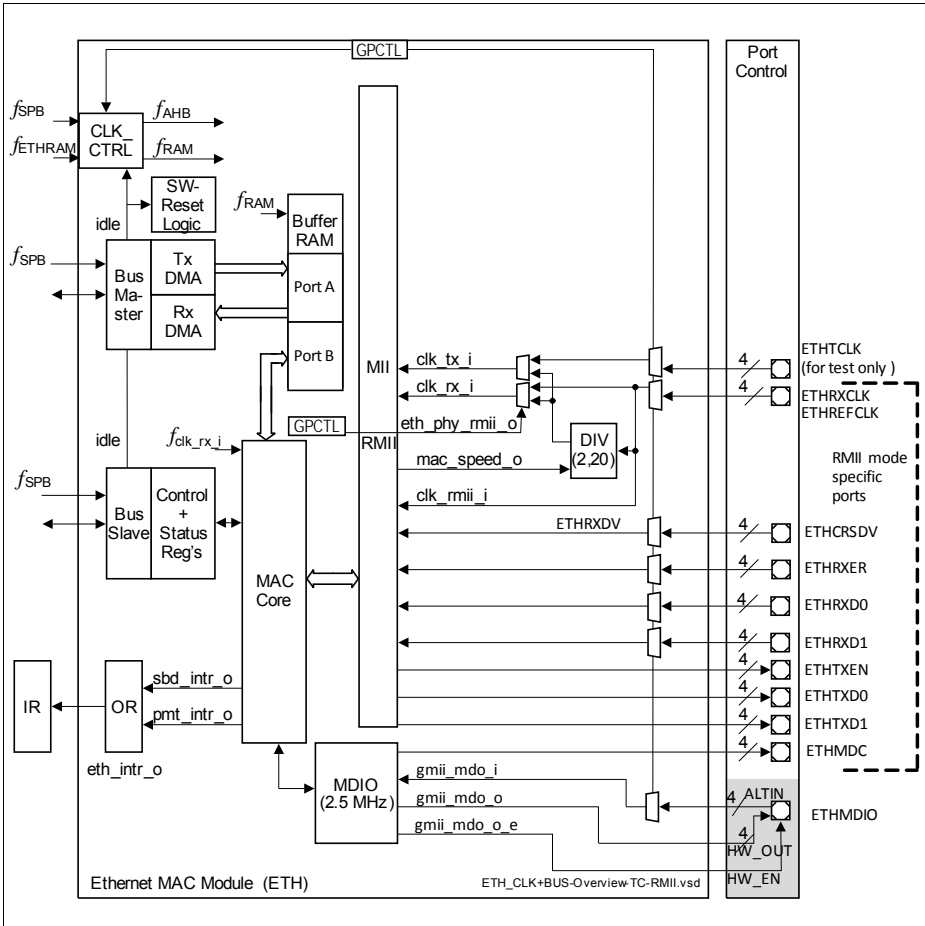


Figure 35-31 Ethernet MAC Module Implementation and Interconnections

35.5.1.1 On-Chip Connections

This section describes the on-chip connections of the Ethernet MAC module.

Interrupt Service Requests

The module has one Service Request Node connecting it to the interrupt system. The interrupt request line is connected to the interrupt controller as shown in [Table 1-112](#).

Table 35-51 Service Request Lines of Ethernet MAC

Request Line	Connected to	Description
ETH_SRC0	eth_intr_o	DMA functions (sbd_intr_o), this internal line is connected via OR gate to ETH.SR0
ETH_SRC0	eth_intr_o	wake up on LAN (pmt_intr_o), this internal line is connected via OR gate to ETH.SR0

35.5.1.2 Clocks

The module has multiple clock inputs connecting it to the system. They are connected to the system as shown in [Table 1-113](#).

Table 35-52 Clock Lines of Ethernet MAC

Clock Line	Connected to	Description
hclk_i / f_{AHB}	f_{SPB}	AHB Master Interface Clock, divided by 2 if GPCTL.DIV is set
clk_ram_i / f_{RAM}	f_{ETHRAM}	RAM Clock (separate clock from SCU.CGU) <ul style="list-style-type: none"> it is by default $2 \times f_{\text{SPB}}$ must always be 2, 3, 4 ... (natural number) times higher than f_{SPB} it must be $\geq 4 \times \text{clk_tx_i} / \text{clk_rx_i}$ (Both are 25 MHz for 100 MBit/s and 2.5 MHz for 10 MBit/s) divided by 2 if GPCTL.DIV is set
clk_csr_i	f_{SPB}	AHB Slave Interface Clock
clk_tx_i	ETHTXCLK (port pin)	Transmission Clock from Phy needs a clock input during SW reset (ETH_BUS_MODE.SWR)
clk_rx_i	ETHRXCLK (port pin)	Reception Clock from Phy needs a clock input during SW reset (ETH_BUS_MODE.SWR)

Ethernet MAC (ETH)

Table 35-52 Clock Lines of Ethernet MAC (cont'd)

Clock Line	Connected to	Description
clk_rmii_i	ETHREFCLK (port pin)	50-MHz clock used by the RMII from Phy needs a clock input during SW reset (ETH_BUS_MODE.SWR)
clk_ptp_ref_i	f_{SPB}	Reference Clock for the Time Stamp Update Logic

35.5.2 Ethernet MAC Module-Implementation Related Registers

The registers listed in [Figure 35-32](#) Ethernet MAC must be programmed for proper operation of the Ethernet MAC module.

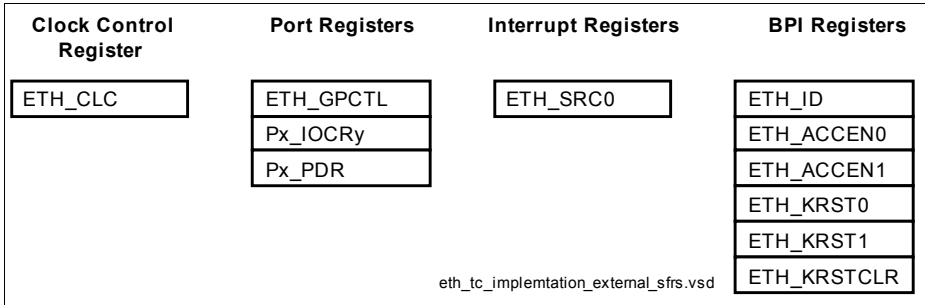


Figure 35-32 Ethernet MAC Implementation-specific Special Function Registers

35.5.2.1 Port Control

The Ethernet MAC is connected to general purpose I/O lines. Each line is connected to up to four different general purpose I/O lines that can be chosen alternatively. The selection from one of these alternatives is done by application SW by programming ETH_GPCTL.ALTIx respectively.

For MDIO, ALTIx selects the direction control signal (“HW_OUT control”) gmii_mdo_o_e_[x] as well. The signals gmii_mdo_o_e_[n] that are not selected are forced passive.

The following port control operations and selections must be executed for these I/O lines:

- Input/output function selection (Port IOCR registers)
- Pad driver characteristics selection for the outputs (Port PDR registers)

[Table 35-53](#) shows an overview how bits and bit fields must be programmed for the required I/O functionality of the Ethernet MAC I/O lines.

Note: Some PHYs provide an interrupt output, signalling e.g. line activity. It might be useful to connect this to an interrupt input on system level. This signal is not connected to the Ethernet MAC!

Table 35-53 Ethernet MAC I/O Control Selection and Setup

Port	MAC signals	R/ Mii	Input Select Register	Input/Output Control Register Bits	I/ O
P02.8	ETHMDC		na	P02_IOCR8.PC8 = 1X110 _B	O

Ethernet MAC (ETH)
Table 35-53 Ethernet MAC I/O Control Selection and Setup (cont'd)

Port	MAC signals	R/ Mii	Input Select Register	Input/Output Control Register Bits	I/ O
P21.0	ETHMDC		na	P21_IOCRO.PC0 = 1X110 _B	O
P21.2	ETHMDC		na	P21_IOCRO.PC2 = 1X101 _B	O
P00.0	ETHMDIOA		ETH_GPCTL.ALTIO = 00 _B	P00_IOCRO.PC0 = 0XXX _B	I/ O
	gmii_mdo_o_e_0 (I/O direction control of MDIO) connected to HW_DIR and HW_EN input of this port; choose Single EN - the pin is controlled by its own, dedicated, single HW_EN signal; gmii_mdo_o is connected to HW_OUT of the port, gmii_mdo_i is connected to ALTIN of the port (via input demultiplexer controlled by ALTI).				
P21.1	ETHMDIOB		ETH_GPCTL.ALTIO = 01 _B	P21_IOCRO.PC1 = 1X110 _B	I/ O
	No support of I/O direction control by HW. Unidirectional input or output only as configured in P21_IOCRO.PC1 by SW.				
P21.3	ETHMDIOD		ETH_GPCTL.ALTIO = 11 _B	P21_IOCRO.PC3 = 0XXX _B	I/ O
	gmii_mdo_o_e_3 (I/O direction control of MDIO) connected to HW_DIR and HW_EN input of this port; choose Single EN - the pin is controlled by its own, dedicated, single HW_EN signal; gmii_mdo_o is connected to HW_OUT of the port, gmii_mdo_i is connected to ALTIN of the port (via input demultiplexer controlled by ALTI).				
P11.12	ETHREFCLKA	R	ETH_GPCTL.ALT11 = 00 _B	P11_IOCRO.PC12 = 0XXX _B	I
	RMII: EXTCLK1 can be selected as output (P11_IOCRO.PC12 = 1X110 _B) while ETH_GPCTL.ALT11 selects this pin as input. This setting supplies both: external PHY and this module with ETHREFCLK. Note that the jitter requirement of IEEE802.3 can not be met with this internal clock. If used as REFCLK, use fast input mode (P11_PDR1.PD12 = X00 _B and P11_IOCRO.PC12 = 0XXXX _B)				
P11.12	ETHXCLKB	R	ETH_GPCTL.ALT110 = 01 _B	P11_IOCRO.PC12 = 0XXX _B	I
	Not for application but for SW development and test. This allows to use MII loop back without external circuit if TXCLK and RXCLK are clocked with EXTCLK1.				

Ethernet MAC (ETH)
Table 35-53 Ethernet MAC I/O Control Selection and Setup (cont'd)

Port	MAC signals	R/ Mii	Input Select Register	Input/Output Control Register Bits	I/ O
P11.11	ETHCRSDVA	R	ETH_GPCTL.ALT14 = 00 _B	P11_IOC8.PC11 = 0XXX _B	I
P21.7	ETHRXERB	R	ETH_GPCTL.ALT15 = 01 _B	P21_IOC4.PC7 = 0XXX _B	I
P11.10	ETHRXD0A	R	ETH_GPCTL.ALT16 = 00 _B	P11_IOC8.PC10 = 0XXX _B	I
P11.9	ETHRXD1A	R	ETH_GPCTL.ALT17 = 00 _B	P11_IOC8.PC9 = 0XXX _B	I
P11.6	ETHTXEN	R	na	P11_IOC4.PC6 = 1X110 _B P11_PCSR.SEL6 = 1 _B	O
P11.3	ETHTXD0	R	na	P11_IOC0.PC3 = 1X110 _B P11_PCSR.SEL3 = 1 _B	O
P11.2	ETHTXD1	R	na	P11_IOC0.PC2 = 1X110 _B P11_PCSR.SEL2 = 1 _B	O

35.5.2.2 Clock Control

The master interface (DMAs) of the Ethernet MAC is connected to f_{SPB} via a clock control in register GPCTL.

Note: The RAM frequency needs to run at double the application clock!

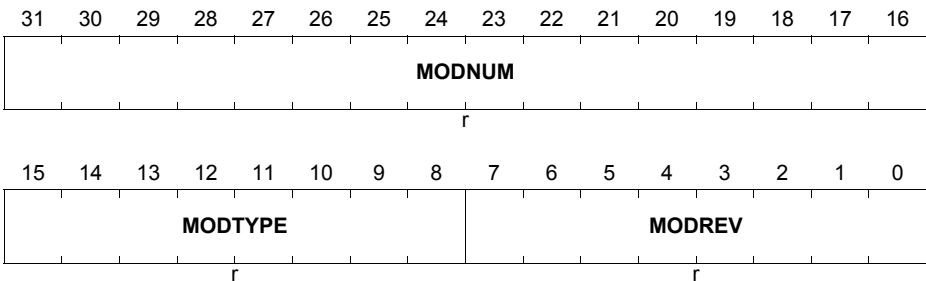
35.5.2.3 Additional Register

Module Identification Register

The Ethernet MAC Module Identification Register ID contains read-only information about the module version.

ETH_ID

Module Identification Register (0004_H) **Reset Value: 00BF C0XX_H**



Field	Bits	Type	Description
MODREV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01 _H (first revision).
MODTYPE	[15:8]	r	Module Type This bit field defines the module as a 32-bit module: C0 _H
MODNUM	[31:16]	r	Module Number Value This bit field defines the module identification number for the Ethernet MAC: 00BF _H

Clock Control Register (CLC)

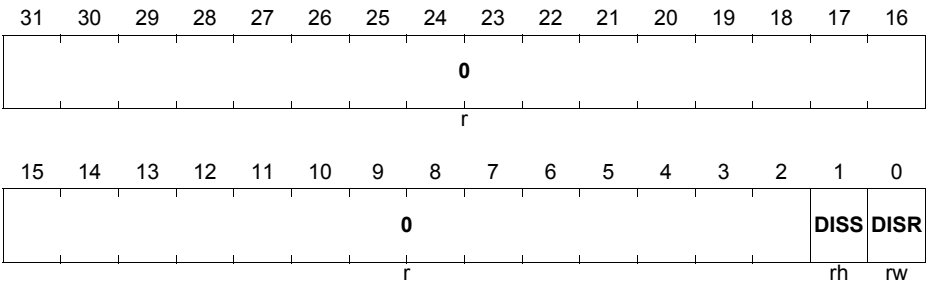
The Clock Control Register CLC allows the programmer to adapt the functionality and power consumption of the module to the requirements of the application. The description below shows the clock control register functionality which is implemented in the BPI_FPI for the module. Where a module kernel is connected to the CLC clock control interface, CLC controls the module clock for the module.

ETH_CLC

Clock Control Register

(0000_H)

Reset Value: 0000 0003_H



Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module. The disable request is delayed internally until all pending transactions on the master and slave interface to the SPB are completed.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Note: After a hardware reset operation, the $f_{Ethernet\ MAC}$ clock is switched off and the Ethernet MAC module is disabled (DISS set).

Input and Output Control Register Functions

The Input and Output Control Register GPCTLx determines for the Ethernet MAC which alternate input is to be used and selects the phy interface.

This register is reset by power on reset and hardware reset.

Note: This register is not affected by the local module reset!

ETH_GPCTL

Input and Output Control Register (0008_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0						DIV	EPR	0		ALT10	ALT9	ALT8			
r						rw	rw	r		rw	rw	rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALT17		ALT16		ALT15		ALT14		ALT13		ALT12		ALT11		ALT10	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
ALT _y (y = 0-10)	[2*y+1: 2*y]	rw	Alternate Input Select Selects the alternate input for channel y. If no signal is connected to an alternate input, it is connected to GND. 0000 _B Alternate Input 0 selected 0001 _B Alternate Input 1 selected ... _B ... 0011 _B Alternate Input 3 selected

Ethernet MAC (ETH)

Field	Bits	Type	Description
EPR	24	rw	External Phy Interface RMMI Mode Bit Used to select the phy interface during module reset of MAC triggered by setting SWR bit of DMA Register 0. If set during this reset, RMI is selected and an internal predivider divides the clock clk_rmii_i from pin ETHRXC by 2 or 20 and provides it to the internal signals clk_tx_i and clk_rx_i . The division factor depends from the signal mac_speed_o , controlled by bit FES in Register 0 (MAC Configuration Register). If cleared during the module reset said above, MII is selected and the divider is not active but the clock signals clk_tx_i and clk_rx_i are connected directly to the referring port pins. Note that the status of this bit is latched in during module reset only.
DIV	25	rw	Module Clock Divider Request Bit If set, f_{AHB} and f_{RAM} will be divided by 2.
0	[31:26] , [23:22]	r	Reserved Read as 0; should be written with 0.

Note: After a hardware reset, all clocks are switched off and the Ethernet module is disabled (DISS set). In this case, only register CLC is accessible for configuration.

Access Enable Register (ACCEN0)

The Access Enable Register 0 controls write access¹⁾ for transactions with the on chip bus master TAG ID 000000_B to 011111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6-bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000_B , EN1 -> TAG ID 000001_B , ... , EN31 -> TAG ID 011111_B .

1) The BPI_FPI Access Enable functionality controls only write transactions to the CLC, KRSTx and the kernel registers. Read transactions are not influenced. SW has to take care for destructive/modifying read functionality in kernel registers

ETH_ACCEN0
Access Enable Register 0
(000C_H)
Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN3	EN3	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN1	EN1	EN1	EN1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN1	EN1	EN1	EN1	EN1	EN1	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
5	4	3	2	1	0										
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

Field	Bits	Type	Description
ENn (n = 0-31)	n	r/w	Access Enable for Master TAG ID n This bit enables write access to the module kernel addresses for transactions with the Master TAG ID n 0 _B Write access will not be executed 1 _B Write access will be executed

Access Enable Register (ACCEN1)

The Access Enable Register 1 controls write access¹⁾ for transactions with the on chip bus master TAG ID 100000_B to 111111_B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The BPI_FPI is prepared for a 6-bit TAG ID. ACCEN1 is not implemented with register bits as the related TAG IDs are not used in the AURIX devices.

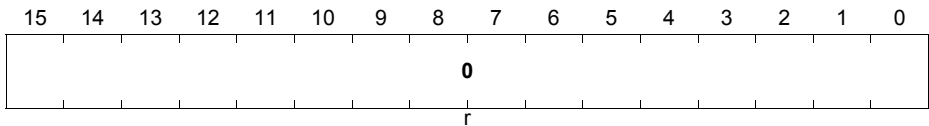
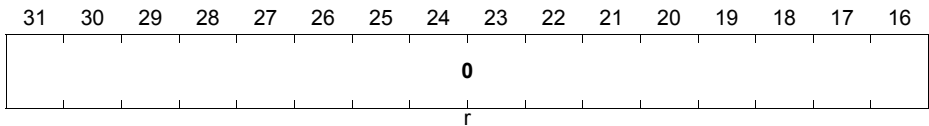
Mapping of TAG IDs to ACCEN1.ENx: EN0 -> TAG ID 100000_B, EN1 -> TAG ID 100001_B, ... , EN31 -> TAG ID 111111_B.

ETH_ACCEN1

Access Enable Register 1

(0010_H)

Reset Value: 0000 0000_H



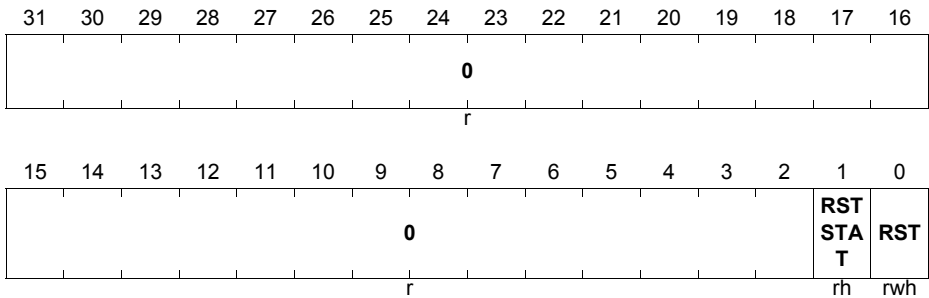
Field	Bits	Type	Description
0	[31:0]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 0 (KRST0)

The Kernel Reset Register 0 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset Registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to '1' by the BPI_FPI in the same clock cycle the RST bit is re-set by the BPI_FPI. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLE.CLR register bit.

Note: During the execution of the kernel reset until RSTSTAT is set, access to the kernel registers will result in an error acknowledge.

ETH_KRST0
Kernel Reset Register 0
(0014_H)
Reset Value: 0000 0000_H


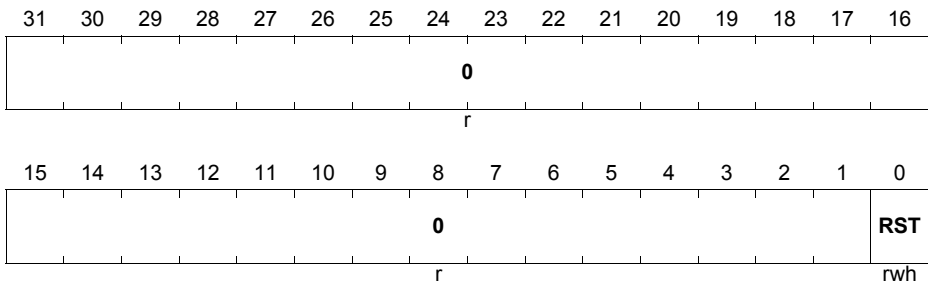
Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set and both: master and slave interface are idle. 0_B No kernel reset was requested 1_B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
RSTSTAT	1	rh	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set by the BPI_FPI after the execution of a kernel reset in the same clock cycle both reset bits. 0_B No kernel reset was executed 1_B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is

Ethernet MAC (ETH)

necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

ETH_KRST1
Kernel Reset Register 1
(0018_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set and both: master and slave interface are idle. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (re-set to '0') by the BPI_FPI after the kernel reset was executed.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

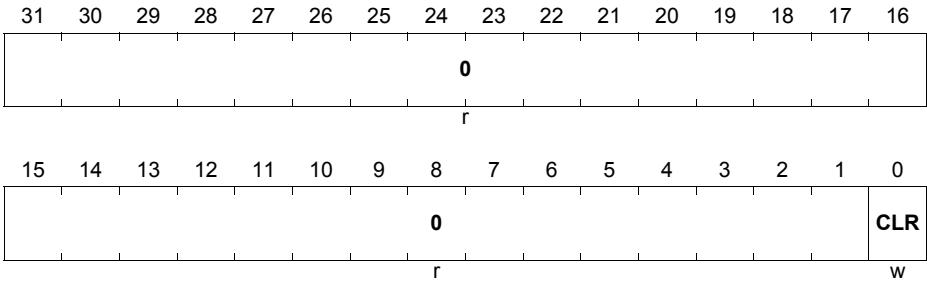
Kernel Reset Status Clear Register (KRSTCLR)

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

ETH_KRSTCLR

Kernel Reset Status Clear Register (001C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0.
0	[31:1]	r	Reserved Read as 0; should be written with 0.

35.6 Revision History

This User's Manual is based on: **Revision DWC Ether MAC 10/100/1000 Universal, V3.7a**".

Table 35-54 Revision History

Version Number	Changes to Previous Version
Rev_0.1	First Draft in IFX template
Rev_0.2	First configured version
Rev_0.3	First TS version
Rev_0.4	First ITS version
Rev_0.5	TS version
Rev_0.8	Removed unresolved cross references, Removed signal sbd_pwr_down_ack_o Added signals eth_phy_rmii_i and eth_rmii_speed_10_i
Rev_0.9	Updated Module Implementation Chapter
Rev_1.1	Expanded GMACREG19 headers for extraction
Rev_1.5	Removed non configured registers: GMACREG48 .. 54, DMAREG10 ..11 Updated reset values of registers GMACREG0, 8, 9, DMAREG22 Added clarification that normal descriptor format can not be supported
Rev_1.6	Updated Registers Chapter
Rev_1.7	Changed bit names to comply with compiler requirements - MAC_CONFIGURATION.2KPE -> TWOKPE - BUS_MODE.8xPBL -> PBLx8

Table 35-54 Revision History (cont'd)

Version Number	Changes to Previous Version
Rev_1.8	<p>P21.1 ETHMDIO: No more support of I/O direction control by HW. P21.3: ETHMDIO mapping added Outputs remapped from O5 to O6 on all output ports</p> <p>P11.11 ETHCRSDV renamed to ETHCRSDVA P11.11 ETHCRSDVA: added ETHRXDVA and ETHCRSB P11.14 ETHCRS renamed to ETHCRSA P11.14 ETHCRSA: added ETHRXDVB and ETHCRSDVB</p> <p>P11.12: ETHTXCLKB mapping added P11.5 ETHTXCLK renamed ETHTXCLKA Fixed Synopsys issue: 9000522312 ("MAC inserts incorrect IP Payload Checksum at incorrect location for IPv6 packets with Authentication extension header") Fixed Synopsys issue: 9000522313 ("MAC provides incorrect IP Payload Checksum Error status when IPv6 packet with Authentication extension header is received") P21.2: ETHMDC added</p>
Rev_1.9	<p>Added port configuration hints for TXD0, TXD1, TXEN and REFCLK P11.13: ETHRXER renamed to ETHRXERA P21.7: ETHRXERB added</p>
Rev_1.10	<p>Port configuration changed for products with RMI only</p>
Rev_1.11	<p>Port configuration changed for TXD2 and TXD3 Added Port configuration for TC23x</p>

Advanced Driver Assistance Subsystem (ADAS)**36 Advanced Driver Assistance Subsystem (ADAS)**

Some product derivatives include hardware support for Advanced Driver Assistance Systems (See Introduction chapter).

In ADAS derivative products the following additional hardware is present:

- Camera and RADAR Interface (CIF) (TC26x and TC29x only)
 - Parallel interface
 - On-the-fly JPEG encoding of camera signals
 - De-interleaving of external ADC radar signals
- Fast Fourier Transform Accelerator (FFT)
Integrated windowing support and up to 1024 point FFTs
- Extended Memory
- A high speed Backbone Bus
- Additional ADC channels (TC23x only)
 - Supports 4 channel concurrent sampling throughout all ADAS products
- A high speed input capture unit
 - For high precision frequency measurements of low speed signals
 - See QSPI module, section “High Speed Input Capture”

This chapter describes these peripherals in the context of radar and video based applications for Advanced Driver Assistance Systems

Figure 36-1 shows an overview of the ADAS subsystem.

Advanced Driver Assistance Subsystem (ADAS)

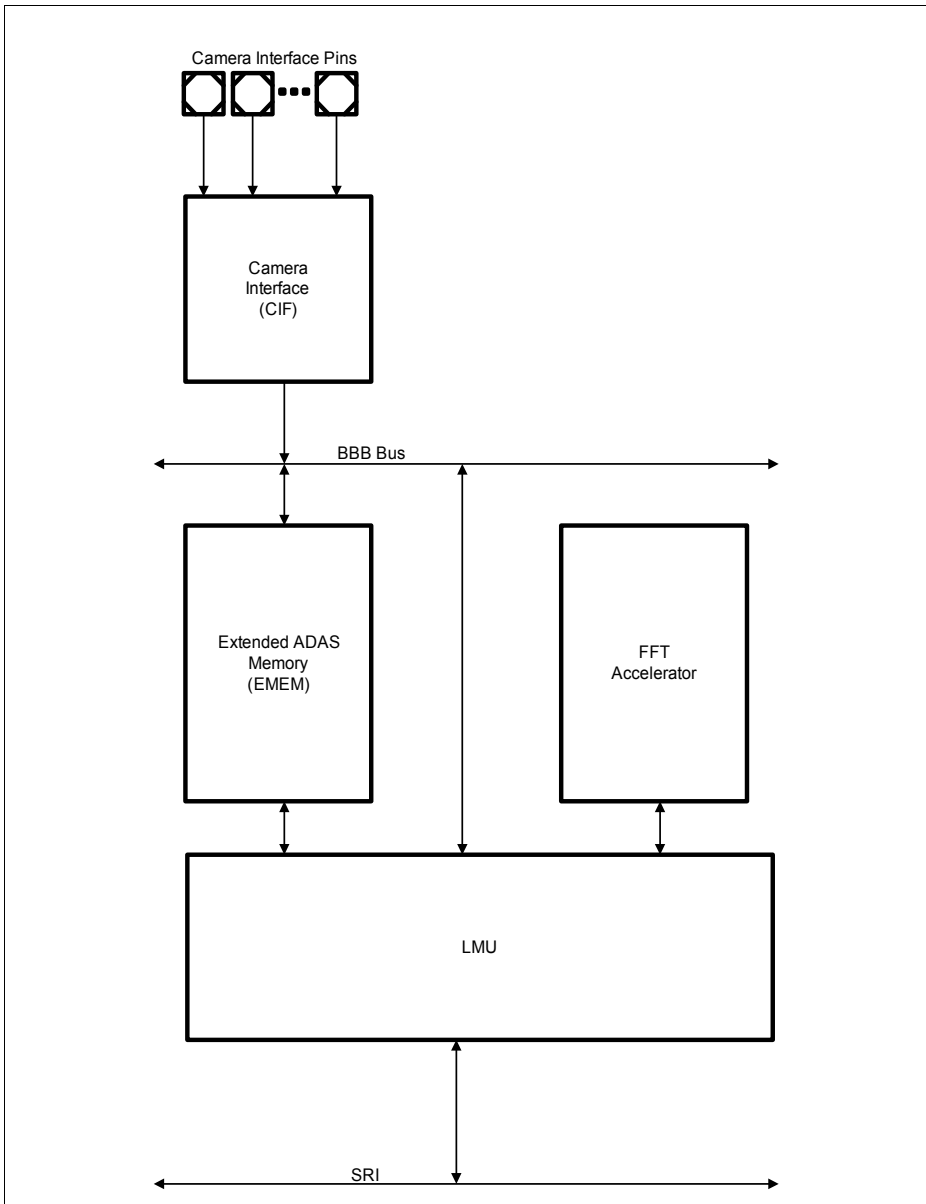


Figure 36-1 Block Diagram of the ADAS Subsystem

36.1 Camera and ADC Interface (CIF)

36.1.1 Introduction

The Camera and ADC Interface (CIF) represents a complete video and still picture input interface transferring data from an image sensor into EMEM. Furthermore several hardware blocks - performing image processing operations on the incoming data - are provided.

36.1.1.1 Camera and ADC Interface Functional Overview

Apart from providing the physical interfacing to various types of camera sensor modules, the CIF block implements image processing and encoding functionality. The integrated image processing unit supports image sensors with integrated YC_bC_r processing. Additionally the CIF also supports transfer of RAW (e.g. Bayer Pattern) images and non frame synchronized data packets.

The CIF block features a 16 bit parallel interface.

All output data is transmitted via the memory interface to a BBB system using the master interface.

Programming of the CIF is done by register read/write transactions using a BBB slave interface.

The following list summarizes the CIF's features:

- Throughput up to 96 MPixel/s
- 32-bit BBB slave programming interface
- BBB master interface
- ITU-R BT 601 compliant video interface supporting YC_bC_r and RAW data transfer
- ITU-R BT 656 compliant video interface supporting YC_bC_r and RAW data transfer
- Non line/frame aligned data transfer (data mode)
- 16-bit parallel camera and ADC interface
- YC_bC_r 4:2:2 processing
- Hardware JPEG encoder (supporting images up to a horizontal resolution of 1280 pixel) incl. JFIF1.02 stream generator and programmable quantization and Huffman tables
- Windowing and frame synchronization
- 1 Main and 5 Extra Image Cropping units allowing parallel transfer and position adjustment of up to 6 subregions
- Path from Main Image Stabilization or from one Extra Image Cropping unit to AGBT debug interface including a Metasymbol Generation unit inserting frame start and timing information into the stream
- Programmable watchdog timer to detect abortion/breaks in frame transmission

- Linear downscaling for the first extra path, supporting a mode for RGB Bayer Pattern
- Frame skip support for video (e.g. MPEG-4) encoding
- Macro block line, frame end, capture error, data loss interrupts and synchronization (h_start, v_start) interrupts
- Programmable polarity for synchronization signals
- Luminance/chrominance and chrominance blue/red swapping for YC_bC_r input signals
- Maximum input resolution of 4095x4095 pixels
- Buffer in EMEM organized as ring-buffer, supporting up to 2x8 Beat Bursts (2x32 Bytes)
- Buffer overflow protection for RAW data transfer and JPEG files
- Asynchronous reset input, software reset for the entire IP and separate software resets for all sub-modules
- Support of planar, semi planar and interleaved storage format (main path)
- Power management by software controlled clock disabling of currently not needed sub-modules

36.1.1.2 Camera and ADC Interface Block Diagram

Figure 36-2 shows the Camera and ADC Interface Block Diagram (top level view).

The BBB interface is divided into a master and a slave interface with their own clock domains. These clock domains may be asynchronous to the CIF Module Clock. To avoid data loss the frequency of the BBB clocks need to fulfill the following requirements:

- BBB Interface Master Clock (DMA) \geq CIF Module Clock
- BBB Interface Master Clock (DMA) $\leq 2 * \text{CIF Module Clock}$
- CIF Module Clock \geq Pixel Clock (PCLK)

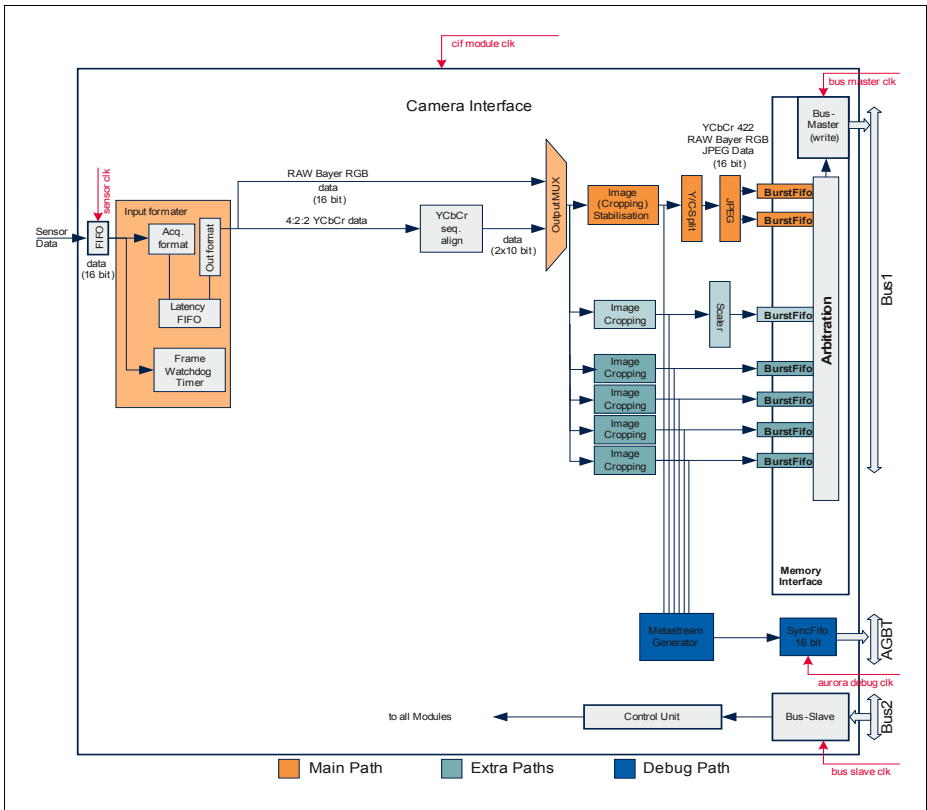


Figure 36-2 Camera and ADC Interface Block Diagram

36.1.2 Camera and ADC Interface Functional Specification

This chapter describes the CIF's functionality.

36.1.2.1 Target Applications

The CIF is targeted on various applications requiring a mega pixel still image and/or video input (e.g.: mobile devices, automotive or industrial vision, ...).

Camera Interface Example

A target application example is shown in [Figure 36-3](#).

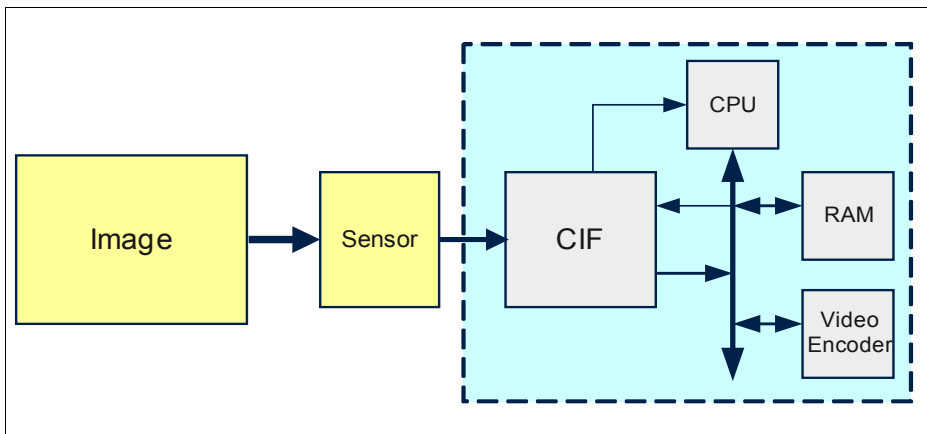


Figure 36-3 Target application example

The CIF provides a sensor/camera interface for a wide variety of video applications and it is optimized for high speed data transmission under terms of low power consumption.

This module is designed to be used for the following use cases:

- Video capturing/encoding
- Still image capturing in $YCbCr$ with on-the-fly JPEG encoding
- RAW frame data capturing
- Simple non line/frame aligned data reception (data mode) used for example for connecting to external ADCs

The CIF requires fast system memory for image storage in either planar/semi-planar/interleaved $YCbCr$ or RAW planar format or as JPEG compressed data. The integrated JPEG encoding engine is able to generate a full JFIF 1.02 compliant JPEG file that can be displayed directly by any image viewer.

All important YC_bC_r formats - which are used for video compression (e.g. MPEG4) for instance - are supported. For on-the-fly encoding macro block line interrupts are generated to trigger video encoding.

Connecting External ADC

The CIF provides a parallel interface that can be used to connect external ADC converters. In order to connect to one or more external ADCs, the CIF module requires a timer module capable of generating the appropriate clock and select signals, like the Generic Timer Module (GTM). The [Figure 36-4](#) shows an example of a connection between a single ADC and the CIF module.

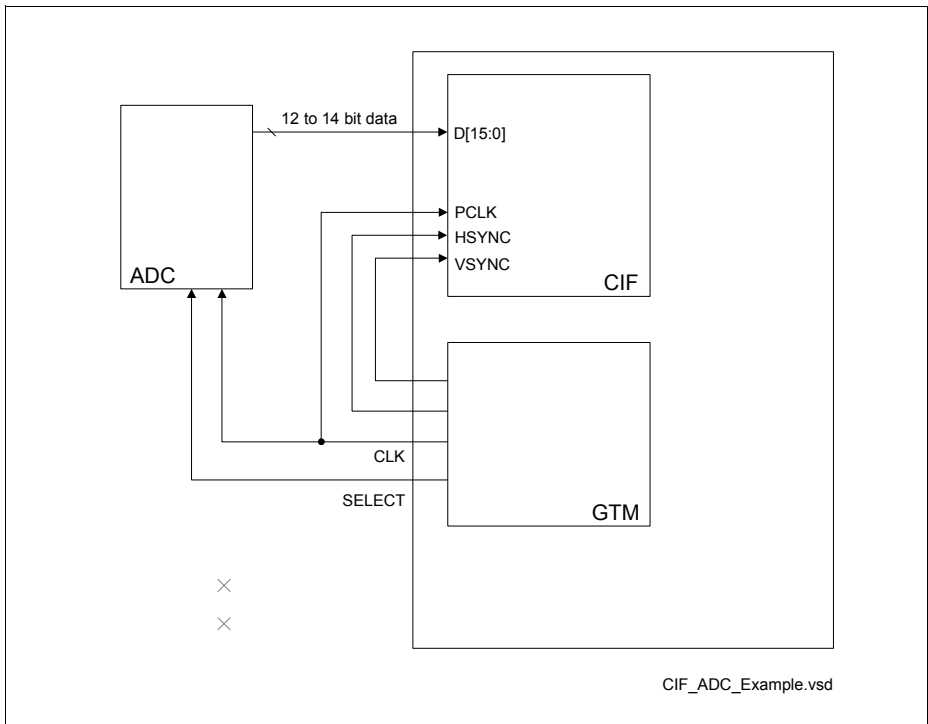


Figure 36-4 Example of a Connection to an External ADC

Connecting multiple ADCs to the CIF module may require an external multiplexer, and additional control signals to control it. Using twin or quad ADCs avoids the need for an external multiplexer.

The ADC data has to be connected MSB aligned to the CIF input pins. The **Figure 36-5** shows examples of data connections between a 12-bit or 14-bit ADC and the CIF module.

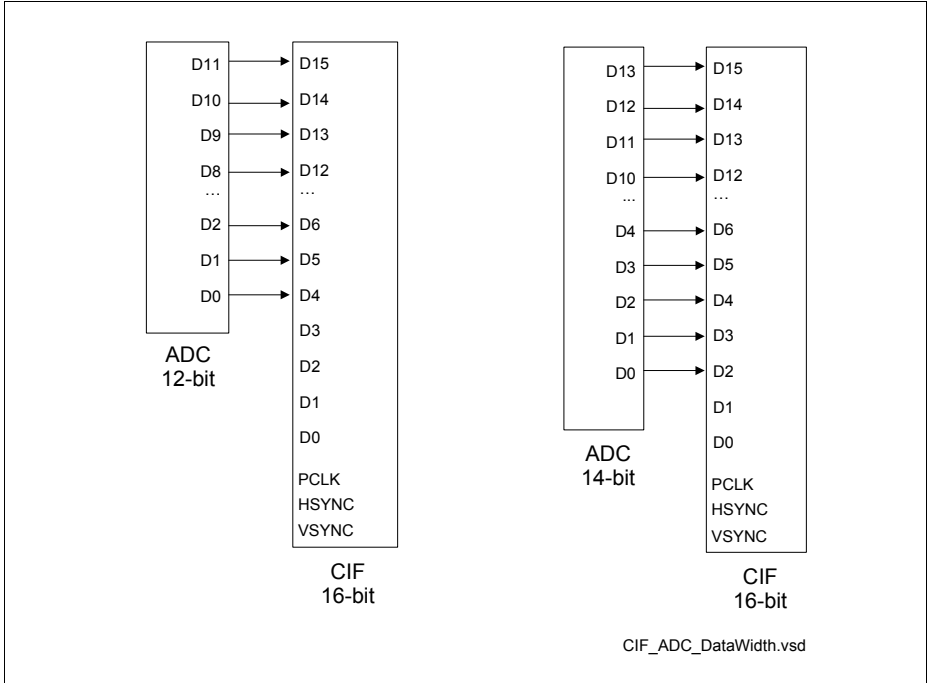


Figure 36-5 External ADC Data Alignment

The pixel clock PCLK controls the data transfer. The first edge of the clock signal can initiate the ADC to provide the data; the second edge can trigger the CIF to latch the data. In the [Figure 36-6](#) the providing of the next parallel data is referred to as “shifting”, which is what happens if the ADC has a data FIFO. The clock polarity can be programmed both in the GTM and in the CIF module. In the CIF module, the PCLK polarity is configured in by the bit field `CIFISP_ACQ_PROP.SAMPLE_EDGE`.

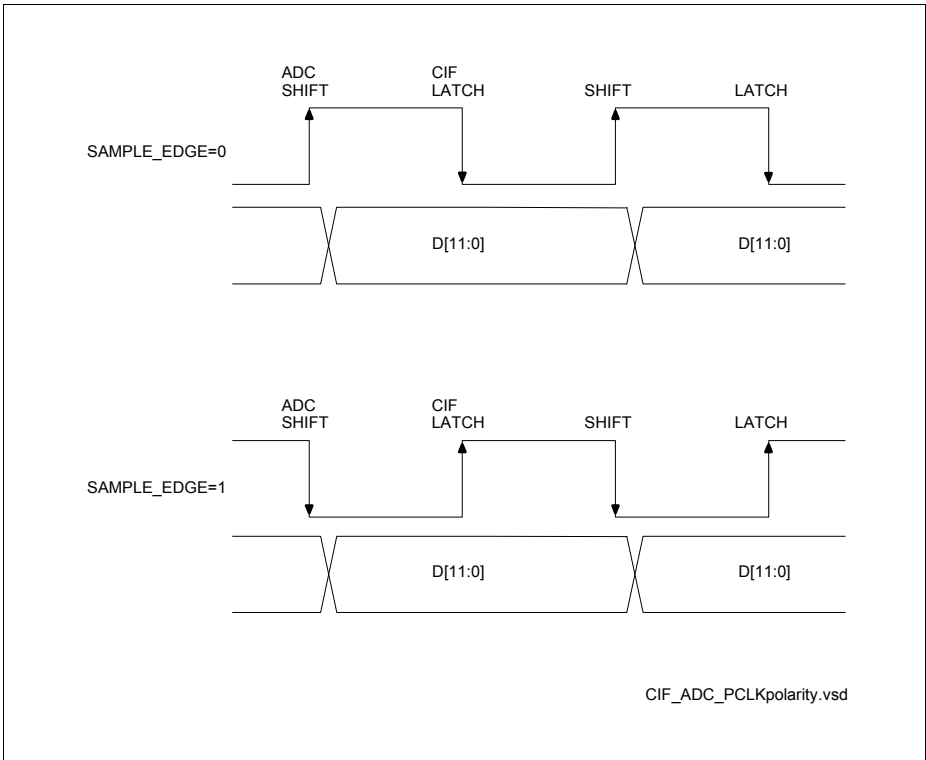


Figure 36-6 PCLK Clock Polarity and Data Transfer (12-bit ADC)

The polarity of the signals HSYNC and VSYNC can also be configured by using the bit fields `CIFISP_ACQ_PROP.HSYNC_POL` and `CIFISP_ACQ_PROP.VSYNC_POL`.

36.1.2.2 Functional Description

The CIF forms a complete video and still picture input interface solution for SoCs for various applications.

It is widely programmable and therefore very flexible to use. It supports input frame resolutions up to 4095x4095 pixels.

It consists of a Video Image Signal Processing unit (ISP), a Security Watchdog unit, luminance/chrominance splitter (Y/C Split), a linear downscaling unit, main and several extra cropping units, debug path, JPEG encoder, output unit, control unit (Ctrl) and a BBB slave interface for programming purposes.

All data transfer between EMEM and CIF is handled via the memory interface block.

For more information and a block diagram please refer to the sub module descriptions below and to [Section 36.1.1.1](#).

Sub Module ISP

The ISP is the interface to the attached sensor device. It accepts either ITU-R BT.601 YCbCr or RAW RGB Bayer data, or ITU-R BT.656 YCbCr or RAW RGB Bayer data. Furthermore a so called "data mode" is supported which accepts non line or frame organized data. In this mode the states of the hsync and vsync hardware lines are considered as "enable" and "transfer indicator". This mode allows connecting the CIF to any source delivering a parallel data stream (e.g. camera sensor including a JPEG encoder).

The input part of the ISP is fully programmable in terms of signal polarities, active video data positions, and luminance/chrominance order.

A handshake based mechanism is used for data qualification. As the sensor device can not be stopped from delivering data, a pipeline stall leads to an error if it occurs within an active video line.

The ISP also contains its own programming registers to be accessed by a 32 bit handshake interface.

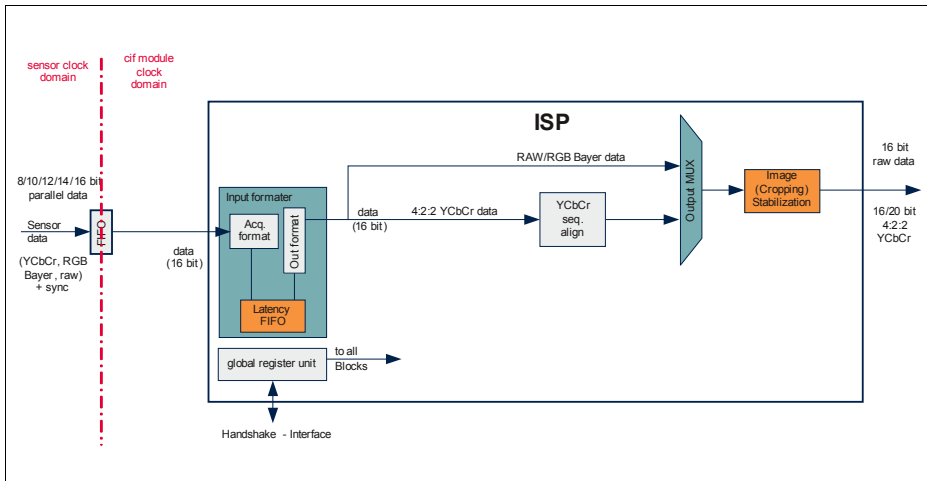


Figure 36-7 Block Diagram of the ISP Sub Module

The following features are implemented:

- CCIR656/601 compatible 16-bit video interface
- Input sampling on positive or negative sample clock
- Cropping of the output picture (to crop interpolation artifacts), also used for windowing

Sub Module Security Watchdog

The Security Watchdog unit is used to monitor the incoming image data. Specifically it can be used to detect and report irregularities or interruptions in the data stream. To do so the horizontal and vertical synchronization signals in the input formatter unit are observed and are compared to programmable time-out frames. When a time-out frame gets violated an interrupt is generated to immediately report this event.

The timing information is retrieved via 16-bit counters. There are two separate counters to measure horizontal and vertical timeframes in parallel. To enhance the configurability of the timing information an additional 16-bit prescaler controls the speed of the counter units, enabling a timing unit granularity from $1 \cdot T_{\text{cif_clk}}$ to $2^{16} \cdot T_{\text{cif_clk}}$.

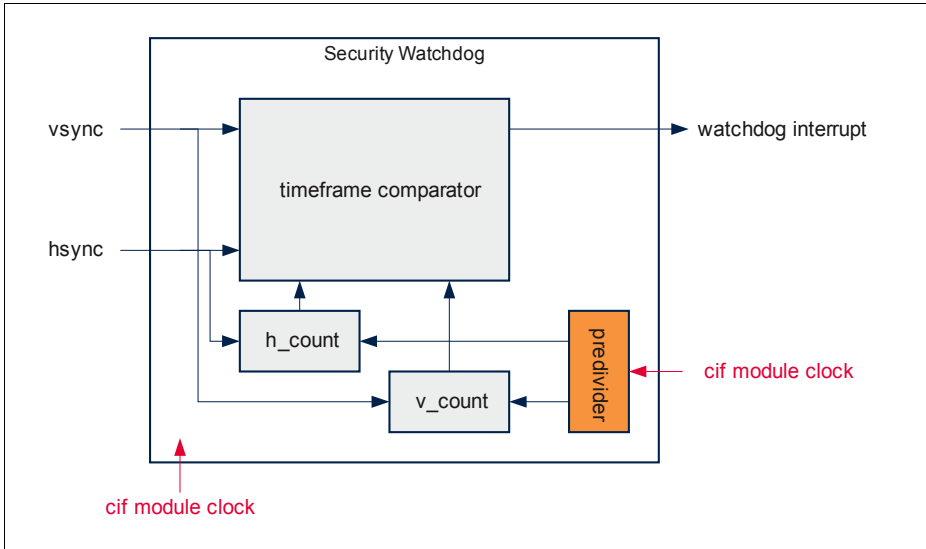


Figure 36-8 Block diagram of the Security Watchdog sub module

As can be seen in [Figure 36-9](#) there are four time frames of interest to be watched. The time between:

- Start and End of a frame (t_{VSE})
- End and Start of the next frame (t_{VES})
- Start and End of a line (t_{HSE})
- End and Start of the next line (t_{HES})

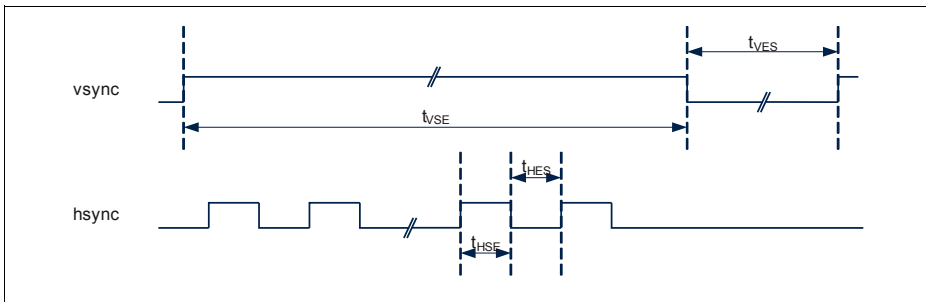


Figure 36-9 Programmable time-outs for the Security Watchdog

All those time-outs are programmable 16 bit values which can be set via the following registers: **“Watchdog Vertical Timeout Register”** on [Page 36-161](#) and **“Watchdog Horizontal Timeout Register”** on [Page 36-162](#). So as soon as the corresponding

counter reaches the programmed value an interrupt is triggered. A programmed value of zero means the time-out is not configured to trigger an event.

It is possible to completely disable the Security Watchdog unit by setting the corresponding bit of its (“**Watchdog Control Register**” on Page 36-160) or by disabling the clock of this sub module in the “**CIF Internal Clock Control Register**” on Page 36-51.

Sub Module Y/C-Split

This module contains a data path splitter separating luminance (Y) and chrominance (C) information of a 16 bit YCbCr 4:2:2. Independent 8 bit Y and C channels are the result of this operation.

The Y/C Splitter must ensure that the output Y and C data streams are fully aligned, i.e. if Y data are processed in one clock cycle only, the Y channel has to be stalled until the C component belonging to the processed Y component has been taken. **Figure 36-10** gives a coarse overview of the data flow from/to the sub-module.

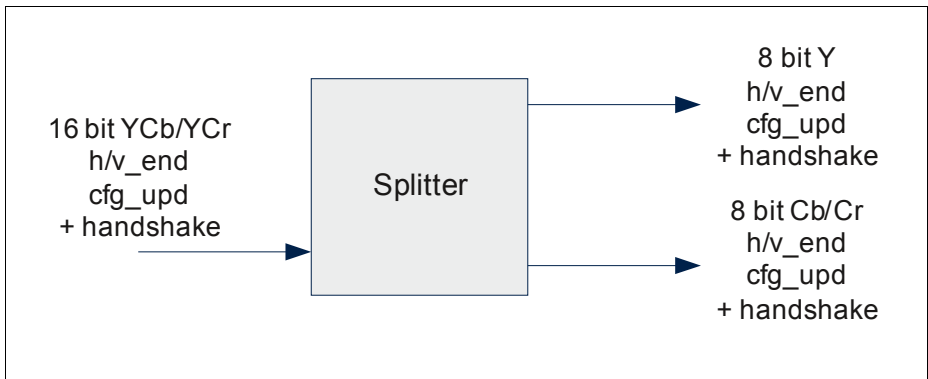


Figure 36-10 Block Diagram of the Y/C-Splitting Sub Module

Sub Module JPEG Encoder

The baseline JPEG encoder module consists of a JPEG encoder pipeline. The pipeline - which is shown in **Figure 36-11 “Block Diagram of the JPEG Encoder Sub Module” on Page 36-14** - is controlled by a register interface that is accessed via handshake interface. The following description gives an overview of the general functionality of the JPEG encoder.

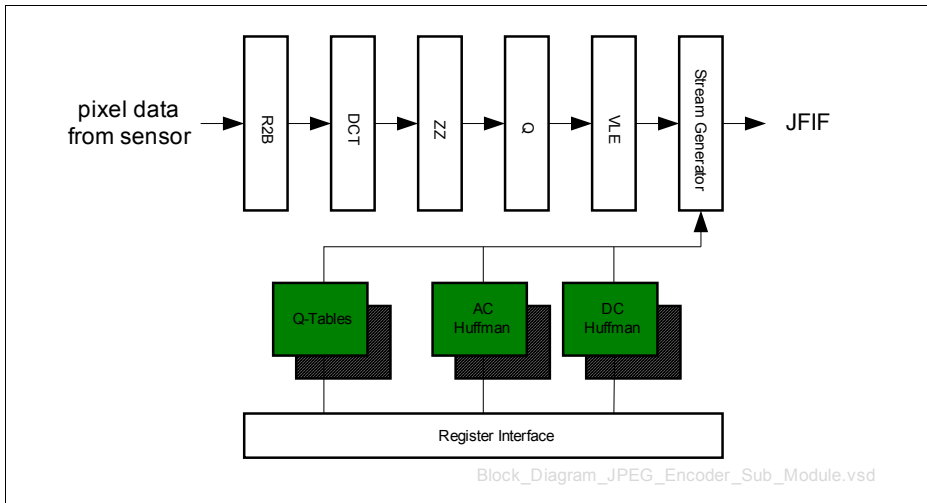


Figure 36-11 Block Diagram of the JPEG Encoder Sub Module

The encoding process starts with a raster to block conversion of the $YCbCr$ 4:2:2 pixel data provided e.g. by an imaging device. The incoming line oriented pixel data is reordered into 8x8 pixel blocks, one for each component.

Every 8x8 pixel block undergoes a baseline DCT computation, a zigzag reordering, a quantization and a variable length encoding (Huffman based). Last encoding step is the generation of the JFIF 1.02 compliant data stream by inserting markers and tables.

The encoder needs two clock cycles per pixel for processing, if $YCbCr$ 4:2:2 data is used, which is the normal mode of the JPEG encoder.

Sub Module Linear Downscaler

The first extra path includes a simple scaling unit which allows to reduce the size of the transferred image by simply skipping lines and columns of a frame. The amount of horizontal (columns) and vertical (lines) data reduction can be programmed separately. The module supports four modes:

- Single skip
- Double or RGB Bayer skip
- Single pass
- Double pass

The single skip mode simply skips every n^{th} line or column, but since this would destroy the integrity of a regular RGB Bayer pattern - see [Figure 36-12](#) - there is also the possibility to use the double skip mode which preserves the pattern.

Since skipping leads to a maximum data reduction of 50% at best there are corresponding pass modes to the skip modes. The difference of these modes is that only every n^{th} pixel (or ever n^{th} two pixels) get(s) passed to the following processing unit, all other pixels get skipped. In other words: The pass mode can downscale the image data at rates $\geq 50\%$, whereas the skip mode allows scaling at rates $\leq 50\%$.

Regarding the single/double pass modes it is also important to mention that whether the bayer pattern stays preserved or not is here only dependant on the pass-value. Odd values of n will destroy the bayer pattern integrity, whereas even values will preserve it.

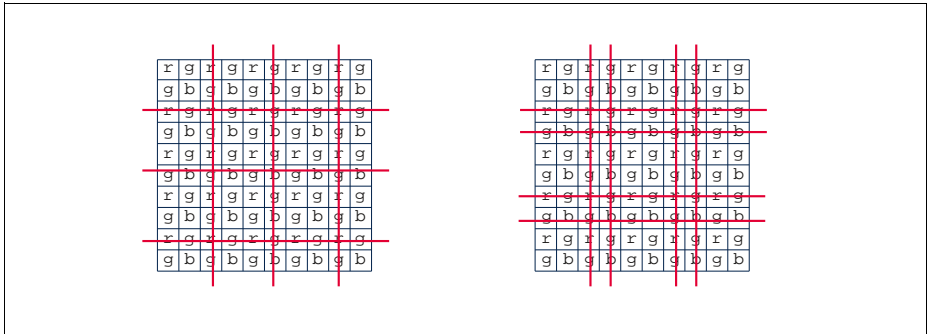


Figure 36-12 Single vs. RGB Bayer mode

It's further worth mentioning that the linear downscaler - like any other image processing unit - can be set to a bypass mode where data is passed on unprocessed to the next module.

Sub Module Extra Path Units

In addition to the main path, the CIF allows to define and transfer up to 5 additional image regions into memory. Those extra paths all have their own image cropping unit¹⁾ which allows to crop a separate sub-region of the image. Furthermore it is possible to utilize the displacement and recenter functionality of the image cropping unit to adjust the position of the cropping window (e.g. tracking a moving object). For details about the functionality of the image stabilization unit see the corresponding section in the CIF Programmer's Manual.

1) In the main path the Image Cropping unit is referred to as Image Stabilization unit, since in YCbCr video mode it may be used to compensate camera shakes.

Figure 36-13 shows the function of the main and extra path units: The main path may capture the full incoming frame utilizing it's downscaling unit to keep the output size - and hence the computational effort for ongoing image processing by the system - low.

In parallel up to 5 (the example below shows four) sub images may be cropped from the same input image. Those images are stored and handled independent of the main path and also independent of each other.

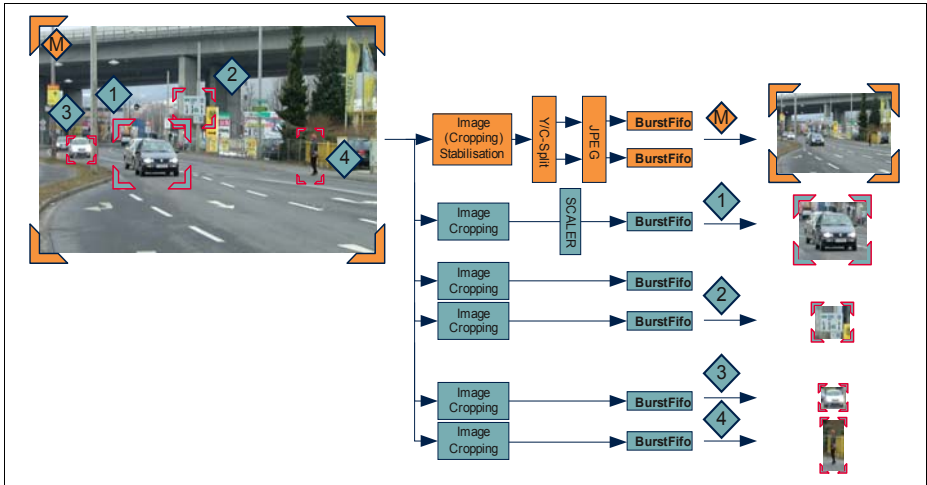


Figure 36-13 Example: Main and four active Extra Paths

Since the Y/C-Split unit is not present in the extra paths, the following restriction is given:

- only one receive buffer (no support for planar/semi-planar storage)

Nevertheless every extra path has it's own storage ring buffer and it's own distinct interrupt generation unit for those. Those separate interrupts of the distinct extra paths are then again merged in a single multiple source interrupt register to a single separate extra path interrupt line.

Only the first extra path has a downscaling unit, so for the other paths (including the main path) it is not possible to reduce the image dimension further after the cropping stages.

Every present extra path can be en/disabled via a programmable control register.

Parallel use of several extra paths

Using multiple overlapping or nested cropping windows leads to parallel activation of several extra paths generating bus load that could overload the memory interface and the back bone bus in case high baud rates are used. More then two nested windows is not recommended. Extra care should be taken when using overlapped cropping windows to keep the peak BBB bus load low. For example, use cropping windows widths which are multiple of the burst size configured in the MI (Memory Interface) block. This allows the cropped regions to be transferred to memory using only bursts on the BBB bus, without a need for single moves to transfer the pixels at the end of a line that do not

make a whole burst. For burst of four 32-bit moves, resulting in 16 bytes per burst, the optimal cropping window width is multiple of 16 pixels (1 byte per pixel). CIF single move takes 8 BBB clock cycles (0.5 bytes/clock cycle), burst of four 12 (1.33 bytes/clock cycle), and burst of eight 16 clock cycles (2 bytes/clock cycle). Burst of eight results in 200 Mbytes/sec for 100MHz BBB bus clock, which is the theoretical maximum if the CIF module is the only master on the BBB bus.

Debug Path

The debug path is used to transfer data from one of the CIF paths (main or extra) to an Aurora Gigabit Trace (AGBT) debug interface. To do so the data from one of the paths - which may also still be passed on to the memory interface - is additionally routed to the debug unit. Despite simply passing this data on to the AGBT interface, the CIF debug path also inserts meta information and CRC checksums to the stream. This helps to ease, enhance and secure the analysis of the retrieved data.

As can be seen in the [Figure 36-14](#) it is possible to select - via a MUX - from which path the data to send via the Aurora interface is taken. The selected data is then transferred into a latency compensation FIFO. This is needed to let the Metastream Generator send several additional information symbols before the received data stream, which shall not block the transfer of this stream to memory meanwhile.

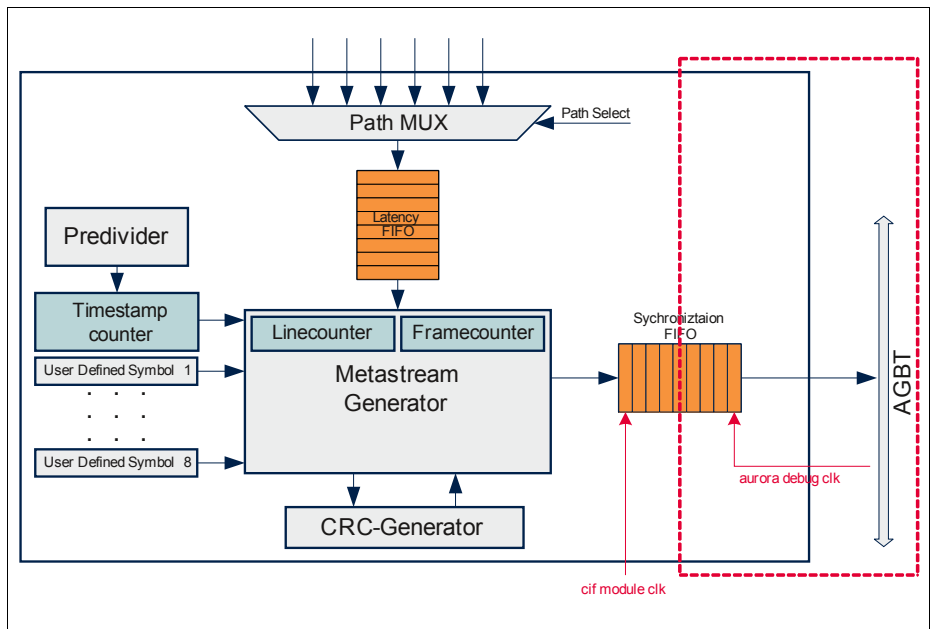


Figure 36-14 Debug Path block diagram

Those additionally inserted symbols are:

- Metasymbol marker
- Metastream type information
- a 30-bit timestamp
- a Frame-/Linenumber
- up to eight 15-bit user defined symbols
- Payload length Information
- Header & Payload checksum

Figure 36-15 shows an example of a pixel line as transferred over the debug interface. The pixel debug stream consists of 16-bit symbols, where the value $FFFF_H$ shall be a reserved value - the so called Metasymbol marker - which distinctly marks the start of a debug frame. This leads to some obvious restrictions in other symbols. The general rule reads as follows: In every symbol despite the Metasymbol marker the highest bit (MSB) has to be 0_B .

The currently supported Metastream types are:

- Start of Frame (SoF) Code 00_H
- Start of Line (SoL) Code 01_H

SoF is a simple Marker which only consists of the type and length information, timestamp, framenumber, user symbols and CRC. The SoF has no payload attached and hence no corresponding information like payload length or payload-CRC. Like the name suggests the SoF is sent only once at the beginning of an image frame. It has to be followed by - depending on the image size - several SoL packets.

Figure 36-15 shows an example of a SoL packet. Basically it contains the same information like a SoF packet, just with a linenumber instead of a framenumber and additional payload information.

As can be also seen in the example the payload symbols - which are the image pixels of a line - are right shifted by one & stuffed with zero on the high significant end. In case of 16 bit data this means that the least significant bit of the payload gets lost. In 14 bit data mode, the data is right-shifted for 1 bit, but the two LSB bits are always zero, so there is no data loss. In YUV16 mode the module shifts right and the chrominance LSB is lost. In RAW16 mode the module does not shift and the LSB bit is lost.

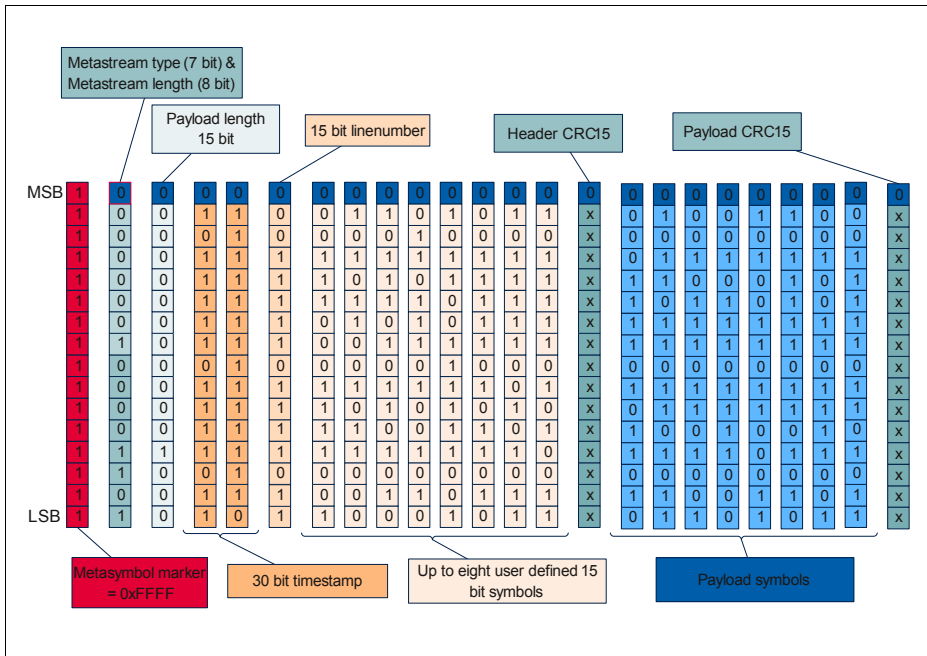


Figure 36-15 Line debug packet

The header length which is always included in a packet and the payload length which is additionally included in SoL packets can be used as additional security measures to quickly identify transmission gaps or similar issues. All given length information in the debug stream is referring to symbols. So in the example of **Figure 36-15** the payload length of 8 means that exactly 8 symbols of payload have to follow the header. The header length symbol gives the length of the header excluding the Metasymbol marker. For both values, header and payload - the CRC is not included.

The timestamp is a simple free running 30 bit counter and can be used to introduce quite accurate timing information to the stream. To enhance the flexibility of this counter a preceding prescaler unit is included which is clocked by the CIF module clock. This unit allows to configure the granularity of the timestamp to be between $1 \cdot T_{\text{cif module clk}}$ to $2^{32} \cdot T_{\text{cif module clk}}$.

User defined symbols are a way to insert/feed back software dependant information into the data stream. One possible application could be the insertion of coarse time/date information which supplements the internally generated timestamp.

The eight user defined symbols are freely programmable registers which can be included into the stream on a per symbol basis. Per default after reset no user defined symbols will be sent in the debug data stream.

Linenumber and framenumber are simple counters which can be reset by software at any time. The framenumber will run free and simply overflow starting again from zero, whereas the linenumber will be reset internally with every new SoF packet.

As CRC a 15 bit version using the same polynomial as the CAN-bus CRC is used. The input to this CRC is a full 16 bit symbol. The header CRC covers all symbols from the Metasymbol marker up to the last active user defined symbol. Whereas the payload CRC covers all payload symbols from the first after the header CRC up to the last.

If it is not needed it is possible to deactivate the whole debug path by disabling it's clock via the **“CIF Internal Clock Control Register” on Page 36-51**.

Note: The CIF module shifts the pixel data to the right when transferring via AGBT, in order to guarantee data symbol MSB of zero. Therefore, the LSB bit is lost in case the data is LSB aligned. This is valid for YUV data and 16-bit radar data. The raw RGB data is 14-bit MSB aligned and no LSB loss occurs.

Sub Module Memory Interface (MI)

The Memory Interface module - shown as in **Figure 36-16** - has the following tasks:

- It collects the CIF internal data streams and writes them into the EMEM.
- It decouples the BBB master clock from the CIF module clock.

Attention: In order to configure the CIF BBB master to access the EMEM, see the dedicated address map for BBB master accesses in the EMEM chapter.

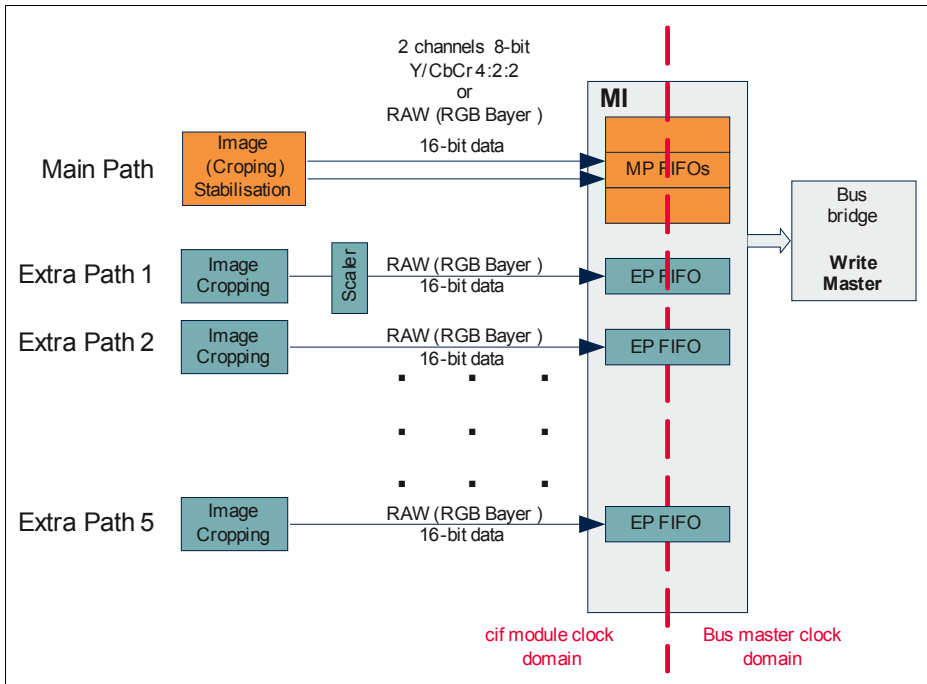


Figure 36-16 Overview Memory Interface connections

The MI supports storage of main picture YCbCr, JPEG and RAW data (up to 16 bit). Y, Cb and Cr data is stored in EMEM supporting the following modes:

- separate buffers (planar)
- two buffers (semi planar)
- one buffer (interleaved)

Figure 36-17 gives an overview of those three modes, but also shows the effect of the byte swapping option which effects the endianness of the output format.

All FIFOs used in MI are synchronization FIFOs which handle the task of decoupling the different clock domains.

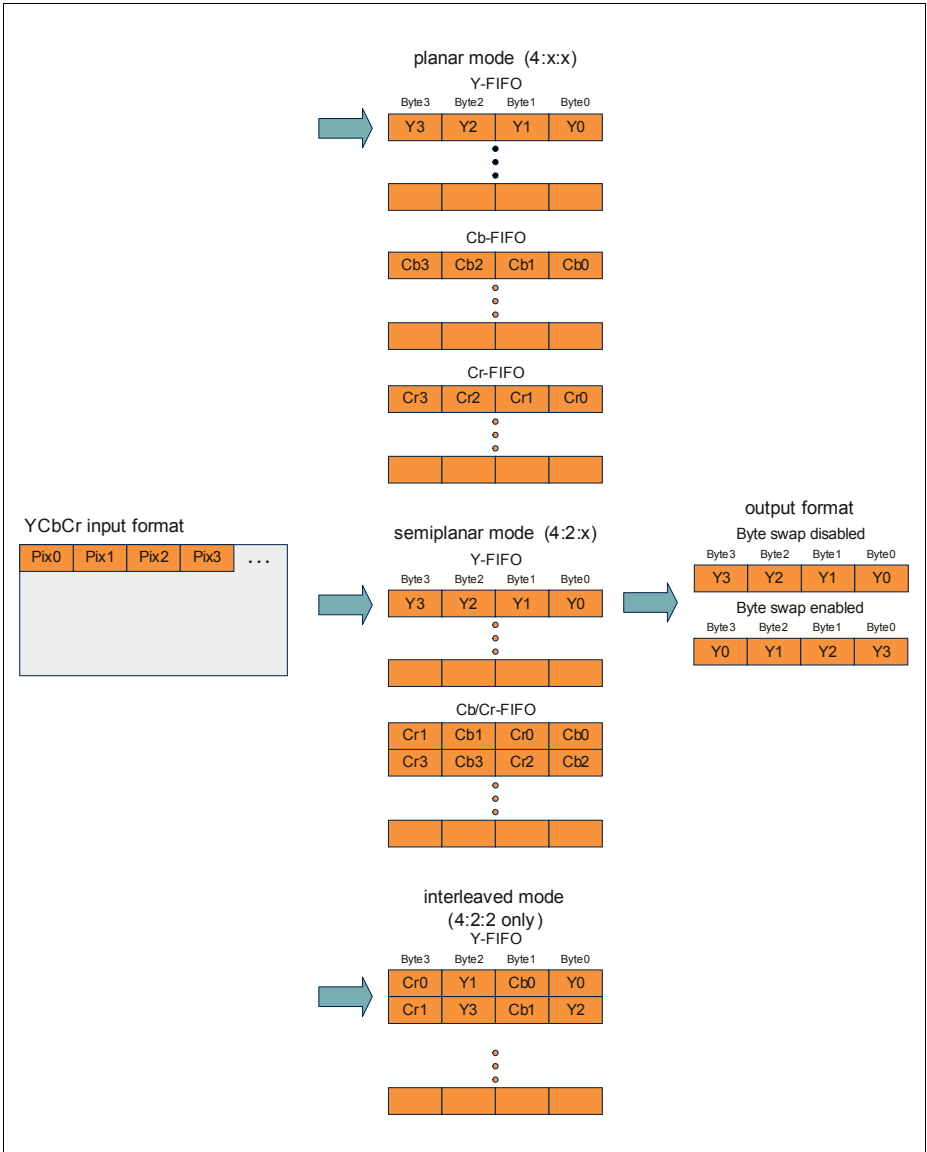


Figure 36-17 Storage scheme YCbCr in MI

For storage of RAW data only one FIFO buffer is supported/needed. If more than 8 bits data are required in RAW mode, then the data bits are stored as 16-bit words in memory. Usually these bits are stored MSB aligned, meaning the unused least significant bits remain zero or become a replica of the RAW data most significant bits (see **“INPUT_SELECTION” on Page 36-58**). It is also possible to disable this MSB alignment (**“INPUT_SELECTION_NO_APP” on Page 36-57**) leading to LSB aligned data values. This especially makes sense for non image RAW data (e.g. data mode). **Figure 36-18** visualizes the previously mentioned facts.

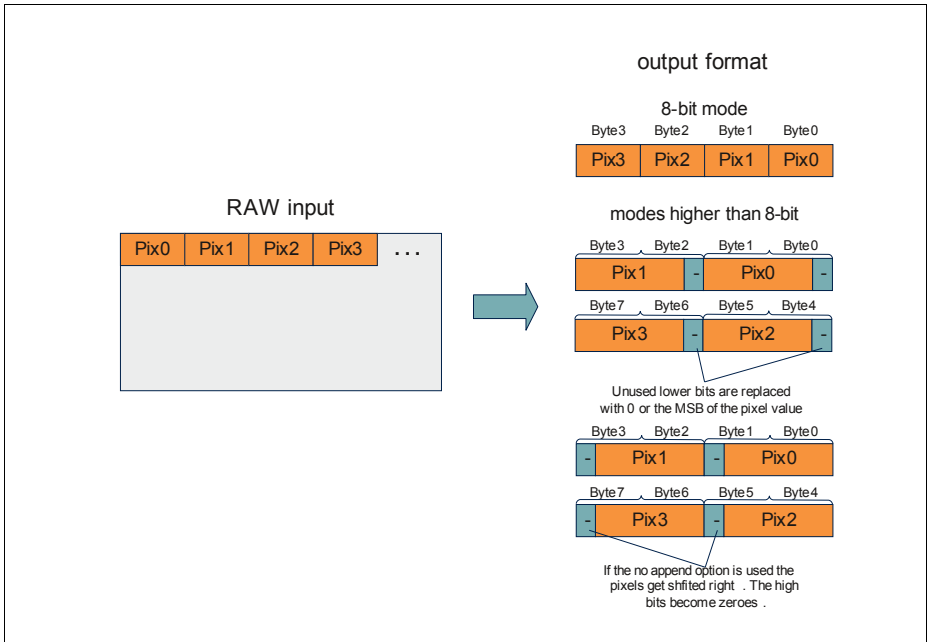


Figure 36-18 RAW data storage scheme in MI

Unlike the main path the extra paths support interleaved storage mode only. This is given by the fact that extra paths only have a single component FIFO.

Write to EMEM

The following **Figure 36-19** shows the basic building blocks of the unit. As can be seen there are several input port handshake interfaces:

- 2 x 8 bit Y, Cb/Cr main picture data or up to 16 bit RAW data
- 64 bit JPEG data
- For every extra path 2 x 8 bit Y, Cb/Cr extra picture data¹⁾ or up to 16 bit RAW data

The following modes of operation are supported:

- image data only
- JPEG data only
- RAW data only

The interface to the EMEM is a simple handshake oriented protocol, which is connected to the BBB master via a bridge. Three kinds of bus accesses are supported, 8-beat bursts, 4-beat bursts and single transfers, all with a data width of 32 bit. The storage areas, which are located in the EMEM, are organized as ring buffers.

Interrupts are generated for the following events:

- picture end of frame
- macro block line (16 lines of Y data and 8 lines of Cb and Cr data of the picture are written to the EMEM)
- programmed fill level of picture Y data reached
- wrap around of the programmed address range separately for Y, Cb and Cr of the picture
- BBB write error propagated by the handshake target interface

Each FIFO can store at least two 8-beat bursts.

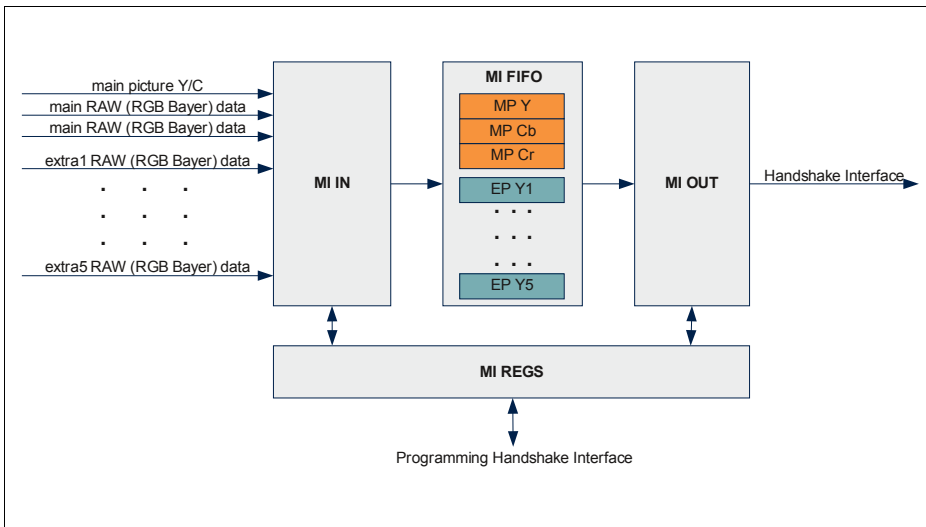


Figure 36-19 Block Diagram of the Memory Interface write path

BBB Master Interface

The BBB Master Interface connects the Memory Interface to the BBB.

1) YCbCr data is handled interleaved as 16 bit data extra paths.

- Unique TAG
- Word Access only
- Priority relative to CPU access via EMEM
- Bandwidth on bus
- RAM arbitration inside EMEM
- Reset behavior
- Bus error handling

BBB Slave Interface

The BBB Slave Interface connects the Control Unit to the BBB system. Internally it represents an AHB-Slave-to-handshake-Initiator block, while from the BBB a standard SPB_BPI is used.

- No safety features.
- No OCDS functionality (Request: Add monitoring via OTGM for the next product!).
- No ENDINT/OEN/... protection
- Standard CLC register.
- Allowed access width is 32-bit only.
8-bit and 16-bit accesses are treated as 32-bit and the content of all 32 data signals of the bus is written to the register.
- Error handling - no notification of the read and write accesses to reserved locations

Control Unit

The Control Unit serves two purposes:

- Interface from BBB Slave Interface to the local configuration register blocks of the other modules
- Clock and reset control registers for the CIF

The Control Unit has one 32 bit handshake input interface, three 16 bit handshake output interfaces, and two 32 bit handshake output interfaces.

As all transfers from the handshake input interface are 32 bit wide, these accesses have to be mapped accordingly to the 16 bit handshake interfaces, e.g. ignoring the upper 16 bits or filling them with zeros respectively.

Each block inside the CIF uses a dedicated clock signal that can be controlled by a programmable register inside the Control Unit.

Existing software resets in the CIF blocks can also be controlled by a programmable register inside the Control Unit. An asynchronous reset for the sensor clock domain has to be generated from the system reset. A soft reset for all registers in the CIF is provided. It behaves like the asynchronous hardware reset.

Shadow Registers

A distinctive feature of the local configuration registers are so called shadow registers. Because it can not be ensured that the complete re-programming of the CIF can be done during the vertical blanking period between two frames shadow registers have been introduced. These shadow registers hold the configuration values that are actually used for processing and are only updated with the latest written configuration values after a special update signal is triggered. This update signal may be generated after a frame has ended but in order to allow a fast configuration it may also be forced to trigger immediately.

36.1.3 CIF Registers Description

The following section describes the CIF registers on bit level.

36.1.3.1 CIF Registers Overview

The communication and control flow to and from the CIF module is realized via internal registers. Access to the registers will be provided via an BBB slave interface. Each access is 32 bit aligned.

The CIF uses a distributed configuration register scheme. So there is no central unit containing all programming registers, but all CIF sub-modules contain their own programming registers.

Note: Write accesses to reserved access locations are ignored. Read access deliver all zeros. No traps are triggered. Accesses of bus masters not enabled with the ACCENx registers cause acknowledge error.

The register set is divided into the following register types:

- Control registers
- Configuration setting registers
- Shadow configuration setting registers
- Interrupt registers

Shadow registers are used to support dynamic data path re-programming. The next programming values are written into the configuration setting registers of a local configuration unit of a CIF internal block. Shadow registers are used for the processing part of the sub-modules to keep the current values for processing. Updating of these shadow registers can be triggered by either the configuration update bits in the sub-module control registers in terms of immediate update, or by triggering the “generate configuration update” bit in the ISP control register to automatically update all shadow registers in the whole data processing pipeline after the last pixel of a processed frame.

All defined configuration register addresses of the CIF are offset addresses. The CIF base address is defined by the external BBB address decoder.

Note: The table below shows the base addresses as seen by the CPUs, after the address translation done by the BBB bridge. On the BBB bus itself the base addresses start with F00E instead of F90E.

Table 36-1 Registers Address Space

Module	Base Address	End Address	Note
CIF	F90E 2000 _H	F90E 201B _H	Base = CIF_CONTROL_BASE
CIFISP	F90E 2400 _H	F90E 2647 _H	Base = CIF_ISP_BASE
CIFMI	F90E 3400 _H	F90E 3513 _H	Base = CIF_MI_BASE
CIFJPE	F90E 3800 _H	F90E 388F _H	Base = CIF_JPE_BASE

Table 36-1 Registers Address Space (cont'd)

Module	Base Address	End Address	Note
CIFISPIS	F90E 4300 _H	F90E 4333 _H	Base = CIF_IS_BASE
CIFWD	F90E 4400 _H	F90E 441F _H	Base = CIF_WATCHDOG_BASE
CIFLDS	F90E 4500 _H	F90E 4507 _H	Base = CIF_LIN_DSCALE_BASE
CIFDP	F90E 4700 _H	F90E 4733 _H	Base = CIF_DEBUG_PATH_BASE
CIFEP	F90E 4900 _H	F90E 5333 _H	Base = CIF_EP_IC_BASE
CIFMIEP	F90E 5400 _H	F90E 592F _H	Base = CIF_EXTRA_PATH_BASE
CIFBBB	F90E 1F00 _H	F90E 1FFF _H	Base = CIF_SPB_BASE

Table 36-2 Registers Overview (in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIF Control Registers, CIF Clock Control Registers			
CIF_CCL	Clock Control Register	0000 _H	36-48
CIF Control Registers, CIF Custom Registers			
CIF_ID	CIF Revision Identification Register	0008 _H	36-50
CIF Control Registers, CIF Internal Control Registers			
CIF_ICCL	CIF Internal Clock Control Register	0010 _H	36-51
CIF_IRCL	CIF Internal Reset Control Register	0014 _H	36-52
CIF_DPCL	CIF Data Path Control Register	0018 _H	36-54
ISP Programming Registers, ISP Control Registers			
CIFISP_CTRL	ISP Global Control Register	0000 _H	36-55
ISP Programming Registers, ISP Acquisition Registers			
CIFISP_ACQ_PROP	ISP Acquisition Properties Register	0004 _H	36-57
CIFISP_ACQ_H_OFFS	ISP Acquisition Horizontal Offset Register	0008 _H	36-59
CIFISP_ACQ_V_OFFS	ISP Acquisition Vertical Offset Register	000C _H	36-60
CIFISP_ACQ_H_SIZE	ISP Acquisition Horizontal Size Register	0010 _H	36-60
CIFISP_ACQ_V_SIZE	ISP Acquisition Vertical Size Register	0014 _H	36-61

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFISP_ACQ_NR_FRAMES	ISP Acquisition Number of Frames Register	0018 _H	36-62
ISP Programming Registers, ISP Output Control Registers			
CIFISP_OUT_H_OFFSETS	ISP Output Window Horizontal Offset Register	0194 _H	36-63
CIFISP_OUT_V_OFFSETS	ISP Output Window Vertical Offset Register	0198 _H	36-63
CIFISP_OUT_H_SIZE	ISP Output Horizontal Picture Size Register	019C _H	36-65
CIFISP_OUT_V_SIZE	ISP Output Vertical Picture Size Register	01A0 _H	36-66
CIFISP_FLAGS_SHD	ISP Shadow Flags Register	01A8 _H	36-66
CIFISP_OUT_H_OFFSETS_SHD	ISP Output Window Horizontal Offset Shadow Register	01AC _H	36-67
CIFISP_OUT_V_OFFSETS_SHD	ISP Output Window Vertical Offset Shadow Register	01B0 _H	36-68
CIFISP_OUT_H_SIZE_SHD	ISP Output Horizontal Picture Size Shadow Register	01B4 _H	36-70
CIFISP_OUT_V_SIZE_SHD	ISP Output Vertical Picture Size Shadow Register	01B8 _H	36-70
ISP Programming Registers, ISP Interrupt Control Registers			
CIFISP_IMSC	ISP Interrupt Mask Register	01BC _H	36-72
CIFISP_RIS	ISP Raw Interrupt Status Register	01C0 _H	36-74
CIFISP_MIS	ISP Masked Interrupt Status Register	01C4 _H	36-75
CIFISP_ICR	ISP Interrupt Clear Register	01C8 _H	36-76
CIFISP_ISR	ISP Interrupt Set Register	01CC _H	36-77
ISP Programming Registers, Miscellaneous ISP Registers			
CIFISP_ERR	ISP Error Register	023C _H	36-79
CIFISP_ERR_CLR	ISP Error Clear Register	0240 _H	36-80
CIFISP_FRAME_COUNT	ISP Frame Counter Register	0244 _H	36-81
Linear Downscaler Programming Registers, Linear Downscaler Configuration Registers			

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFLDS_CTRL	Linear Downscaler Control Register	0000 _H	36-82
CIFLDS_FAC	Linear Downscaler Factor Register	0004 _H	36-83
Memory Interface Programming Registers, Memory Interface Control Registers			
CIFMI_CTRL	Memory Interface Global Control Register	0000 _H	36-84
CIFMI_INIT	Memory Interface Control Register For Address Init And Skip Function Register	0004 _H	36-88
CIFMI_MP_Y_BASE_AD_INIT	Memory Interface Base Address For Main Picture Y Component, JPEG or RAW Data Register	0008 _H	36-90
CIFMI_MP_Y_SIZE_INIT	Memory Interface Size of main picture Y component, JPEG or RAW data Register	000C _H	36-91
CIFMI_MP_Y_OFFSET_CNTR_INIT	Memory Interface Offset Counter Init Value For Main Picture Y, JPEG or RAW Data Register	0010 _H	36-92
CIFMI_MP_Y_OFFSET_CNTR_START	Memory Interface Offset Counter Start Value For Main Picture Y, JPEG or RAW Data Register	0014 _H	36-93
CIFMI_MP_Y_IRQ_OFFSET_CNTR_INIT	Memory Interface Fill Level Interrupt Offset Value For Main Picture Y, JPEG or RAW Data Register	0018 _H	36-94
CIFMI_MP_CB_BASE_ADDR_INIT	Memory Interface Base Address For Main Picture Cb Component Ring Buffer Register	001C _H	36-95
CIFMI_MP_CB_SIZE_INIT	Memory Interface Size Of Main Picture Cb Component Ring Buffer Register	0020 _H	36-96
CIFMI_MP_CB_OFFSET_CNTR_INIT	Memory Interface Offset Counter Init Value For Main Picture Cb Component Ring Buffer Register	0024 _H	36-97
CIFMI_MP_CB_OFFSET_CNTR_START	Memory Interface Offset Counter Start Value For Main Picture Cb Component Ring Buffer Register	0028 _H	36-98

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFMI_MP_CR_BASE_AD_INIT	Memory Interface Base Address For Main Picture Cr Component Ring Buffer Register	002C _H	36-99
CIFMI_MP_CR_SIZE_INIT	Memory Interface Size Of Main Picture Cr Component Ring Buffer Register	0030 _H	36-100
CIFMI_MP_CR_OFFSET_COUNTER_INIT	Memory Interface Offset Counter Init value For Main Picture Cr Component Ring Buffer Register	0034 _H	36-101
CIFMI_MP_CR_OFFSET_COUNTER_START	Memory Interface Offset Counter Start Value For Main Picture Cr Component Ring Buffer Register	0038 _H	36-102
CIFMI_BYTE_CNT	Memory Interface Counter Value of JPEG or RAW Data Bytes Register	0070 _H	36-103
Memory Interface Programming Registers, Memory Interface Shadow Registers			
CIFMI_CTRL_SHD	Memory Interface Global Control Internal Shadow Register	0074 _H	36-104
CIFMI_MP_Y_BASE_ADDRESS_SHD	Memory Interface Base Address Shadow Register For Main Picture Y Component, JPEG Register	0078 _H	36-105
CIFMI_MP_Y_SIZE_SHD	Memory Interface Size Shadow Register of Main Picture Y Component, JPEG or RAW Data Register	007C _H	36-106
CIFMI_MP_Y_OFFSET_COUNTER_SHD	Memory Interface Current Offset Counter of Main Picture Y Component, JPEG or RAW Register	0080 _H	36-107
CIFMI_MP_Y_IRQ_OFFSET_SHD	Memory Interface Shadow Register of Fill Level Interrupt Offset Value For Main Picture Y Register	0084 _H	36-108
CIFMI_MP_CB_BASE_ADDRESS_SHD	Memory Interface Base Address Shadow Register For Main Picture Cb Component Ring Register	0088 _H	36-109

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFMI_MP_CB_SIZE_SHADOW	Memory Interface Size Shadow Register Of Main Picture Cb Component Ring Buffer Register	008C _H	36-110
CIFMI_MP_CB_OFFSET_COUNTER_SHADOW	Memory Interface Current Offset Counter Of Main Picture Cb Component Ring Buffer Register	0090 _H	36-111
CIFMI_MP_CR_BASE_ADDRESS_SHADOW	Memory Interface Base Address Shadow Register For Main Picture Cr Component Ring Register	0094 _H	36-112
CIFMI_MP_CR_SIZE_SHADOW	Memory Interface Size Shadow Register Of Main Picture Cr Component Ring Buffer Register	0098 _H	36-113
CIFMI_MP_CR_OFFSET_COUNTER_SHADOW	Memory Interface Current Offset Counter Of Main Picture Cr Component Ring Buffer Register	009C _H	36-114
Memory Interface Programming Registers, Memory Interface Interrupt Registers			
CIFMI_IMSC	MI Interrupt Mask ('1': interrupt active, '0': interrupt masked)	00F8 _H	36-115
CIFMI_RIS	MI Raw Interrupt Status Register	00FC _H	36-116
CIFMI_MIS	MI Masked Interrupt Status Register	0100 _H	36-118
CIFMI_ICR	MI Interrupt Clear Register	0104 _H	36-120
CIFMI_ISR	MI Interrupt Set Register	0108 _H	36-121
CIFMI_STATUS	MI Status Register	010C _H	36-122
CIFMI_STATUS_CLR	MI Status Clear Register	0110 _H	36-123
JPEG Encoder Programming Registers, JPEG Encoder Control Registers			
CIFJPE_GEN_HEADER	JPE Command To Start Stream Header Generation Register	0000 _H	36-125
CIFJPE_ENCODE	JPE Start Command To Start JFIF Stream Encoding Register	0004 _H	36-126
CIFJPE_INIT	JPE Automatic Configuration Update (INIT) Register	0008 _H	36-127
CIFJPE_Y_SCALE_EN	JPE Y Value Scaling Control Register	000C _H	36-128

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFJPE_CBCR_SCALE_EN	JPE Cb/Cr Value Scaling Control Register	0010 _H	36-129
CIFJPE_TABLE_FLUSH	JPE Header Generation Debug Register	0014 _H	36-130
CIFJPE_ENC_HSIZE	JPEG Codec Horizontal Image Size For Encoding Register	0018 _H	36-131
CIFJPE_ENC_VSIZE	JPEG Codec Vertical Image Size For Encoding Register	001C _H	36-131
CIFJPE_PIC_FORMAT	JPEG Picture Encoding Format Register	0020 _H	36-133
CIFJPE_RESTART_INTERVAL	JPE Restart Marker Insertion Register	0024 _H	36-133
CIFJPE_TQ_Y_SELECT	Q- table Selector 0, Quant. Table For Y Component	0028 _H	36-135
CIFJPE_TQ_U_SELECT	Q- table Selector 1, Quant. Table For U Component	002C _H	36-136
CIFJPE_TQ_V_SELECT	Q- table Selector 2, Quant. Table For V Component	0030 _H	36-137
CIFJPE_DC_TABLE_SELECT	JPE Huffman Table Selector For DC Values Register	0034 _H	36-138
CIFJPE_AC_TABLE_SELECT	JPE Huffman Table Selector For AC Values Register	0038 _H	36-139
CIFJPE_TABLE_DATA	JPE Table Programming Register	003C _H	36-140
CIFJPE_TABLE_ID	JPE Table Programming Select Register	0040 _H	36-140
CIFJPE_TAC0_LEN	JPE Huffman AC Table 0 Length Register	0044 _H	36-142
CIFJPE_TDC0_LEN	JPE Huffman DC Table 0 Length Register	0048 _H	36-142
CIFJPE_TAC1_LEN	JPE Huffman AC Table 1 Length Register	004C _H	36-144
CIFJPE_TDC1_LEN	JPE Huffman DC Table 1 Length Register	0050 _H	36-144

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFJPE_ENCODER_BUSY	JPE Encoder Status Flag Register	0058 _H	36-146
CIFJPE_HEADER_MODE	JPE Header Mode Definition Register	005C _H	36-146
CIFJPE_ENCODE_MODE	JPE Encode Mode Register	0060 _H	36-147
CIFJPE_DEBUG	JPE Debug Information Register	0064 _H	36-148
JPEG Encoder Programming Registers, JPEG Encoder Interrupt Registers			
CIFJPE_ERROR_IMR	JPE Error Interrupt Mask Register	0068 _H	36-150
CIFJPE_ERROR_RIS	JPE Error Raw Interrupt Status Register	006C _H	36-151
CIFJPE_ERROR_MIS	JPE Error Masked Interrupt Status Register	0070 _H	36-152
CIFJPE_ERROR_ISR	JPE Error Interrupt Set Register	0078 _H	36-153
CIFJPE_ERROR_ICR	JPE Error Interrupt Clear Register	0074 _H	36-154
CIFJPE_STATUS_IMR	JPEG Status Interrupt Mask Register	007C _H	36-155
CIFJPE_STATUS_RIS	JPEG Status Raw Interrupt Status Register	0080 _H	36-156
CIFJPE_STATUS_MIS	JPEG Status Masked Interrupt Status Register	0084 _H	36-157
CIFJPE_STATUS_ICR	JPEG Status Interrupt Clear Register	0088 _H	36-158
CIFJPE_STATUS_ISR	JPEG Status Interrupt Set Register	008C _H	36-159
Security Watchdog Programming Registers, Watchdog Configuration Registers			
CIFWD_CTRL	Watchdog Control Register	0000 _H	36-160
CIFWD_V_TIMEOUT	Watchdog Vertical Timeout Register	0004 _H	36-161
CIFWD_H_TIMEOUT	Watchdog Horizontal Timeout Register	0008 _H	36-162
Security Watchdog Programming Registers, Watchdog Interrupt Registers			
CIFWD_IMSC	Watchdog Interrupt Mask Register	000C _H	36-163
CIFWD_RIS	Watchdog Raw Interrupt Status Register	0010 _H	36-164

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFWD_MIS	Watchdog Masked Interrupt Status Register	0014 _H	36-165
CIFWD_ICR	Watchdog Interrupt Clear Register	0018 _H	36-166
CIFWD_ISR	Watchdog Interrupt Set Register	001C _H	36-167
ISP Image Stabilization Registers, Image Stabilization Control Registers			
CIFISPIS_CTRL	ISP Image Stabilization Control Register	0000 _H	36-168
CIFISPIS_RECENTER	ISP Image Stabilization Recenter Register	0004 _H	36-169
CIFISPIS_H_OFFS	ISP Image Stabilization Horizontal Offset Of Output Window Register	0008 _H	36-169
CIFISPIS_V_OFFS	ISP Image Stabilization Vertical Offset Of Output Window Register	000C _H	36-171
CIFISPIS_H_SIZE	ISP Image Stabilization Output Horizontal Picture Size Register	0010 _H	36-172
CIFISPIS_V_SIZE	ISP Image Stabilization Output Vertical Picture Size Register	0014 _H	36-173
CIFISPIS_MAX_DX	ISP Image Stabilization Maximum Horizontal Displacement Register	0018 _H	36-173
CIFISPIS_MAX_DY	ISP Image Stabilization Maximum Vertical Displacement Register	001C _H	36-175
CIFISPIS_DISPLACE	ISP Image Stabilization Camera Displacement Register	0020 _H	36-175
ISP Image Stabilization Registers, Image Stabilization Shadow Registers			
CIFISPIS_H_OFFS_SHD	ISP Image Current Horizontal Offset Of Output Window (Shadow Register) Register	0024 _H	36-176
CIFISPIS_V_OFFS_SHD	ISP Image Current Vertical Offset Of Output Window (Shadow Register) Register	0028 _H	36-177
CIFISPIS_H_SIZE_SHD	ISP Image Current Output Horizontal Picture Size (Shadow Register) Register	002C _H	36-178

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFISPIS_V_SIZE_SHD	ISP Image Current Output Vertical Picture Size (Shadow Register) Register	0030 _H	36-178
Extra Path Programming Registers, Extra Path Error Registers			
CIFMIEP_STA_ERR	Extra Path Error Register	0000 _H	36-180
CIFMIEP_STA_ERR_CLR	Extra Path Error Clear Register	0004 _H	36-182
Extra Path Programming Registers, Memory Interface Extra Path Interrupt Registers			
CIFMIEP_IMSC	MI Extra Path Interrupt Mask ('1': interrupt active, '0': interrupt masked)	0008 _H	36-183
CIFMIEP_RIS	MI Extra Path Raw Interrupt Status Register	000C _H	36-185
CIFMIEP_MIS	MI Extra Path Masked Interrupt Status Register	0010 _H	36-188
CIFMIEP_ICR	MI Extra Path Interrupt Clear Register	0014 _H	36-191
CIFMIEP_ISR	MI Extra Path Interrupt Set Register	0018 _H	36-193
Extra Path Programming Registers, Memory Interface Extra Path Control Registers			
CIFMIEP_z_CTRL (z=1-5)	Memory Interface Extra Path 1 Control Register	0100 _H	36-195
CIFMIEP_z_INIT (z=1-5)	Memory Interface Extra Path 1 Address Init and Skip Function Register	0104 _H	36-198
CIFMIEP_z_BASE_ADDR_INIT (z=1-5)	Memory Interface Base Address For Extra Path 1 Data Buffer Register	0108 _H	36-200
CIFMIEP_k_SIZE_INIT (k=1-5)	Memory Interface Size of Extra Path 1 Data Buffer Register	010C _H	36-201
CIFMIEP_z_OFFSETS_CNT_INIT (z=1-5)	Memory Interface Offset Counter Init Value For Extra Path 1 Data Buffer Register	0110 _H	36-202
CIFMIEP_z_OFFSETS_CNT_START (z=1-5)	Memory Interface Offset Counter Start Value For Extra Path 1 Data Buffer Register	0114 _H	36-203

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFMIEP_z_IRQ_OFFS_I NIT (z=1-5)	Memory Interface Fill Level Interrupt Offset Value For Extra Path 1 Data Buffer Register	0118 _H	36-204
CIFMIEP_z_CTRL (z=1-5)	Memory Interface Extra Path 2 Control Register	0200 _H	36-195
CIFMIEP_z_INIT (z=1-5)	Memory Interface Extra Path 2 Address Init and Skip Function Register	0204 _H	36-198
CIFMIEP_z_BASE_AD_I NIT (z=1-5)	Memory Interface Base Address For Extra Path 2 Data Buffer Register	0208 _H	36-200
CIFMIEP_k_SIZE_INIT (k=1-5)	Memory Interface Size of Extra Path 2 Data Buffer Register	020C _H	36-201
CIFMIEP_z_OFFS_CNT_ INIT (z=1-5)	Memory Interface Offset Counter Init Value For Extra Path 2 Data Buffer Register	0210 _H	36-202
CIFMIEP_z_OFFS_CNT_ START (z=1-5)	Memory Interface Offset Counter Start Value For Extra Path 2 Data Buffer Register	0214 _H	36-203
CIFMIEP_z_IRQ_OFFS_I NIT (z=1-5)	Memory Interface Fill Level Interrupt Offset Value For Extra Path 2 Data Buffer Register	0218 _H	36-204
CIFMIEP_z_CTRL (z=1-5)	Memory Interface Extra Path 3 Control Register	0300 _H	36-195
CIFMIEP_z_INIT (z=1-5)	Memory Interface Extra Path 3 Address Init and Skip Function Register	0304 _H	36-198
CIFMIEP_z_BASE_AD_I NIT (z=1-5)	Memory Interface Base Address For Extra Path 3 Data Buffer Register	0308 _H	36-200
CIFMIEP_k_SIZE_INIT (k=1-5)	Memory Interface Size of Extra Path 3 Data Buffer Register	030C _H	36-201
CIFMIEP_z_OFFS_CNT_ INIT (z=1-5)	Memory Interface Offset Counter Init Value For Extra Path 3 Data Buffer Register	0310 _H	36-202

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFMIEP_z_OFFS_CNT_START (z=1-5)	Memory Interface Offset Counter Start Value For Extra Path 3 Data Buffer Register	0314 _H	36-203
CIFMIEP_z_IRQ_OFFS_INIT (z=1-5)	Memory Interface Fill Level Interrupt Offset Value For Extra Path 3 Data Buffer Register	0318 _H	36-204
CIFMIEP_z_CTRL (z=1-5)	Memory Interface Extra Path 4 Control Register	0400 _H	36-195
CIFMIEP_z_INIT (z=1-5)	Memory Interface Extra Path 4 Address Init and Skip Function Register	0404 _H	36-198
CIFMIEP_z_BASE_AD_INIT (z=1-5)	Memory Interface Base Address For Extra Path 4 Data Buffer Register	0408 _H	36-200
CIFMIEP_k_SIZE_INIT (k=1-5)	Memory Interface Size of Extra Path 4 Data Buffer Register	040C _H	36-201
CIFMIEP_z_OFFS_CNT_INIT (z=1-5)	Memory Interface Offset Counter Init Value For Extra Path 4 Data Buffer Register	0410 _H	36-202
CIFMIEP_z_OFFS_CNT_START (z=1-5)	Memory Interface Offset Counter Start Value For Extra Path 4 Data Buffer Register	0414 _H	36-203
CIFMIEP_z_IRQ_OFFS_INIT (z=1-5)	Memory Interface Fill Level Interrupt Offset Value For Extra Path 4 Data Buffer Register	0418 _H	36-204
CIFMIEP_z_CTRL (z=1-5)	Memory Interface Extra Path 5 Control Register	0500 _H	36-195
CIFMIEP_z_INIT (z=1-5)	Memory Interface Extra Path 5 Address Init and Skip Function Register	0504 _H	36-198
CIFMIEP_z_BASE_AD_INIT (z=1-5)	Memory Interface Base Address For Extra Path 5 Data Buffer Register	0508 _H	36-200
CIFMIEP_k_SIZE_INIT (k=1-5)	Memory Interface Size of Extra Path 5 Data Buffer Register	050C _H	36-201

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFMIEP_z_OFFS_CNT_INIT (z=1-5)	Memory Interface Offset Counter Init Value For Extra Path 5 Data Buffer Register	0510 _H	36-202
CIFMIEP_z_OFFS_CNT_START (z=1-5)	Memory Interface Offset Counter Start Value For Extra Path 5 Data Buffer Register	0514 _H	36-203
CIFMIEP_z_IRQ_OFFS_INIT (z=1-5)	Memory Interface Fill Level Interrupt Offset Value For Extra Path 5 Data Buffer Register	0518 _H	36-204
Extra Path Programming Registers, Extra Path Memory Interface Shadow Registers			
CIFMIEP_z_CTRL_SHD (z=1-5)	Memory Interface Extra Path 1 Control Internal Shadow Register	011C _H	36-205
CIFMIEP_z_BASE_AD_SHD (z=1-5)	Memory Interface Extra Path 1 Control Internal Shadow Register	0120 _H	36-206
CIFMIEP_k_SIZE_SHD (k=1-5)	Memory Interface Extra Path 1 Base Address Shadow Register	0124 _H	36-206
CIFMIEP_z_OFFS_CNT_SHD (z=1-5)	Memory Interface Extra Path 1 Current Offset Counter Shadow Register	0128 _H	36-207
CIFMIEP_z_IRQ_OFFS_SHD (z=1-5)	Memory Interface Extra Path 1 Fill Level Interrupt Offset Value Shadow Register	012C _H	36-208
CIFMIEP_z_CTRL_SHD (z=1-5)	Memory Interface Extra Path 2 Control Internal Shadow Register	021C _H	36-205
CIFMIEP_z_BASE_AD_SHD (z=1-5)	Memory Interface Extra Path 2 Base Address Shadow Register	0220 _H	36-206
CIFMIEP_k_SIZE_SHD (k=1-5)	Memory Interface Extra Path 2 Size Shadow Register	0224 _H	36-206
CIFMIEP_z_OFFS_CNT_SHD (z=1-5)	Memory Interface Extra Path 2 Current Offset Counter Shadow Register	0228 _H	36-207
CIFMIEP_z_IRQ_OFFS_SHD (z=1-5)	Memory Interface Extra Path 2 Fill Level Interrupt Offset Value Shadow Register	022C _H	36-208

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFMIEP_z_CTRL_SHD (z=1-5)	Memory Interface Extra Path 3 Control Internal Shadow Register	031C _H	36-205
CIFMIEP_z_BASE_AD_SHD (z=1-5)	Memory Interface Extra Path 3 Base Address Shadow Register	0320 _H	36-206
CIFMIEP_k_SIZE_SHD (k=1-5)	Memory Interface Extra Path 3 Size Shadow Register	0324 _H	36-206
CIFMIEP_z_OFFS_CNT_SHD (z=1-5)	Memory Interface Extra Path 3 Current Offset Counter Shadow Register	0328 _H	36-207
CIFMIEP_z_IRQ_OFFS_SHD (z=1-5)	Memory Interface Extra Path 3 Fill Level Interrupt Offset Value Shadow Register	032C _H	36-208
CIFMIEP_z_CTRL_SHD (z=1-5)	Memory Interface Extra Path 4 Control Internal Shadow Register	041C _H	36-205
CIFMIEP_z_BASE_AD_SHD (z=1-5)	Memory Interface Extra Path 4 Base Address Shadow Register	0420 _H	36-206
CIFMIEP_k_SIZE_SHD (k=1-5)	Memory Interface Extra Path 4 Size Shadow Register	0424 _H	36-206
CIFMIEP_z_OFFS_CNT_SHD (z=1-5)	Memory Interface Extra Path 4 Current Offset Counter Shadow Register	0428 _H	36-207
CIFMIEP_z_IRQ_OFFS_SHD (z=1-5)	Memory Interface Extra Path 4 Fill Level Interrupt Offset Value Shadow Register	042C _H	36-208
CIFMIEP_z_CTRL_SHD (z=1-5)	Memory Interface Extra Path 5 Control Internal Shadow Register	051C _H	36-205
CIFMIEP_z_BASE_AD_SHD (z=1-5)	Memory Interface Extra Path 5 Base Address Shadow Register	0520 _H	36-206
CIFMIEP_k_SIZE_SHD (k=1-5)	Memory Interface Extra Path 5 Size Shadow Register	0524 _H	36-206
CIFMIEP_z_OFFS_CNT_SHD (z=1-5)	Memory Interface Extra Path 5 Current Offset Counter Shadow Register	0528 _H	36-207

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFMIEP_z_IRQ_OFFS_SHD (z=1-5)	Memory Interface Extra Path 5 Fill Level Interrupt Offset Value Shadow Register	052C _H	36-208
Extra Path Programming Registers, Extra Path Image Cropping Control Registers			
CIFEP_z_IC_CTRL (z=1-5)	Extra Path 1 Image Cropping Control Register	0000 _H	36-209
CIFEP_z_IC_RECENTER (z=1-5)	Extra Path 1 Image Cropping Recenter Register	0004 _H	36-210
CIFEP_z_IC_H_OFFS (z=1-5)	Extra Path 1 Image Cropping Horizontal Offset of Output Window Register	0008 _H	36-210
CIFEP_z_IC_V_OFFS (z=1-5)	Extra Path 1 Image Cropping Vertical Offset Of Output Window Register	000C _H	36-212
CIFEP_z_IC_H_SIZE (z=1-5)	Extra Path 1 Image Cropping Output Horizontal Picture Size Register	0010 _H	36-213
CIFEP_k_IC_V_SIZE (k=1-5)	Extra Path 1 Image Cropping Output Vertical Picture Size Register	0014 _H	36-214
CIFEP_k_IC_MAX_DX (k=1-5)	Extra Path 1 Image Cropping Maximum Horizontal Displacement Register	0018 _H	36-214
CIFEP_z_IC_MAX_DY (z=1-5)	Extra Path 1 Image Cropping Maximum Vertical Displacement Register	001C _H	36-216
CIFEP_z_IC_DISPLACE (z=1-5)	Extra Path 1 Image Cropping Camera Displacement Register	0020 _H	36-216
CIFEP_z_IC_CTRL (z=1-5)	Extra Path 2 Image Cropping Control Register	0100 _H	36-209
CIFEP_z_IC_RECENTER (z=1-5)	Extra Path 2 Image Cropping Recenter Register	0104 _H	36-210
CIFEP_z_IC_H_OFFS (z=1-5)	Extra Path 2 Image Cropping Horizontal Offset Register	0108 _H	36-210
CIFEP_z_IC_V_OFFS (z=1-5)	Extra Path 2 Image Cropping Vertical Offset Register	010C _H	36-212

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFEP_z_IC_H_SIZE (z=1-5)	Extra Path 2 Image Cropping Horizontal Picture Size Register	0110 _H	36-213
CIFEP_k_IC_V_SIZE (k=1-5)	Extra Path 2 Image Cropping Vertical Picture Size Register	0114 _H	36-214
CIFEP_k_IC_MAX_DX (k=1-5)	Extra Path 2 Image Cropping Maximum Horizontal Displacement Register	0118 _H	36-214
CIFEP_z_IC_MAX_DY (z=1-5)	Extra Path 2 Image Cropping Maximum Vertical Displacement Register	011C _H	36-216
CIFEP_z_IC_DISPLACE (z=1-5)	Extra Path 2 Image Cropping Displacement Register	0120 _H	36-216
CIFEP_z_IC_CTRL (z=1-5)	Extra Path 3 Image Cropping Control Register	0200 _H	36-209
CIFEP_z_IC_RECENTER (z=1-5)	Extra Path 3 Image Cropping Recenter Register	0204 _H	36-210
CIFEP_z_IC_H_OFFS (z=1-5)	Extra Path 3 Image Cropping Horizontal Offset Register	0208 _H	36-210
CIFEP_z_IC_V_OFFS (z=1-5)	Extra Path 3 Image Cropping Vertical Offset Register	020C _H	36-212
CIFEP_z_IC_H_SIZE (z=1-5)	Extra Path 3 Image Cropping Horizontal Picture Size Register	0210 _H	36-213
CIFEP_k_IC_V_SIZE (k=1-5)	Extra Path 3 Image Cropping Vertical Picture Size Register	0214 _H	36-214
CIFEP_k_IC_MAX_DX (k=1-5)	Extra Path 3 Image Cropping Maximum Horizontal Displacement Register	0218 _H	36-214
CIFEP_z_IC_MAX_DY (z=1-5)	Extra Path 3 Image Cropping Maximum Vertical Displacement Register	021C _H	36-216
CIFEP_z_IC_DISPLACE (z=1-5)	Extra Path 3 Image Cropping Displacement Register	0220 _H	36-216
CIFEP_z_IC_CTRL (z=1-5)	Extra Path 4 Image Cropping Control Register	0300 _H	36-209

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFEP_z_IC_RECENTER (z=1-5)	Extra Path 4 Image Cropping Recenter Register	0304 _H	36-210
CIFEP_z_IC_H_OFFSETS (z=1-5)	Extra Path 4 Image Cropping Horizontal Offset Register	0308 _H	36-210
CIFEP_z_IC_V_OFFSETS (z=1-5)	Extra Path 4 Image Cropping Vertical Offset Register	030C _H	36-212
CIFEP_z_IC_H_SIZE (z=1-5)	Extra Path 4 Image Cropping Horizontal Picture Size Register	0310 _H	36-213
CIFEP_k_IC_V_SIZE (k=1-5)	Extra Path 4 Image Cropping Vertical Picture Size Register	0314 _H	36-214
CIFEP_k_IC_MAX_DX (k=1-5)	Extra Path 4 Image Cropping Maximum Horizontal Displacement Register	0318 _H	36-214
CIFEP_z_IC_MAX_DY (z=1-5)	Extra Path 4 Image Cropping Maximum Vertical Displacement Register	031C _H	36-216
CIFEP_z_IC_DISPLACE (z=1-5)	Extra Path 4 Image Cropping Displacement Register	0320 _H	36-216
CIFEP_z_IC_CTRL (z=1-5)	Extra Path 5 Image Cropping Control Register	0400 _H	36-209
CIFEP_z_IC_RECENTER (z=1-5)	Extra Path 5 Image Cropping Recenter Register	0404 _H	36-210
CIFEP_z_IC_H_OFFSETS (z=1-5)	Extra Path 5 Image Cropping Horizontal Offset Register	0408 _H	36-210
CIFEP_z_IC_V_OFFSETS (z=1-5)	Extra Path 5 Image Cropping Vertical Offset Register	040C _H	36-212
CIFEP_z_IC_H_SIZE (z=1-5)	Extra Path 5 Image Cropping Horizontal Picture Size Register	0410 _H	36-213
CIFEP_k_IC_V_SIZE (k=1-5)	Extra Path 5 Image Cropping Vertical Picture Size Register	0414 _H	36-214
CIFEP_k_IC_MAX_DX (k=1-5)	Extra Path 5 Image Cropping Maximum Horizontal Displacement Register	0418 _H	36-214

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFEP_z_IC_MAX_DY (z=1-5)	Extra Path 5 Image Cropping Maximum Vertical Displacement Register	041C _H	36-216
CIFEP_z_IC_DISPLACE (z=1-5)	Extra Path 5 Image Cropping Displacement Register	0420 _H	36-216
Extra Path Programming Registers, Extra Path Image Cropping Shadow Registers			
CIFEP_k_IC_H_OFFS_S HD (k=1-5)	Extra Path 1 Image Current Horizontal Offset Of Output Window Shadow Register	0024 _H	36-217
CIFEP_k_IC_V_OFFS_S HD (k=1-5)	Extra Path 1 Image Current Vertical Offset Of Output Window Shadow Register	0028 _H	36-218
CIFEP_k_IC_H_SIZE_S HD (k=1-5)	Extra Path 1 Image Current Output Horizontal Picture Size Shadow Register	002C _H	36-218
CIFEP_k_IC_V_SIZE_SHADOW (k=1-5)	Extra Path 1 Image Current Output Vertical Picture Size Shadow Register	0030 _H	36-219
CIFEP_k_IC_H_OFFS_S HD (k=1-5)	Extra Path 2 Image Cropping Current Horizontal Offset of Output Window Shadow Register	0124 _H	36-217
CIFEP_k_IC_V_OFFS_S HD (k=1-5)	Extra Path 2 Image Cropping Current Vertical Offset of Output Window(Shadow Register	0128 _H	36-218
CIFEP_k_IC_H_SIZE_S HD (k=1-5)	Extra Path 2 Image Cropping Current Output Horizontal Picture Size Shadow Register	012C _H	36-218
CIFEP_k_IC_V_SIZE_SHADOW (k=1-5)	Extra Path 2 Image Cropping Current Output Vertical Picture Size Shadow Register	0130 _H	36-219
CIFEP_k_IC_H_OFFS_S HD (k=1-5)	Extra Path 3 Image Cropping Current Horizontal Offset of Output Window Shadow Register	0224 _H	36-217

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFEP_k_IC_V_OFFS_S HD (k=1-5)	Extra Path 3 Image Cropping Current Horizontal Offset of Output Window Shadow Register	0228 _H	36-218
CIFEP_k_IC_H_SIZE_S HD (k=1-5)	Extra Path 3 Image Cropping Current Output Horizontal Picture Size Shadow Register	022C _H	36-218
CIFEP_k_IC_V_SIZE_SHADOW (k=1-5)	Extra Path 3 Image Cropping Current Output Vertical Picture Size Shadow Register	0230 _H	36-219
CIFEP_k_IC_H_OFFS_S HD (k=1-5)	Extra Path 4 Image Cropping Current Horizontal Offset of Output Window Shadow Register	0324 _H	36-217
CIFEP_k_IC_V_OFFS_S HD (k=1-5)	Extra Path 4 Image Cropping Current Vertical Offset of Output Window Shadow Register	0328 _H	36-218
CIFEP_k_IC_H_SIZE_S HD (k=1-5)	Extra Path 4 Image Cropping Current Output Horizontal Picture Size Shadow Register	032C _H	36-218
CIFEP_k_IC_V_SIZE_SHADOW (k=1-5)	Extra Path 4 Image Cropping Current Output Vertical Picture Size Shadow Register	0330 _H	36-219
CIFEP_k_IC_H_OFFS_S HD (k=1-5)	Extra Path 5 Image Cropping Current Horizontal Offset of Output Window Shadow Register	0424 _H	36-217
CIFEP_k_IC_V_OFFS_S HD (k=1-5)	Extra Path 5 Image Cropping Current Vertical Offset of Output Window Shadow Register	0428 _H	36-218
CIFEP_k_IC_H_SIZE_S HD (k=1-5)	Extra Path 5 Image Cropping Current Output Horizontal Picture Size Shadow Register	042C _H	36-218
CIFEP_k_IC_V_SIZE_SHADOW (k=1-5)	Extra Path 5 Image Cropping Current Output Vertical Picture Size Shadow Register	0430 _H	36-219
Debug Path Programming Registers, Debug Path Control Registers			
CIFDP_CTRL	Debug Path Control Register	0000 _H	36-221

Table 36-2 Registers Overview (cont'd)(in alphabetical order)

Register Short Name	Register Long Name	Offset Address	Page Number
CIFDP_PDIV_CTRL	Debug Path Predivider Control Register	0004 _H	36-224
Debug Path Programming Registers, Debug Path Status Registers			
CIFDP_FLC_STAT	Debug Path Frame/Line Counter Status Register	0008 _H	36-225
CIFDP_PDIV_STAT	Debug Path Predivider Counter Status Register	000C _H	36-226
CIFDP_TSC_STAT	Debug Path Timestamp Counter Status Register	0010 _H	36-226
Debug Path Programming Registers, Debug Path User Defined Symbols Registers			
CIFDP_UDS_x (x=1-8)	Debug Path User Defined Symbol 1 Register	0014 _H	36-227
CIFDP_UDS_x (x=1-8)	Debug Path User Defined Symbol 2 Register	0018 _H	36-227
CIFDP_UDS_x (x=1-8)	Debug Path User Defined Symbol 3 Register	001C _H	36-227
CIFDP_UDS_x (x=1-8)	Debug Path User Defined Symbol 4 Register	0020 _H	36-227
CIFDP_UDS_x (x=1-8)	Debug Path User Defined Symbol 5 Register	0024 _H	36-227
CIFDP_UDS_x (x=1-8)	Debug Path User Defined Symbol 6 Register	0028 _H	36-227
CIFDP_UDS_x (x=1-8)	Debug Path User Defined Symbol 7 Register	002C _H	36-227
CIFDP_UDS_x (x=1-8)	Debug Path User Defined Symbol 8 Register	0030 _H	36-227

The register is addressed wordwise.

Table 36-3 Registers Access Types

Mode	Symbol	Description Hardware (HW)	Description Software (SW)
Basic Access Types			
read/write	rw	Register is used as input for the HW	Register is read and writable by SW
read	r	Register is written by HW (register between input and output -> one cycle delay)	Value written by SW is ignored by HW; that is, SW may write any value to this field without affecting HW behavior
write	w	Register is written by software and affects hardware behavior with every write by software.	Register is writable by SW. When read, the register does not return the value that has been written previously, but some constant value instead.

36.1.3.2 CIF Control Registers

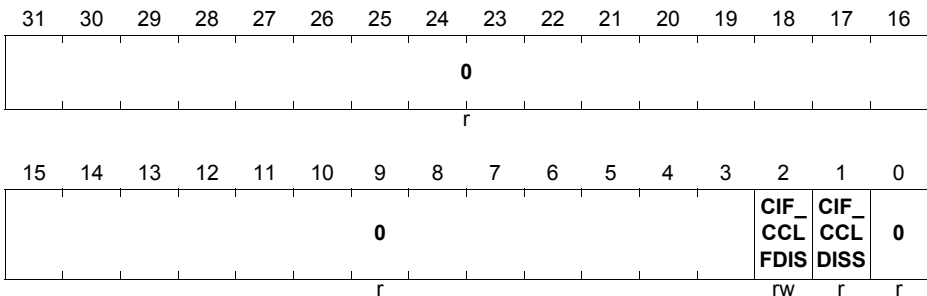
The address of each CIF control register is evaluated as CIF_CONTROL_BASE + Offset.

CIF Clock Control Registers

CIF Clock Control Register

CIF_CCL

Clock Control Register (0000_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
CIF_CCLFDS	2	rw	Clock Control Logic disable 0 _B processing/cfg-clocks for all CIF sub modules enabled 1 _B processing/cfg-clocks for all CIF sub modules disabled w/o access to ID and CIF_CCL register Reset: 0 _B
CIF_CCLDISS	1	r	Status of cif_ccl[2] bit (copy of cif_ccl[2]) Reset: 0 _B
0	31:3, 0	r	Reserved Read as 0, should be written with 0.

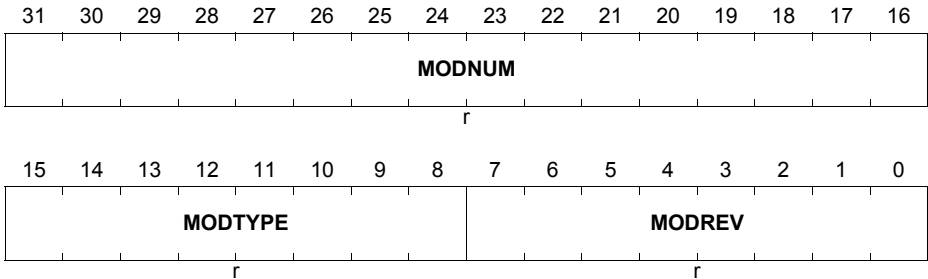
CIF Custom Registers

CIF Revision Identification Register

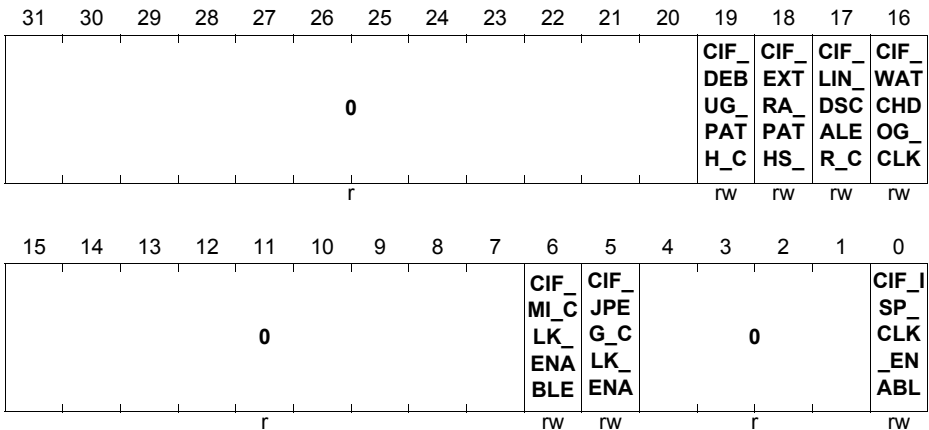
CIF_ID

CIF Revision Identification Register (0008_H)

Reset Value: 00B3 C002_H



Field	Bits	Type	Description
MODNUM	31:16	r	Module Number Value This bit field defines the module as a CIF. Reset: B3 _H
MODTYPE	15:8	r	Module Type This bit field defines the module as a 32-bit module. Reset: C0 _H
MODREV	7:0	r	Module Revision Number This bit field defines the module revision number. Reset: 02 _H

CIF Internal Control Registers
CIF Internal Clock Control Register
CIF_ICCL
CIF Internal Clock Control Register (0010_H)
Reset Value: 000F 0061_H


Field	Bits	Type	Description
CIF_DEBUG_PATH_CLK_EN	19	rw	Debug Path clock enable 0 _B power safe 1 _B processing mode Reset: 1 _B
CIF_EXTRA_PATHS_CLK_ENABLE	18	rw	Extra Paths clock enable 0 _B power safe 1 _B processing mode Reset: 1 _B
CIF_LIN_DSCALER_CLK_ENABLE	17	rw	Linear Downscaler clock enable 0 _B power safe 1 _B processing mode Reset: 1 _B
CIF_WATCHDOG_CLK_ENABLE	16	rw	Security Watchdog clock enable 0 _B power safe 1 _B processing mode Reset: 1 _B

Field	Bits	Type	Description
CIF_MI_CLK_ENABLE	6	rw	Memory interface clock enable 0 _B power safe 1 _B processing mode Reset: 1 _B
CIF_JPEG_CLK_ENABLE	5	rw	JPEG encoder clock enable 0 _B power safe 1 _B processing mode Reset: 1 _B
CIF_ISP_CLK_ENABLE	0	rw	ISP processing clock enable 0 _B power safe 1 _B processing mode Reset: 1 _B
0	31:20, 15:7, 4:1	r	Reserved Read as 0, should be written with 0.

Internal Reset Control Register

CIF_IRCL

CIF Internal Reset Control Register (0014_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
0												CIF_DEB	CIF_EXT	CIF_LIN	CIF_WAT		
r												UG_PAT	RA_PAT	DSC_ALE	CHD_OG		
r												H_R	HS_R	R_R	RST		
r												rw	rw	rw	rw		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
0											CIF_GLO	CIF_MI_S	CIF_JPE		CIF_YCS		CIF_I
r											BAL_RS	OFT_RS	G_S	0	_SO	0	SP_SOF
r											T	T	OFT_RS		FT_RST		T_RST
r											rw	rw	rw	r	rw	r	rw

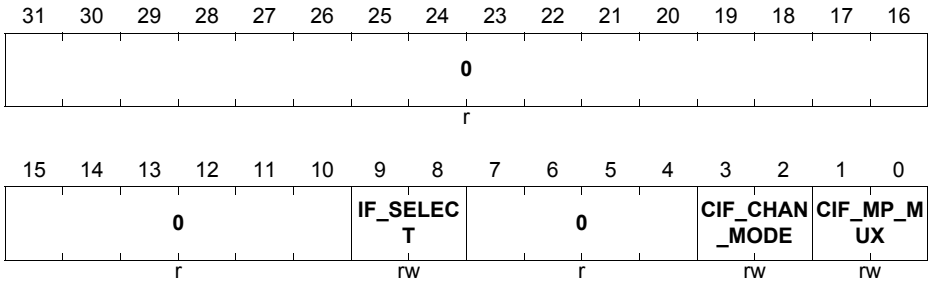
Field	Bits	Type	Description
CIF_DEBUG_PATH_RST	19	rw	Debug Path software reset 0 _B processing mode 1 _B reset state
CIF_EXTRA_PATHS_RST	18	rw	Extra Paths software reset 0 _B processing mode 1 _B reset state
CIF_LIN_DSCALE_R_RST	17	rw	Linear Downscaler software reset 0 _B processing mode 1 _B reset state
CIF_WATCHDOG_RST	16	rw	Security Watchdog software reset 0 _B processing mode 1 _B reset state
CIF_GLOBAL_RST	7	rw	Soft reset of entire CIF 0 _B processing mode 1 _B reset state
CIF_MI_SOFT_RST	6	rw	Memory interface software reset 0 _B processing mode 1 _B reset state
CIF_JPEG_SOFT_RST	5	rw	JPEG encoder software reset 0 _B processing mode 1 _B reset state
CIF_YCS_SOFT_RST	2	rw	Y/C splitter software reset 0 _B processing mode 1 _B reset state
CIF_ISP_SOFT_RST	0	rw	Isp software reset 0 _B processing mode 1 _B reset state
0	31:20, 15:8, 4:3, 1	r	Reserved Read as 0, should be written with 0.

CIF Data Path Control Register

For configuration of the Extra Data Paths see [“Extra Path Programming Registers” on Page 36-180](#).

CIF_DPCL

CIF Data Path Control Register (0018_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
IF_SELECT	9:8	rw	Selects input interface 0 _D parallel interface Other values are reserved. Reset: 0 _D
CIF_CHAN_MODE	3:2	rw	Y/C splitter channel mode 0 _D disabled 1 _D main path and RAW data mode Other values are reserved. Reset: 0 _D
CIF_MP_MUX	1:0	rw	Data path selector for main path 00 _B disabled 01 _B data to MI uncompressed 10 _B data to JPEG encoder Other values are reserved. Reset: 00 _B
0	31:10, 7:4	r	Reserved Read as 0, should be written with 0.

36.1.3.3 ISP Programming Registers

The address of each CIF ISP programming register is evaluated as CIF_ISP_BASE + Offset.

ISP Control Registers

ISP Global Control Register

CIFISP_CTRL

ISP Global Control Register (0000_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	ISP_CSM_C_RANGE	ISP_CSM_Y_RANGE	0	ISP_GEN_CF_G_UPD	ISP_CFG_UPD			0			ISP_INFO_RM_ENABLE	ISP_MODE		ISP_ENABLE	
r	rw	rw	r	w	w			r			rw	rw		rw	

Field	Bits	Type	Description
ISP_CSM_C_RANGE	14	rw	Color Space Matrix chrominance clipping range for ISP output 0 _B CbCr range 16..240 according to ITU-R BT.601standard 1 _B full UV range 0..255
ISP_CSM_Y_RANGE	13	rw	Color Space Matrix luminance clipping range for ISP output 0 _B Y range 16..235 according to ITU-R BT.601standard 1 _B full Y range 0..255

Field	Bits	Type	Description
ISP_GEN_CFG_UPD	10	w	ISP Generate Config Update 1 _B generate frame synchronous configuration signal at the output of ISP for shadow registers of the following processing modules 0 _B no effect Reset: 0 _B
ISP_CFG_UPD	9	w	ISP Config Update 1 _B immediately configures (update) shadow registers 0 _B no effect Reset: 0 _B
ISP_INFORM_ENABLE	4	rw	ISP Input Formater Enable Controls input formater frame synchronously, if <code>isp_gen_cfg_upd</code> is used to activate this bit. For immediate update <code>isp_cfg_upd</code> must be used 1 _B Input Formater is active 0 _B Input Formater is in deactivated Reset: 0 _B
ISP_MODE	3:1	rw	ISP Mode Unused values are reserved. 000 _B RAW picture 001 _B ITU-R BT.656 (YUV with embedded sync) 010 _B ITU-R BT.601 (YUV input with H and Vsync signals) 100 _B datamode (ISP bypass, sync signals interpreted as data enable) 110 _B RAW picture mode with ITU-R BT.656 synchronization Reset: 000 _B
ISP_ENABLE	0	rw	ISP output enable Controls output formater frame synchronously, if <code>isp_gen_cfg_upd</code> is used to activate this bit. For immediate update <code>isp_cfg_upd</code> must be used. 0 _B ISP output OFF 1 _B ISP output ON Reset: 0 _B

Field	Bits	Type	Description
0	31:15, 12:11, 8:5	r	Reserved Read as 0, should be written with 0.

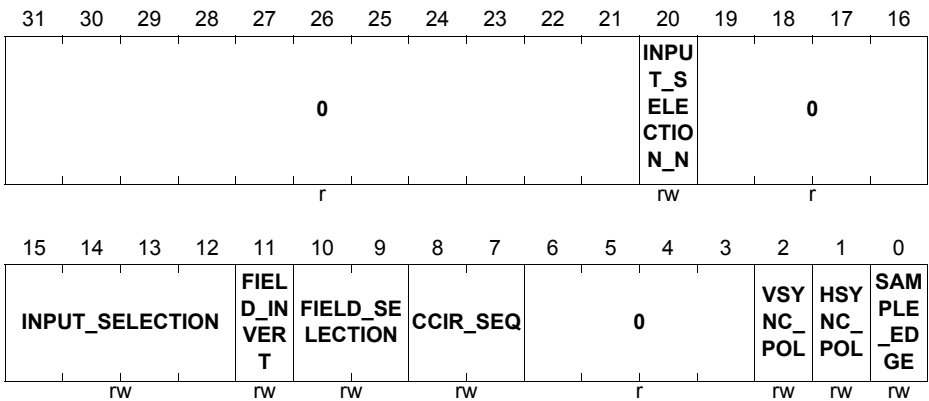
ISP Acquisition Registers

ISP Acquisition Properties Register

CIFISP_ACQ_PROP

ISP Acquisition Properties Register (0004_H)

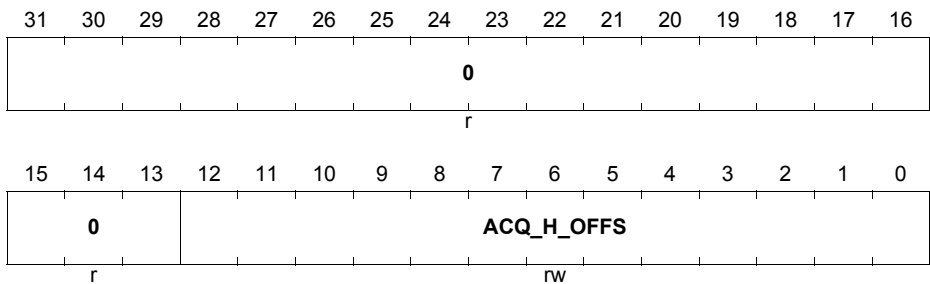
Reset Value: 0000 0000_H



Field	Bits	Type	Description
INPUT_SELECTION_NO_APP	20	rw	Input Selection No Append This field controls if the input of an external Interface gets appended with zeroes or it MSBs or shifted right to be LSB aligned. 0 _B append enabled 1 _B append disabled

Field	Bits	Type	Description
INPUT_SELECTIO N	15:12	rw	Input Selection 0000 _B 8-bit external Interface if enabled append 8 MSBs as LSBs 0001 _B 8-bit external Interface if enabled append 8 zeroes as LSBs 0010 _B 10-bit external Interface if enabled append 6 MSBs as LSBs 0011 _B 10-bit external Interface if enabled append 6 zeroes as LSBs 0100 _B 12-bit external Interface if enabled append 4 MSBs as LSBs 0101 _B 12-bit external Interface if enabled append 4 zeroes as LSBs 0110 _B 14-bit external Interface if enabled append 2 MSBs as LSBs 0111 _B 14-bit external Interface if enabled append 2 zeroes as LSBs 1000 _B 16-bit external Interface Reset: 0000 _B
FIELD_INVERT	11	rw	Field Invert If set to 1 _B the field-id will be inverted (even fields will become odd ones and vice versa). Reset: 0 _B
FIELD_SELECTIO N	10:9	rw	Field Selection 00 _B sample all fields (don't care about fields) 01 _B sample only even fields 10 _B sample only odd fields 11 _B reserved Reset: 00 _B
CCIR_SEQ	8:7	rw	CCIR Sequence This bit field defines the output sequence of the Acquisition Format block. 00 _B YCbYCr 01 _B YCrYCb 10 _B CbYCrY Reset: 00 _B

Field	Bits	Type	Description
VSYNC_POL	2	rw	Vertical sync polarity 0 _B high active 1 _B low active Reset: 0 _B
HSYNC_POL	1	rw	Horizontal sync polarity 0 _B high active 1 _B low active Reset: 0 _B
SAMPLE_EDGE	0	rw	Sample Edge 0 _B negative edge sampling 1 _B positive edge sampling Reset: 0 _B
0	31:21, 19:16, 6:3	r	Reserved Read as 0, should be written with 0.

ISP Acquisition Horizontal Offset Register
CIFISP_ACQ_H_OFFS
ISP Acquisition Horizontal Offset Register
(0008_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
ACQ_H_OF FS	12:0	rw	Horizontal sample offset In sensor data samples (yuv: 4 samples=2pix) Reset: 0000 _H
0	31:13	r	Reserved Read as 0, should be written with 0.

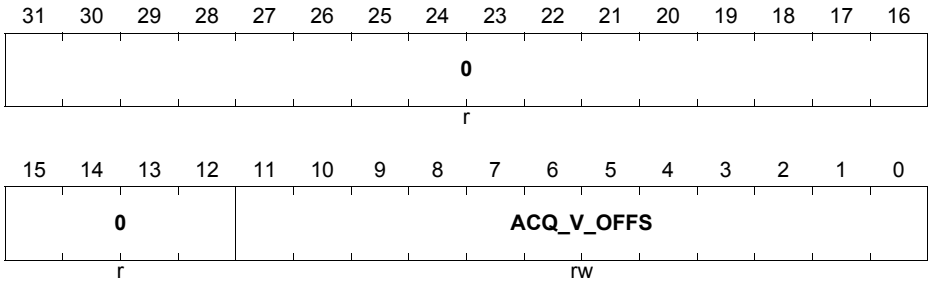
ISP Acquisition Vertical Offset Register

CIFISP_ACQ_V_OFFS

ISP Acquisition Vertical Offset Register

(000C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
ACQ_V_OF FS	11:0	rw	Vertical sample offset In lines Reset: 000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

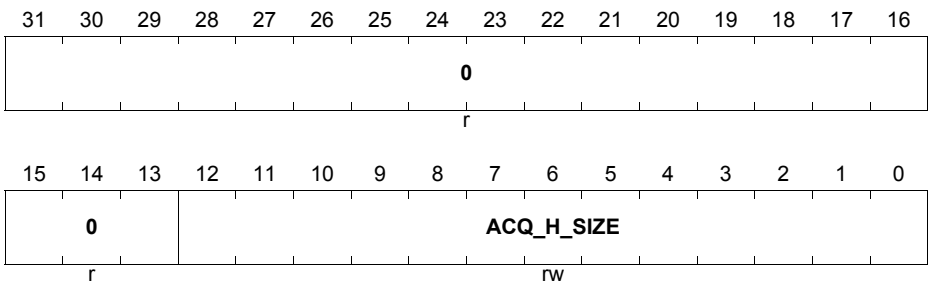
ISP Acquisition Horizontal Size Register

CIFISP_ACQ_H_SIZE

ISP Acquisition Horizontal Size Register

(0010H)

Reset Value: 0000 0A28_H



Field	Bits	Type	Description
ACQ_H_SIZE E	12:0	rw	Horizontal sample size In sensor data samples YUV input: 2 samples=1 pixel, else 1 sample=1 pixel; So in YUV mode ACQ_H_SIZE must be twice as large as horizontal image size Horizontal image size must always be even except in RAW picture mode; if an odd size is programmed the value will be truncated to even size Reset: 0A28 _H
0	31:13	r	Reserved Read as 0, should be written with 0.

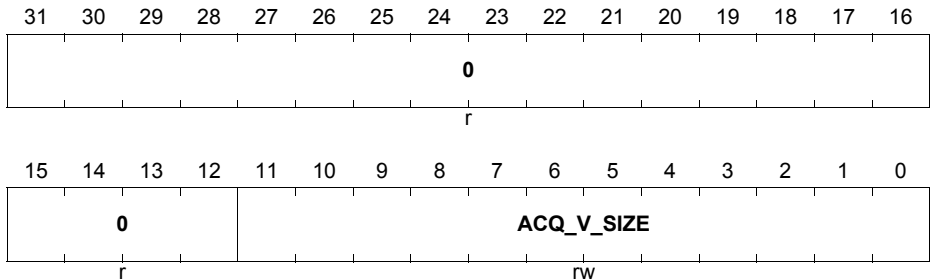
ISP Acquisition Vertical Size Register

CIFISP_ACQ_V_SIZE

ISP Acquisition Vertical Size Register

(0014_H)

Reset Value: 0000 0800_H



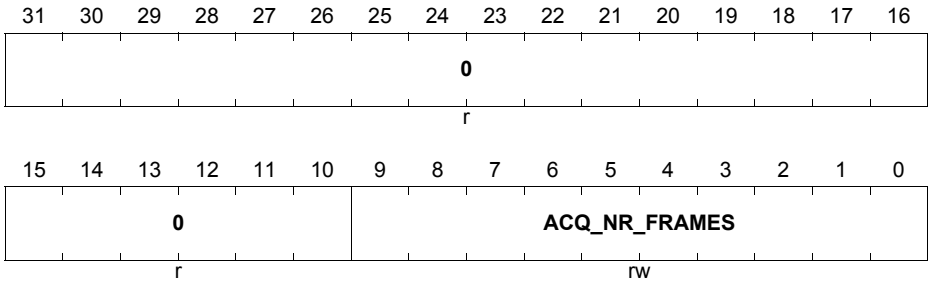
Field	Bits	Type	Description
ACQ_V_SIZE E	11:0	rw	Vertical sample size In lines Reset: 800 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

ISP Acquisition Number of Frames Register

CIFISP_ACQ_NR_FRAMES

**ISP Acquisition Number of Frames Register
(0018_H)**

Reset Value: 0000 0000_H



Field	Bits	Type	Description
ACQ_NR_FRAME S	9:0	rw	Number of Input Frames To be sampled (0 = continuous) Reset: 000 _H
0	31:10	r	Reserved Read as 0, should be written with 0.

ISP Output Control Registers

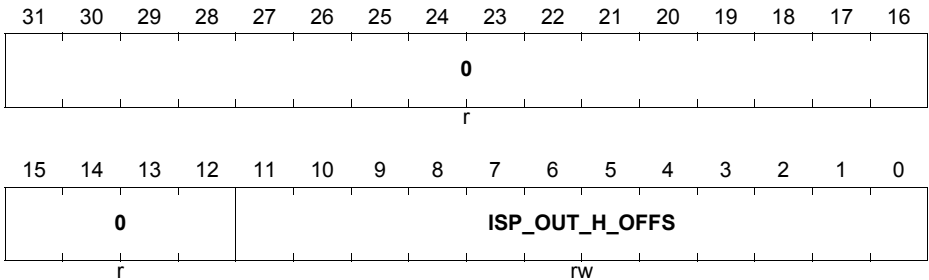
ISP Output Window Horizontal Offset Register

CIFISP_OUT_H_OFFS

ISP Output Window Horizontal Offset Register

(0194_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
ISP_OUT_H_OFFS	11:0	rw	Horizontal Picture Offset Unit = pixel Reset: 000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

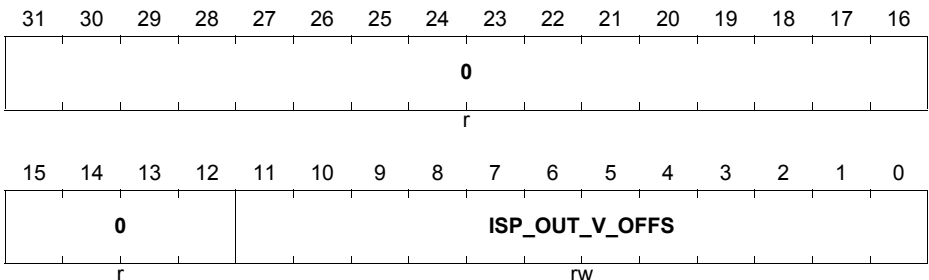
ISP Output Window Vertical Offset Register

CIFISP_OUT_V_OFFS

ISP Output Window Vertical Offset Register

(0198_H)

Reset Value: 0000 0000_H



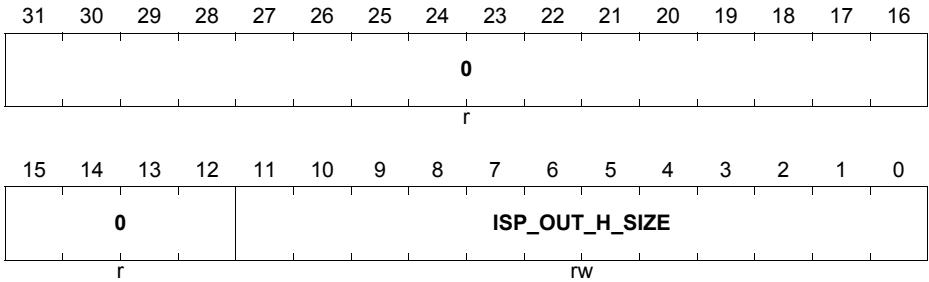
Field	Bits	Type	Description
ISP_OUT_V_OFFS	11:0	rw	Vertical Picture Offset Unit = lines Reset: 000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

ISP Output Horizontal Picture Size Register

CIFISP_OUT_H_SIZE

**ISP Output Horizontal Picture Size Register
(019C_H)**

Reset Value: 0000 0A28_H



Field	Bits	Type	Description
ISP_OUT_H_SIZE	11:0	rw	Horizontal picture size in pixel. If ISP_MODE is set to ... <ul style="list-style-type: none"> • 001_B : ITU-R BT.656 YUV • 010_B : ITU-R BT.601 YUV ... only even numbers are accepted, because complete quadruples of YUYV(YCbYCr) are needed for the 422 output. (if an odd size is programmed the value will be truncated to even size) Reset: 0A28 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

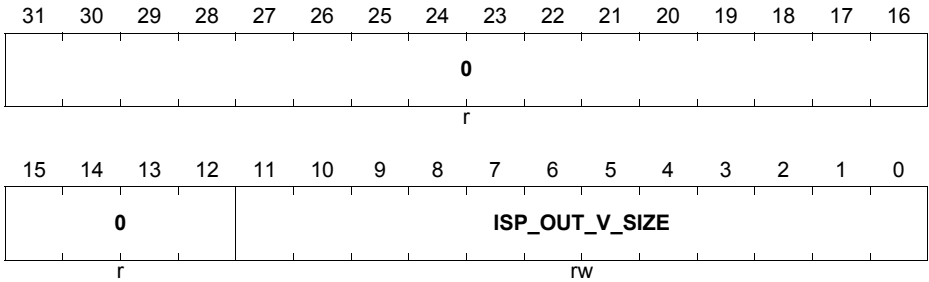
ISP Output Vertical Picture Size Register

CIFISP_OUT_V_SIZE

ISP Output Vertical Picture Size Register

(01A0_H)

Reset Value: 0000 0800_H



Field	Bits	Type	Description
ISP_OUT_V_SIZE	11:0	rw	Vertical picture size In lines Reset: 800 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

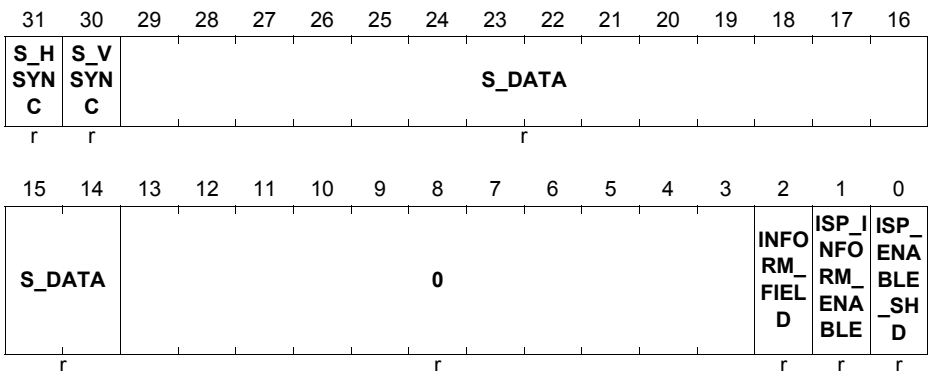
ISP Shadow Flags Register

CIFISP_FLAGS_SHD

ISP Shadow Flags Register

(01A8_H)

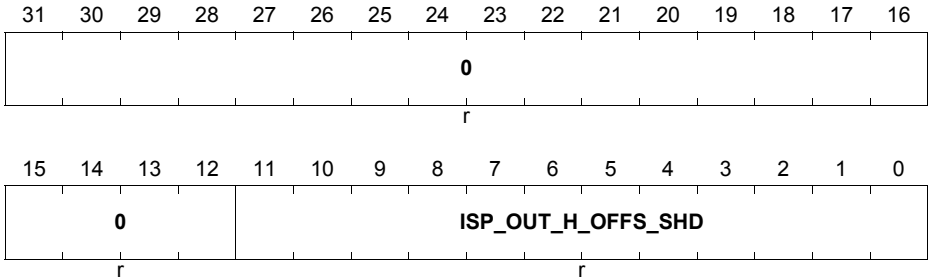
Reset Value: FFFF C000_H



Field	Bits	Type	Description
S_HSYNC	31	r	State of ISP input port s_hsync For test purposes Reset: 0 _B
S_VSYNC	30	r	State of ISP input port s_vsync For test purposes Reset: 0 _B
S_DATA	29:14	r	State of ISP input port s_data For test purposes Reset: 000 _H
INFORM_FIELD	2	r	Current field information 0 _B = odd 1 _B = even Reset: 0 _B
ISP_INFORM_ENAB LE_SHD	1	r	Input formatter enable shadow register Reset: 0 _B
ISP_ENABLE_SHD	0	r	ISP enable shadow register 0 _B no output of data 1 _B ISP currently outputs data Reset: 0 _B
0	13:3	r	Reserved Read as 0.

ISP Output Window Horizontal Offset Register

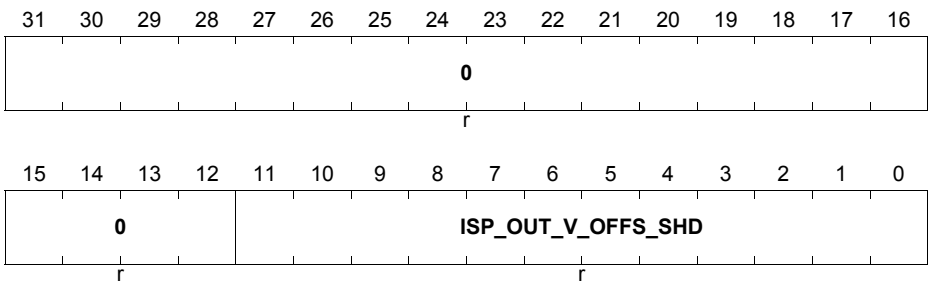
Current horizontal offset of output window (shadow register).

CIFISP_OUT_H_OFFS_SHD
ISP Output Window Horizontal Offset Shadow Register
(01AC_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
ISP_OUT_H_OFFS_SHD	11:0	r	Current horizontal picture offset In lines Reset: 000 _H
0	31:12	r	Reserved Read as 0.

ISP Output Window Vertical Offset Register

Current vertical offset of output window (shadow register).

CIFISP_OUT_V_OFFS_SHD
ISP Output Window Vertical Offset Shadow Register
(01B0_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
ISP_OUT_V_OFFSET_SH D	11:0	r	Current vertical picture offset In lines Reset: 000 _H
0	31:12	r	Reserved Read as 0.

ISP Output Horizontal Picture Size Shadow Register

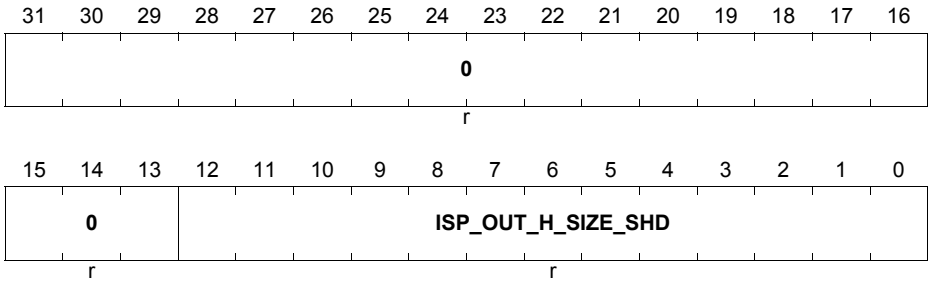
Current output horizontal picture size (shadow register)

CIFISP_OUT_H_SIZE_SHD

ISP Output Horizontal Picture Size Shadow Register

(01B4_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
ISP_OUT_H_SIZE_SHD	12:0	r	Current horizontal picture size In pixels Reset: 0000 _H
0	31:13	r	Reserved Read as 0.

ISP Output Vertical Picture Size Shadow Register

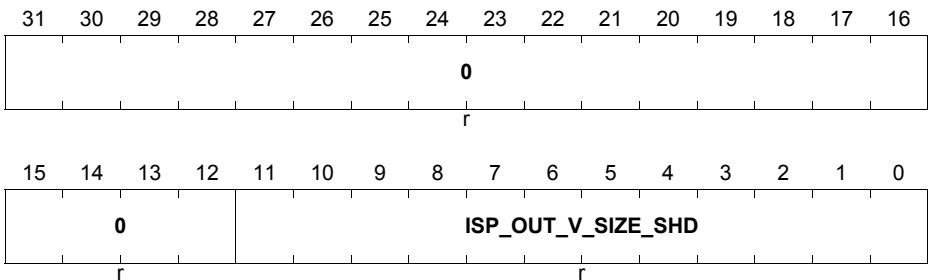
Current output vertical picture size (shadow register).

CIFISP_OUT_V_SIZE_SHD

ISP Output Vertical Picture Size Shadow Register

(01B8_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
ISP_OUT_V_SIZE_S HD	11:0	r	Current vertical pic size In lines Reset: 000 _H
0	31:12	r	Reserved Read as 0.

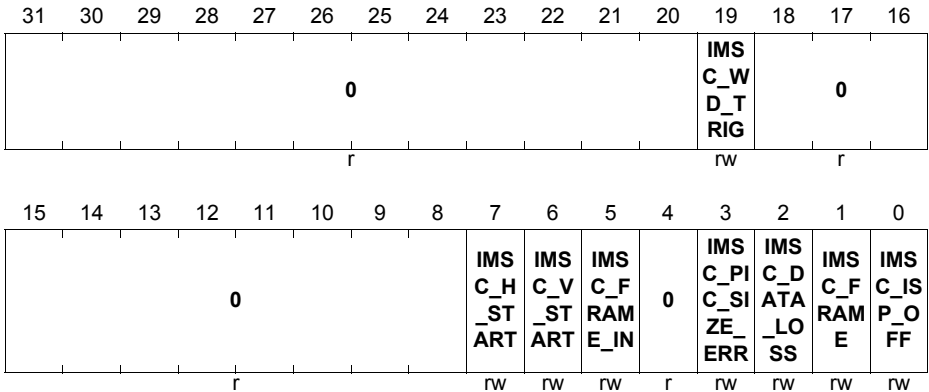
ISP Interrupt Control Registers

ISP Interrupt Mask Register

CIFISP_IMSC

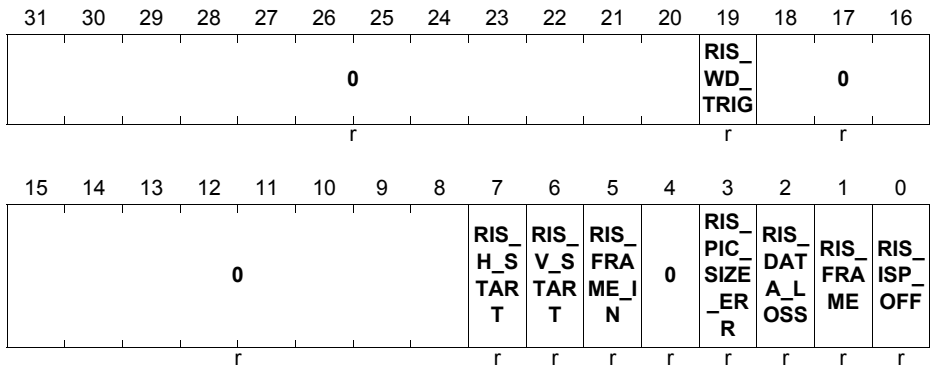
ISP Interrupt Mask Register

 (01BC_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
IMSC_WD_TRIG	19	rw	A watchdog timeout was triggered at the ISP input 0 _B mask out 1 _B enable interrupt Reset: 0 _B
IMSC_H_START	7	rw	Start Edge of h_sync 0 _B mask out 1 _B enable interrupt Reset: 0 _B
IMSC_V_START	6	rw	Start Edge of v_sync 0 _B mask out 1 _B enable interrupt Reset: 0 _B
IMSC_FRAME_IN	5	rw	Sampled Input Frame is Complete 0 _B mask out 1 _B enable interrupt Reset: 0 _B

Field	Bits	Type	Description
IMSC_PIC_SIZE_ER R	3	rw	Pic Size Violation Occurred Programming seems wrong 0 _B mask out 1 _B enable interrupt Reset: 0 _B
IMSC_DATA_LOSS	2	rw	Loss of Data Within a line, processing failure 0 _B mask out 1 _B enable interrupt Reset: 0 _B
IMSC_FRAME	1	rw	Frame was Completely Put Out 0 _B mask out 1 _B enable interrupt Reset: 0 _B
IMSC_ISP_OFF	0	rw	Isp was Turned Off (vsynced) Due to f_cnt reached or manual 0 _B mask out 1 _B enable interrupt Reset: 0 _B
0	31:20, 18:8, 4	r	Reserved Read as 0, should be written with 0.

ISP Raw Interrupt Status Register
CIFISP_RIS
ISP Raw Interrupt Status Register (01C0_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RIS_WD_TRIG	19	r	A watchdog timeout was triggered at the ISP input Reset: 0 _B
RIS_H_START	7	r	Start Edge of h_sync Reset: 0 _B
RIS_V_START	6	r	Start Edge of v_sync Reset: 0 _B
RIS_FRAME_IN	5	r	Sampled Input Frame is Complete Reset: 0 _B
RIS_PIC_SIZE_ERR	3	r	Pic Size Violation Occurred Programming seems wrong Reset: 0 _B
RIS_DATA_LOSS	2	r	Loss of Data Within a line, processing failure Reset: 0 _B
RIS_FRAME	1	r	Frame was Completely Put Out Reset: 0 _B
RIS_ISP_OFF	0	r	Isp was Turned Off (vsynced) Due to f_cnt reached or manual Reset: 0 _B

Field	Bits	Type	Description
0	31:20, 18:8, 4	r	Reserved

ISP Masked Interrupt Status Register

CIFISP_MIS

ISP Masked Interrupt Status Register

 (01C4_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0												MIS_WD_TRIG	0			
r												r	r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0							MIS_H_START	MIS_V_START	MIS_FRAME_IN	0	MIS_PIC_SIZE_ERR	MIS_DATA_LOSS	MIS_FRAME	MIS_ISP_OFF		
r							r	r	r	r	r	r	r	r		

Field	Bits	Type	Description
MIS_WD_TRIG	19	r	A watchdog timeout was triggered at the ISP input Reset: 0 _B
MIS_H_START	7	r	Start Edge of h_sync Reset: 0 _B
MIS_V_START	6	r	Start Edge of v_sync Reset: 0 _B
MIS_FRAME_IN	5	r	Sampled Input Frame is Complete Reset: 0 _B
MIS_PIC_SIZE_ERR	3	r	Pic Size Violation Occurred Programming seems wrong Reset: 0 _B
MIS_DATA_LOSS	2	r	Loss of Data Within a line, processing failure Reset: 0 _B

Field	Bits	Type	Description
MIS_FRAME	1	r	Frame was Completely Put Out Reset: 0 _B
MIS_ISP_OFF	0	r	Isp was Turned Off (vsyncd) Due to f_cnt reached or manual Reset: 0 _B
0	31:20, 18:8, 4	r	Reserved Read as 0.

ISP Interrupt Clear Register

CIFISP_ICR

ISP Interrupt Clear Register (01C8_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0												ICR_WD_TRIG	0			
r												w	r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0							ICR_H_START	ICR_V_START	ICR_FRAME_IN	0	ICR_PICTURE_SIZE	ICR_DATA_LOSS	ICR_FRAME	ICR_ISP_OFF		
r							w	w	w	r	w	w	w	w		

Field	Bits	Type	Description
ICR_WD_TRIG	19	w	A watchdog timeout was triggered at the ISP input Clear Interrupt Reset: 0 _B
ICR_H_START	7	w	Start Edge of h_sync Clear Interrupt Reset: 0 _B
ICR_V_START	6	w	Start Edge of v_sync Clear Interrupt Reset: 0 _B

Field	Bits	Type	Description
ICR_FRAME_IN	5	w	Sampled Input Frame is Complete Clear Interrupt Reset: 0 _B
ICR_PIC_SIZE_ERR	3	w	Pic Size Violation Occurred Clear Interrupt Reset: 0 _B
ICR_DATA_LOSS	2	w	Loss of Data Within a line, processing failure Clear Interrupt Reset: 0 _B
ICR_FRAME	1	w	Frame was Completely Put Out Clear Interrupt Reset: 0 _B
ICR_ISP_OFF	0	w	Isp was Turned Off (vsyncd) Clear Interrupt Reset: 0 _B
0	31:20, 18:8, 4	r	Reserved Read as 0, should be written with 0.

ISP Interrupt Set Register

CIFISP_ISR

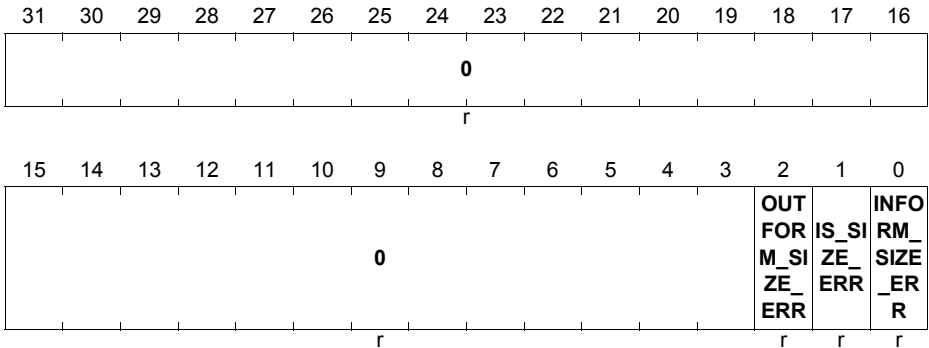
ISP Interrupt Set Register

 (01CC_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
0												ISR_WD_TRIG	0			
r												w	r			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0							ISR_H_S_TAR_T	ISR_V_S_TAR_T	ISR_FRA_ME_I_N	0	ISR_PIC_SIZE_ER_R	ISR_DAT_A_L_OSS	ISR_FRA_ME	ISR_ISP_OFF		
r							w	w	w	r	w	w	w	w		

Field	Bits	Type	Description
ISR_WD_TRIG	19	w	A watchdog timeout was triggered at the ISP input Set Interrupt Reset: 0 _B
ISR_H_START	7	w	Start Edge of h_sync Set Interrupt Reset: 0 _B
ISR_V_START	6	w	Start Edge of v_sync Set Interrupt Reset: 0 _B
ISR_FRAME_IN	5	w	Sampled Input Frame is Complete Set Interrupt Reset: 0 _B
ISR_PIC_SIZE_ERR	3	w	Pic Size Violation Occurred Set Interrupt Reset: 0 _B
ISR_DATA_LOSS	2	w	Loss of Data Within a line, processing failure Set Interrupt Reset: 0 _B
ISR_FRAME	1	w	Frame was Completely Put Out Set Interrupt Reset: 0 _B
ISR_ISP_OFF	0	w	Isp was Turned Off (vsynced) Set Interrupt Reset: 0 _B
0	31:20, 18:8, 4	r	Reserved Read as 0, should be written with 0.

Miscellaneous ISP Registers
ISP Error Register
CIFISP_ERR
ISP Error Register
(023C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
OUTFORM_SIZE_ERR	2	r	Size error is generated in outmux submodule Reset: 0 _B
IS_SIZE_ERR	1	r	Size error is generated in image stabilization submodule Reset: 0 _B
INFORM_SIZE_ERR	0	r	Size error is generated in inform submodule Reset: 0 _B
0	31:3	r	Reserved Read as 0.

CIFISP_ERR and CIFISP_ERR_CLR

For debug purposes the **CIFISP_ERR** and **CIFISP_ERR_CLR** are implemented. In the case a PIC_SIZE_ERR interrupt is signalled the SW is able to see in which submodule this error was generated. Writing to the **CIFISP_ERR_CLR** register clears this bit.

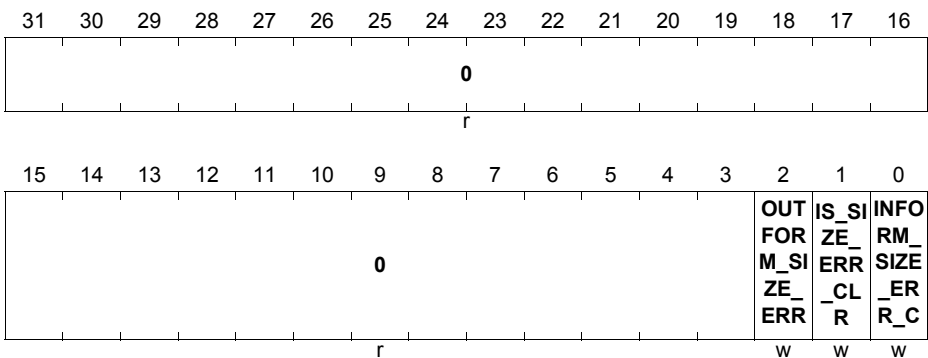
ISP Error Clear Register

CIFISP_ERR_CLR

ISP Error Clear Register

(0240_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
OUTFORM_SIZE_ERR_CLR	2	w	Size error is cleared Reset: 0 _B
IS_SIZE_ERR_CLR	1	w	Size error is cleared Reset: 0 _B
INFORM_SIZE_ERR_CLR	0	w	Size error is cleared Reset: 0 _B
0	31:3	r	Reserved Read as 0, should be written with 0.

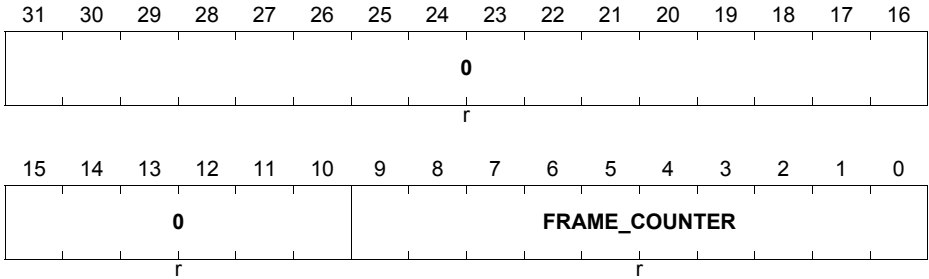
ISP Frame Counter Register

CIFISP_FRAME_COUNT

ISP Frame Counter Register

(0244_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
FRAME_COUNTER	9:0	r	<p>Current Frame Count of Processing</p> <p>In the ISP_FRAME_COUNT register the number of processed frames are displayed. For example: If a 8 is programmed into the ISP_ACQ_NR_FRAMES register, a read access to the ISP_FRAME_COUNT register during processing of the first picture shows a 7.</p> <p>After the entire frames are processed the ISP_OFF interrupt is generated and the ISP_FRAME_COUNT has the count zero. In case a '0' is programmed into the ISP_ACQ_NR_FRAMES register (continues mode) the ISP_FRAME_COUNT register keeps the value '0'. Reset: 000_H</p>
0	31:10	r	<p>Reserved</p> <p>Read as 0.</p>

36.1.3.4 Linear Downscaler Programming Registers

The address of each CIF Linear Downscaler register is evaluated as CIF_LIN_DSCALE_BASE + Offset.

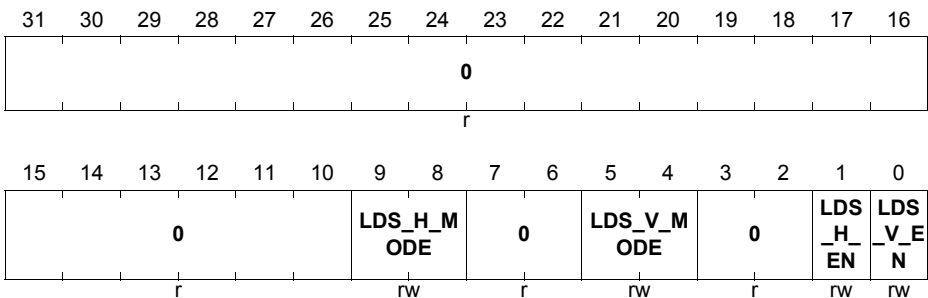
Linear Downscaler Configuration Registers

Linear Downscaler Control Register

CIFLDS_CTRL

Linear Downscaler Control Register (0000_H)

Reset Value: 0000 0000_H



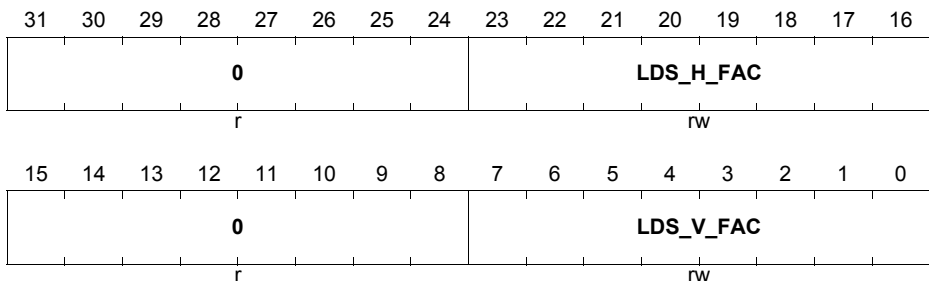
Field	Bits	Type	Description
LDS_H_MODE	9:8	rw	Horizontal scaling mode 00 _B Single skip 01 _B Double skip 10 _B Single pass 11 _B Double pass Reset: 00 _B
LDS_V_MODE	5:4	rw	Vertical scaling mode 00 _B Single skip 01 _B Double skip 10 _B Single pass 11 _B Double pass Reset: 00 _B
LDS_H_EN	1	rw	Horizontal scaling enable 0 _B Horizontal downscaling is disabled 1 _B Horizontal downscaling is enabled Reset: 0 _B

Field	Bits	Type	Description
LDS_V_EN	0	rw	Vertical scaling enable 0 _B Vertical downscaling is disabled 1 _B Vertical downscaling is enabled Reset: 0 _B
0	31:10, 7:6, 3:2	r	Reserved Read as 0, should be written with 0.

Linear Downscaler Factor Register

CIFLDS_FAC

Linear Downscaler Factor Register (0004_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
LDS_H_FAC	23:16	rw	Horizontal scaling factor Depending on the configured mode every lds_h_fac+1th pixel (or double pixel) will be skipped or passed on to the next module. Reset: 00 _H
LDS_V_FAC	7:0	rw	Vertical scaling factor Depending on the configured mode every lds_v_fac+1th line (or double line) will be skipped or passed on to the next module. Reset: 00 _H
0	31:24, 15:8	r	Reserved Read as 0, should be written with 0.

36.1.3.5 Memory Interface Programming Registers

The address of each CIF memory interface register is evaluated as CIF_MI_BASE + Offset.

Memory Interface Control Registers

Memory Interface Global Control Register

CIFMI_CTRL

Memory Interface Global Control Register

(0000_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								MP_WRI T	INIT_ OFF SET _EN	INIT_ BAS E_E N	0	BUR ST_L EN_ CHR OM	0	BUR ST_L EN_ LUM	
r								r/w	r/w	r/w	r	r/w	r	r/w	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							BYT E_S WAP	0		RAW _EN ABL E	JPE G_E NAB LE	0	MP_ ENA BLE		
r							r/w	r		r/w	r/w	r	r/w		

Field	Bits	Type	Description
MP_WRITE_FORMAT	23:22	rw	<p>Main Picture YCbCr Write Format Defines how YCbCr main picture data is written to memory. Ignored if JPEG data is chosen. The description listed first in the table below indicates the write format as used in YCbCr mode, the second one is valid for RAW data mode only.</p> <p><i>Note: Programmed value becomes effective immediately. So write to the register only if no picture data is sent to the path.</i></p> <p>00_B planar (YCbCr mode) / RAW & data mode (8 bit) 01_B semi-planar for YCbCr 4:2:x / RAW 8 bit 10_B interleaved_combined for YCbCr 4:2:2 only / RAW & data mode (greater 8 up to 16 bit) 11_B reserved / RAW greater 8 up to 16 bit Reset: 00_B</p>
INIT_OFFSET_EN	21	rw	<p>Init Offset Counter Enable Enables updating of the offset counters shadow registers for picture to the programmed register init values. MI_MP_Y/CB/CR_OFFS_CNT_INIT -> MI_MP_Y/CB/CR_OFFS_CNT_SHD The update will be executed either when a forced software update occurs (in register MI_INIT bit cfg_upd = 1) or when an automatic config update signal arrives at the MI input port. So only the corresponding shadow registers are affected. After a picture skip has been performed init_offset_en selects between skip restart and skip init mode (see bit skip in register MI_INIT). Reset: 0_B</p>

Field	Bits	Type	Description
INIT_BASE_EN	20	rw	<p>Init Base Address Enable</p> <p>Enables updating of the base address and buffer size shadow registers for picture to the programmed register init values.</p> <p>MI_MP_Y/CB/CR_BASE_AD_INIT -> MI_MP_Y/CB/CR_BASE_AD_SHD</p> <p>MI_MP_Y/CB/CR_SIZE_INIT -> MI_MP_Y/CB/CR_SIZE_SHD</p> <p>The update will be executed either when a forced software update occurs (in register MI_INIT bit <code>cfg_upd = 1</code>) or when an automatic config update signal arrives at the MI input port. So only the corresponding shadow registers are affected.</p> <p>Reset: 0_B</p>
BURST_LEN_CHROM	18	rw	<p>Chrominance Burst Length</p> <p>Burst length for Cb or Cr data affecting write port.</p> <p>Ignored if 8-beat bursts not supported.</p> <p><i>Note: Programmed value becomes effective immediately. So write to the register only if no picture data is sent .</i></p> <p>0_B 4-beat bursts 1_B 8-beat bursts</p> <p>Reset: 0_B</p>
BURST_LEN_LUM	16	rw	<p>Luminance Burst Length</p> <p>Burst length for Y, JPEG, or RAW data affecting write port.</p> <p>Ignored if 8-beat bursts not supported.</p> <p>This burst configuration is also applied to the Extra Paths.</p> <p><i>Note: Programmed value becomes effective immediately. So write to the register only if no picture data is sent .</i></p> <p>0_B 4-beat bursts 1_B 8-beat bursts</p> <p>Reset: 0_B</p>

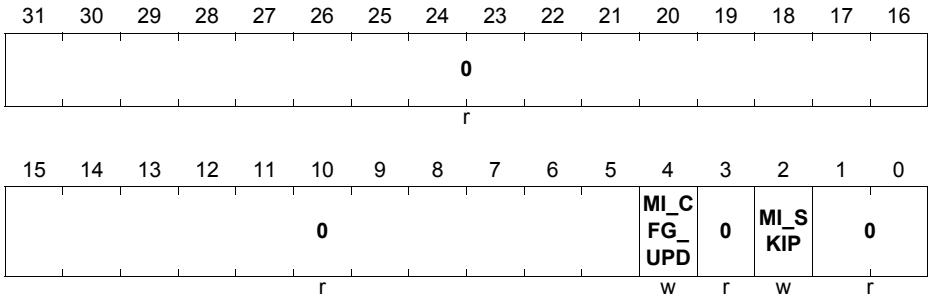
Field	Bits	Type	Description
BYTE_SWAP	7	rw	<p>Byte Swap Enable Enables change of byte order of the 32 bit output word at write port.</p> <p><i>Note: Programmed value becomes effective immediately. So write to the register only if no picture data is sent.</i></p> <p>0_B no byte mirroring 1_B byte order is mirrored but the bit order within one byte doesn't change</p> <p>Reset: 0_B</p>
RAW_ENABLE	3	rw	<p>Enables RAW mode Programmed value becomes effective (visible in control shadow register) after a soft reset or a forced software update. Affects MI_IN and MI_OUT module.</p> <p>Reset: 0_B</p>
JPEG_ENABLE	2	rw	<p>Enables JPEG mode Programmed value becomes effective (visible in control shadow register) after a soft reset or a forced software. Affects MI_IN and MI_OUT module.</p> <p>Reset: 0_B</p>
MP_ENABLE	0	rw	<p>Enables main picture data path, YCbCr mode Programmed value becomes effective (visible in control shadow register) after a soft reset, a forced software update or an automatic config update. Affects MI_IN and MI_OUT module.</p> <p>Reset: 0_B</p>
0	31:24, 19, 17, 15:8, 6:4, 1	r	<p>Reserved Read as 0, should be written with 0.</p>

Note: In order to select the appropriate mode, set one bit from RAW_ENABLE, JPEG_ENABLE and MP_ENABLE to 1, and set the the others to 0.

Memory Interface Control Register For Address Init And Skip Function Register

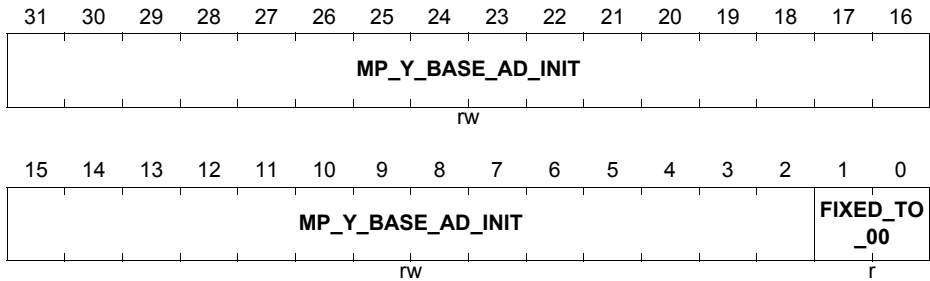
CIFMI_INIT

**Memory Interface Control Register For Address Init And Skip Function Register
(0004_H) Reset Value: 0000 0000_H**

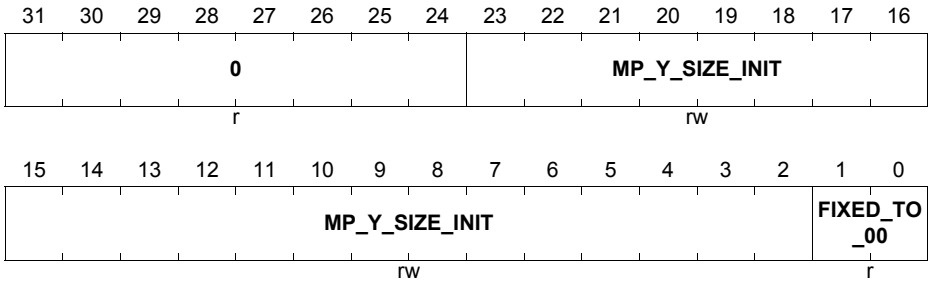


Field	Bits	Type	Description
MI_CFG_UPDATE	4	w	Forced Configuration Update Leads to an immediate update of the shadow registers. Depending on the two init enable bits in the MI_CTRL register (init_offset_en and init_base_en) the offset counter, base address and buffer size shadow registers are also updated. Reset: 0 _B

Field	Bits	Type	Description
MI_SKIP	2	w	<p>Skip Picture Skip of current or next starting main picture: Aborts writing of main picture image data of the current frame to RAM (after the current burst transmission has been completed). Further main picture data up to the end of the current frame are discarded. No further macroblock line interrupt (<code>_line</code>), no wrap around interrupt for main picture (<code>wrap_mp_y/cb/cr</code>) and no fill level interrupt (<code>fill_mp_y</code>) are generated. Skip does not affect the generation of the main path frame end interrupt (<code>mp_frame_end</code>). The byte counter (register <code>MI_BYTE_CNT</code>) is not affected. It produces the correct number of JPEG or RAW data bytes at the end of the current (skipped) frame. After a skip has been performed the offset counter for the main picture at the start of the following frame are set depending on the bit <code>init_offset_en</code> in register <code>MI_CTRL</code>:</p> <p>A) Skip restart mode (<code>init_offset_en = 0</code>) The offset counters of the main picture are restarted at the old start value of the previous skipped frame.</p> <p>B) Skip init mode (<code>init_offset_en = 1</code>) The offset counters of the main picture are initialized with the register contents of the offset counter init registers without any additional forced software update or automatic config update.</p> <p>Reset: 0_B</p>
0	31:5, 3, 1:0	r	<p>Reserved Read as 0, should be written with 0.</p>

Memory Interface Base Address For Main Picture Y Component, JPEG or RAW Data Register
CIFMI_MP_Y_BASE_AD_INIT
Memory Interface Base Address For Main Picture Y Component, JPEG or RAW Data Register (0008_H) Reset Value: 0000 0000_H


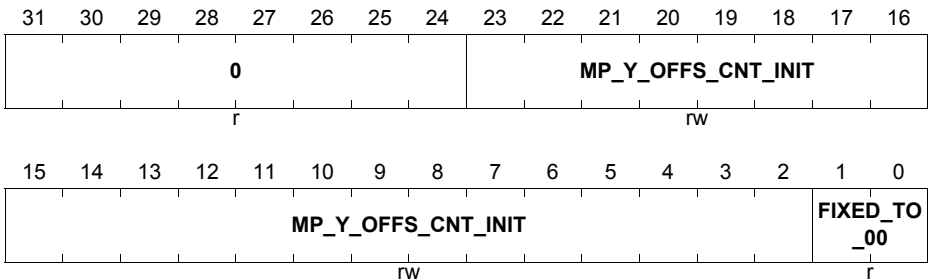
Field	Bits	Type	Description
MP_Y_BASE_AD_INIT	31:2	rw	Main Picture Y Base Address Init Base address of main picture Y component ring buffer, JPEG ring buffer or RAW data ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update. <i>Note: Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B

Memory Interface Size Of Main Picture Y Component, JPEG or RAW Data Register
CIFMI_MP_Y_SIZE_INIT
**Memory Interface Size of main picture Y component, JPEG or RAW data Register
(000C_H) Reset Value: 0000 0000_H**


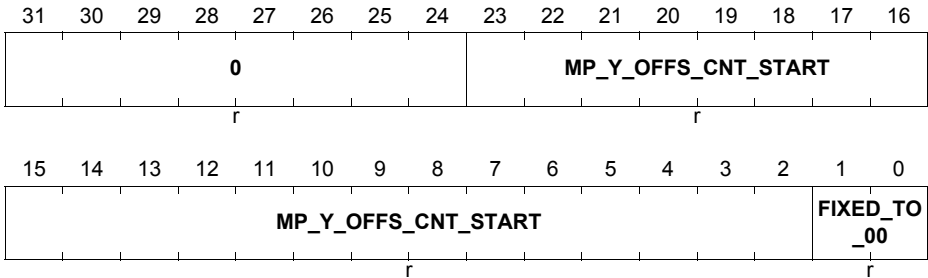
Field	Bits	Type	Description
MP_Y_SIZE_INIT	23:2	rw	Main Picture Y Size Init Size of main picture Y component ring buffer, JPEG ring buffer or RAW data ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update. <i>Note: Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0, should be written with 0.

Memory Interface Offset Counter Init Value For Main Picture Y, JPEG or RAW Data Register
CIFMI_MP_Y_OFFSETS_CNT_INIT
Memory Interface Offset Counter Init Value For Main Picture Y, JPEG or RAW Data Register

 (0010_H)

 Reset Value: 0000 0000_H


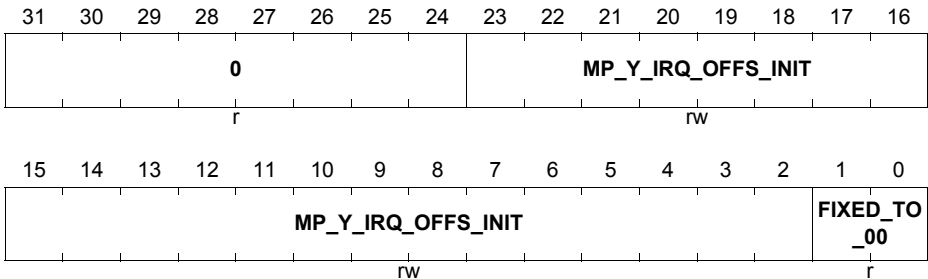
Field	Bits	Type	Description
MP_Y_OFFSETS_CNT_INIT	23:2	rw	<p>Main Picture Y Offset Counter Init Offset counter init value of main picture Y component ring buffer, JPEG ring buffer or RAW data ring buffer.</p> <p>Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note: Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect. Check exceptional handling in skip modes.</i></p> <p>Reset: 0_D</p>
FIXED_TO_00	1:0	r	<p>Bits [1:0] are set to “00” (word aligned value). Reset: 00_B</p>
0	31:24	r	<p>Reserved Read as 0, should be written with 0.</p>

Memory Interface Offset Counter Start Value For Main Picture Y, JPEG or RAW Data Register
CIFMI_MP_Y_OFFS_CNT_START
Memory Interface Offset Counter Start Value For Main Picture Y, JPEG or RAW Data Register
 (0014_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
MP_Y_OFFS_CNT_S TART	23:2	r	Main Picture Y Offset Counter Start Offset counter value which points to the start address of the previously processed picture (main picture Y component, JPEG or RAW data). Updated at frame end. Note: A soft reset resets the contents to the reset value. Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

Memory Interface Fill Level Interrupt Offset Value For Main Picture Y, JPEG or RAW Data Register

Register 236

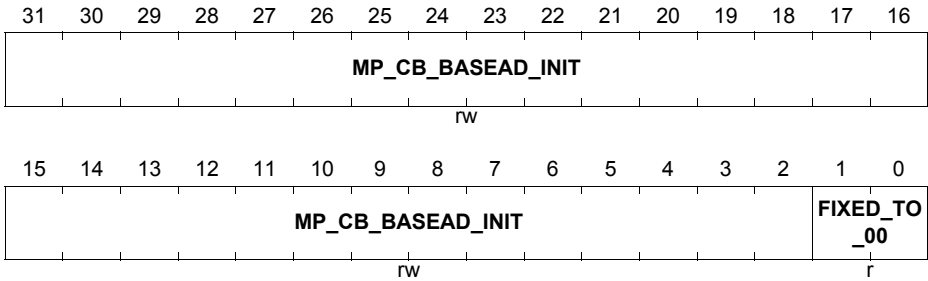
CIFMI_MP_Y_IRQ_OFFS_INIT
Memory Interface Fill Level Interrupt Offset Value For Main Picture Y, JPEG or RAW Data Register (0018_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
MP_Y_IRQ_OFFS_INIT	23:2	rw	Main Picture Y IRQ Offset Init Reaching this programmed value by the current offset counter for addressing main picture Y component, JPEG or RAW data leads to generation of fill level interrupt fill_mp_y. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update. Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0, should be written with 0.

Memory Interface Base Address For Main Picture Cb Component Ring Buffer Register

CIFMI_MP_CB_BASE_AD_INIT

Memory Interface Base Address For Main Picture Cb Component Ring Buffer Register (001C_H) **Reset Value: 0000 0000_H**



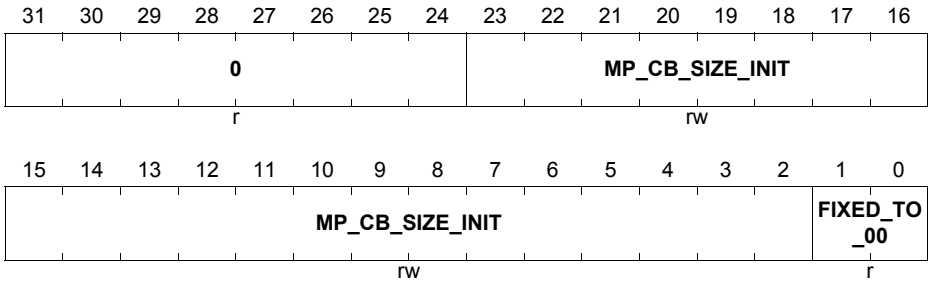
Field	Bits	Type	Description
MP_CB_BASEAD_INIT	31:2	rw	<p>Main Picture Cb Base Address Init Base address of main picture Cb component ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note: Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect.</i></p> <p>Reset: 0_D</p>
FIXED_TO_00	1:0	r	<p>Bits [1:0] are set to "00" (word aligned value). Reset: 00_B</p>

Memory Interface Size Of Main Picture Cb Component Ring Buffer Register

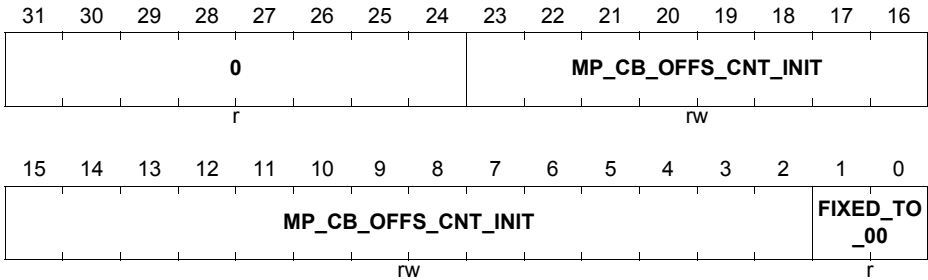
Register 238

CIFMI_MP_CB_SIZE_INIT
Memory Interface Size Of Main Picture Cb Component Ring Buffer Register

 (0020_H)

 Reset Value: 0000 0000_H


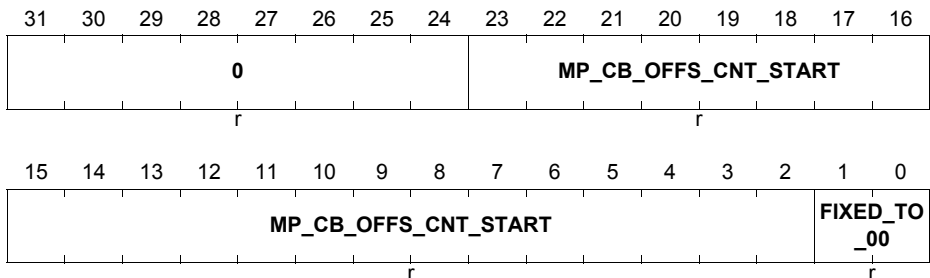
Field	Bits	Type	Description
MP_CB_SIZE_INIT	23:2	rw	Main Picture Cb Size Init Size of main picture Cb component ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update. <i>Note: Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0, should be written with 0.

Memory Interface Offset Counter Init Value For Main Picture Cb Component Ring Buffer Register
CIFMI_MP_CB_OFFS_CNT_INIT
Memory Interface Offset Counter Init Value For Main Picture Cb Component Ring Buffer Register
 (0024_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
MP_CB_OFFS_CNT_INIT	23:2	rw	Main Picture Cb Offset Counter Init Offset counter init value of main picture Cb component ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update. <i>Note: Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect. Check exceptional handling in skip modes.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0, should be written with 0.

Memory Interface Offset Counter Start Value For Main Picture Cb Component Ring Buffer Register

Register 240

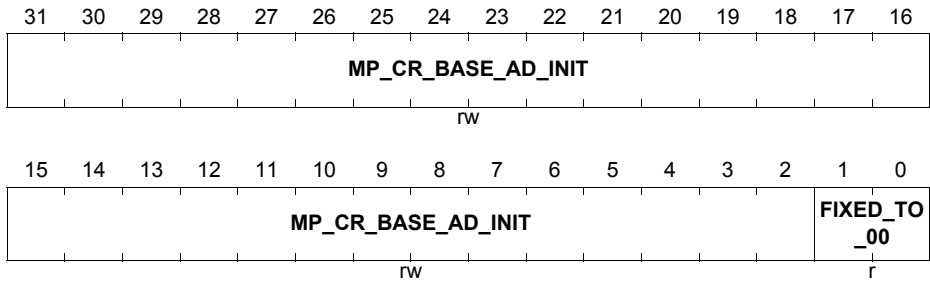
CIFMI_MP_CB_OFFS_CNT_START
Memory Interface Offset Counter Start Value For Main Picture Cb Component Ring Buffer Register
 (0028_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
MP_CB_OFFS_CNT_START	23:2	r	Main Picture Cb Offset Count Start Offset counter value which points to the start address of the previously processed picture (main picture Cb component). Updated at frame end. Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

Memory Interface Base Address For Main Picture Cr Component Ring Buffer Register

CIFMI_MP_CR_BASE_AD_INIT

Memory Interface Base Address For Main Picture Cr Component Ring Buffer Register (002C_H) Reset Value: 0000 0000_H



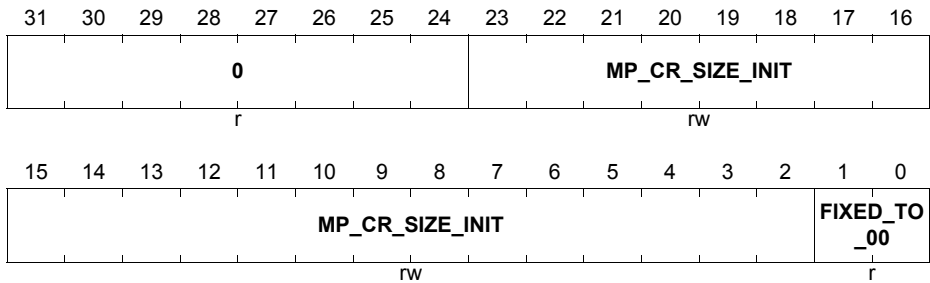
Field	Bits	Type	Description
MP_CR_BASE_AD_INIT	31:2	rw	<p>Main Picture Cr Base Address Init</p> <p>Base address of main picture Cr component ring buffer.</p> <p>Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update.</p> <p><i>Note: Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect.</i></p> <p>Reset: 0_D</p>
FIXED_TO_00	1:0	r	<p>Bits [1:0] are set to “00” (word aligned value).</p> <p>Reset: 00_B</p>

Memory Interface Size Of Main Picture Cr Component Ring Buffer Register

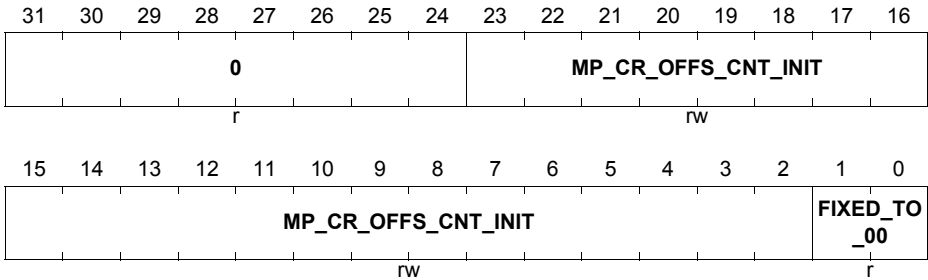
Register 242

CIFMI_MP_CR_SIZE_INIT
Memory Interface Size Of Main Picture Cr Component Ring Buffer Register

 (0030_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
MP_CR_SIZE_INIT	23:2	rw	Main Picture Cr Size Init Size of main picture Cr component ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update. <i>Note: Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to "00" (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0, should be written with 0.

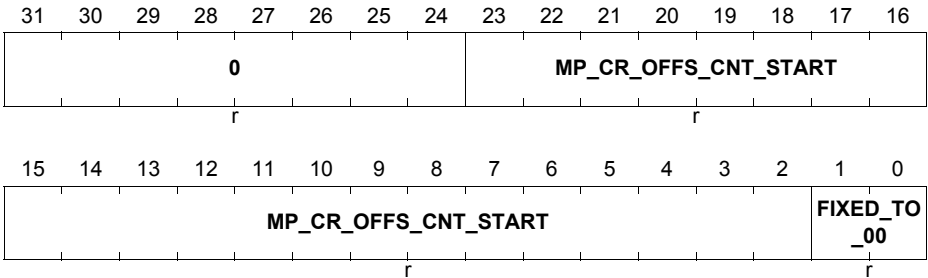
Memory Interface Offset Counter Init value For Main Picture Cr Component Ring Buffer Register
CIFMI_MP_CR_OFFS_CNT_INIT
Memory Interface Offset Counter Init value For Main Picture Cr Component Ring Buffer Register (0034_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
MP_CR_OFFS_CNT_INIT	23:2	rw	Main Picture Cr Offset Counter Init Offset counter init value of main picture Cr component ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update. <i>Note: Set control bit init_base_en before updating so that a forced or automatic update can take effect. Check exceptional handling in skip modes.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0, should be written with 0.

Memory Interface Offset Counter Start Value For Main Picture Cr Component Ring Buffer Register

CIFMI_MP_CR_OFFS_CNT_START

Memory Interface Offset Counter Start Value For Main Picture Cr Component Ring Buffer Register (0038_H) Reset Value: 0000 0000_H

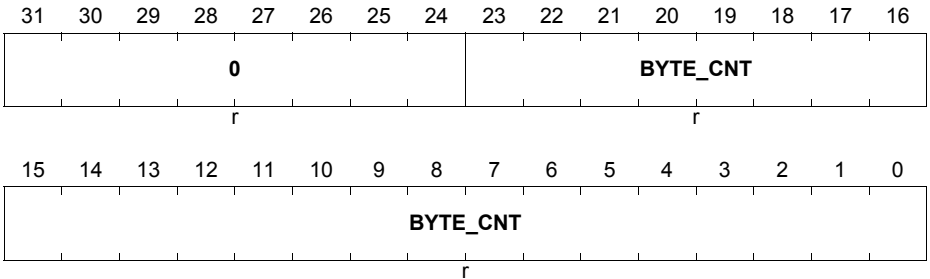


Field	Bits	Type	Description
MP_CR_OFFS_CNT_START	23:2	r	Main Picture Cr Offset Counter Start Offset counter value which points to the start address of the previously processed picture (main picture Cr component). Updated at frame end. <i>Note: Soft reset will reset the contents to reset value.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

Memory Interface Counter Value of JPEG or RAW Data Bytes Register

CIFMI_BYTE_CNT

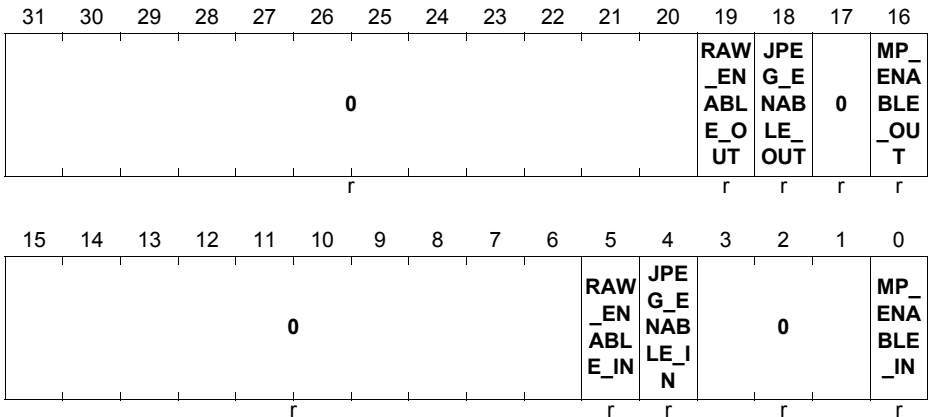
Memory Interface Counter Value of JPEG or RAW Data Bytes Register
(0070_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
BYTE_CNT	23:0	r	Byte Count Counter value specifies the number of JPEG or RAW data bytes of the last transmitted frame. Updated at frame end. A soft reset will set the byte counter to zero. Reset: 00 0000 _H
0	31:24	r	Reserved Read as 0.

Memory Interface Shadow Registers
Memory Interface Global Control Internal Shadow Register
CIFMI_CTRL_SHD
Memory Interface Global Control Internal Shadow Register

 (0074_H)

 Reset Value: 0000 0000_H


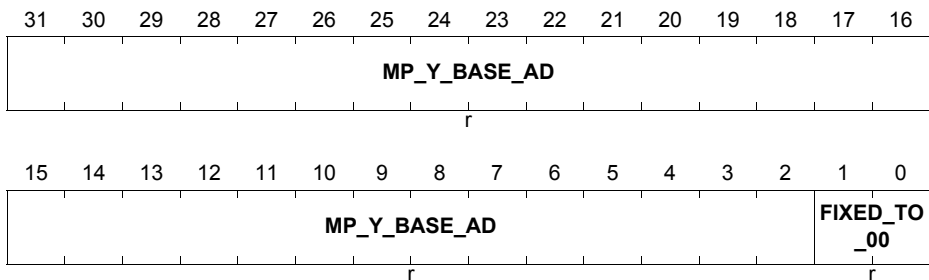
Field	Bits	Type	Description
RAW_ENABLE_OUT	19	r	RAW Out Enable Raw data used in module MI_OUT Reset: 0 _B
JPEG_ENABLE_OUT	18	r	JPEG Out Enable JPEG data used in module MI_OUT Reset: 0 _B
MP_ENABLE_OUT	16	r	Main Picture Out Enable Main picture used in module MI_OUT Reset: 0 _B
RAW_ENABLE_IN	5	r	RAW In Enable RAW data used in module MI_IN Reset: 0 _B
JPEG_ENABLE_IN	4	r	JPEG In Enable JPEG data used in module MI_IN Reset: 0 _B

Field	Bits	Type	Description
MP_ENABLE_IN	0	r	Main Picture In Enable Main picture data used in module MI_IN Reset: 0 _B
0	31:20, 17, 15:6, 3:1	r	Reserved Read as 0.

Memory Interface Base Address Shadow Register For Main Picture Y Component, JPEG Register

CIFMI_MP_Y_BASE_AD_SHD

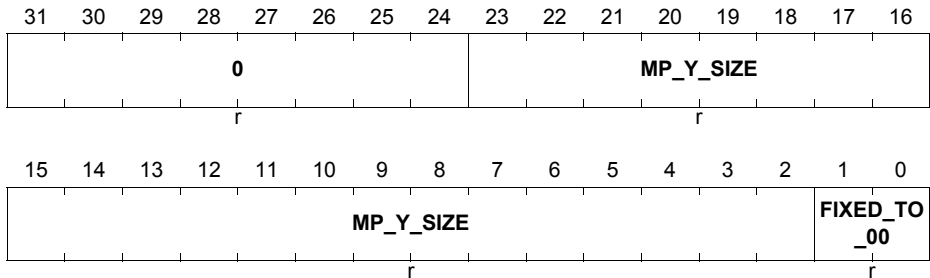
Memory Interface Base Address Shadow Register For Main Picture Y Component, JPEG Register (0078_H) Reset Value: 0000 0000_H



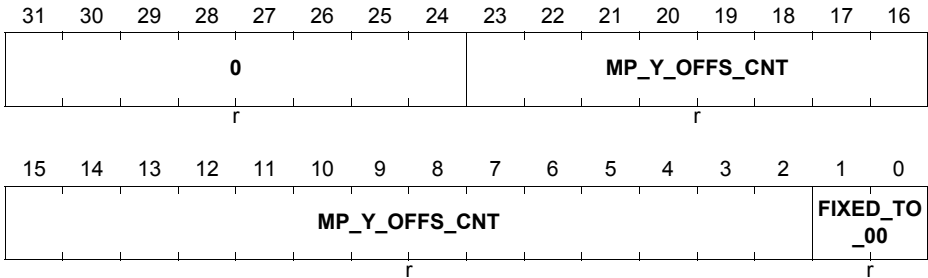
Field	Bits	Type	Description
MP_Y_BASE_AD	31:2	r	Main Picture Y Base Address Base address of main picture Y component ring buffer, JPEG ring buffer or RAW data ring buffer Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to "00" (word aligned value). Reset: 00 _B

Memory Interface Size Shadow Register of Main Picture Y Component, JPEG or RAW Data Register

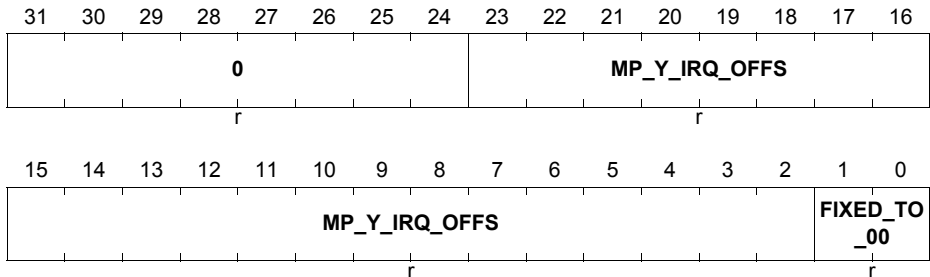
Register 264

CIFMI_MP_Y_SIZE_SHD
Memory Interface Size Shadow Register of Main Picture Y Component, JPEG or RAW Data Register (007C_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
MP_Y_SIZE	23:2	r	Main Picture Y Size Size of main picture Y component ring buffer, JPEG ring buffer or RAW data ring buffer Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

Memory Interface Current Offset Counter of Main Picture Y Component, JPEG or RAW Register
CIFMI_MP_Y_OFFS_CNT_SHD
Memory Interface Current Offset Counter of Main Picture Y Component JPEG or RAW Register (0080_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
MP_Y_OFFS_CNT	23:2	r	Main Picture Y Offset Counter Current offset counter of main picture Y component, JPEG or RAW data ringbuffer for address generation <i>Note: Soft reset will reset the contents to reset value.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

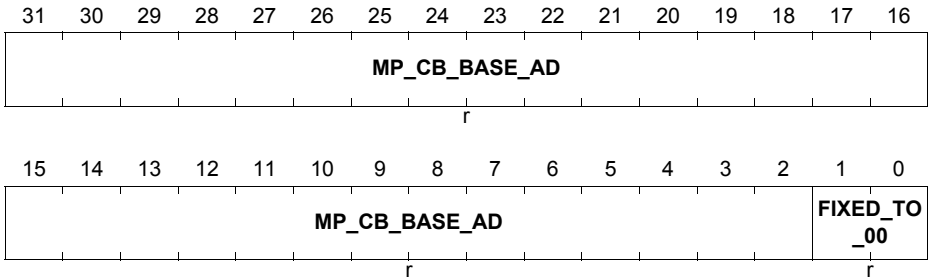
Memory Interface Shadow Register of Fill Level Interrupt Offset Value For Main Picture Y Register
CIFMI_MP_Y_IRQ_OFFS_SHD
Memory Interface Shadow Register of Fill Level Interrupt Offset Value For Main Picture Y Register (0084_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
MP_Y_IRQ_OFFSETS	23:2	r	Main Picture Y IRQ Offset Reaching this offset value by the current offset counter for addressing main picture Y component, JPEG or RAW data leads to generation of fill level interrupt fill_mp_y. Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

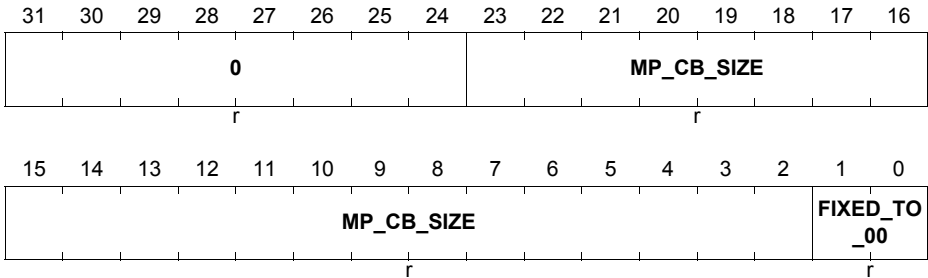
Memory Interface Base Address Shadow Register For Main Picture Cb Component Ring Register

CIFMI_MP_CB_BASE_AD_SHD

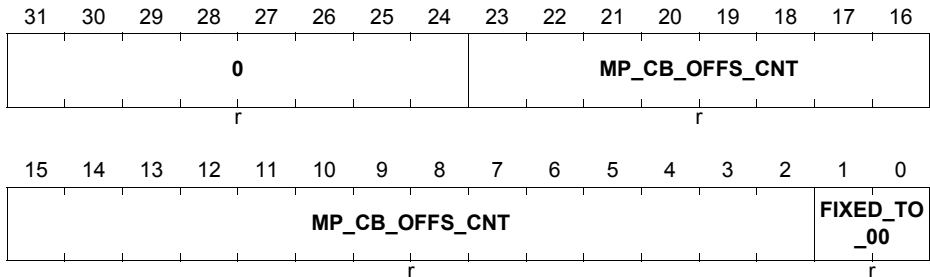
Memory Interface Base Address Shadow Register For Main Picture Cb Component Ring Register (0088_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
MP_CB_BASE_AD	31:2	r	Main Picture Cb Base Address Base address of main picture Cb component ring buffer Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to "00" (word aligned value). Reset: 00 _B

Memory Interface Size Shadow Register Of Main Picture Cb Component Ring Buffer Register
CIFMI_MP_CB_SIZE_SHD
Memory Interface Size Shadow Register Of Main Picture Cb Component Ring Buffer Register (008C_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
MP_CB_SIZE	23:2	r	Main Picture Cb Size Size of main picture Cb component ring buffer Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

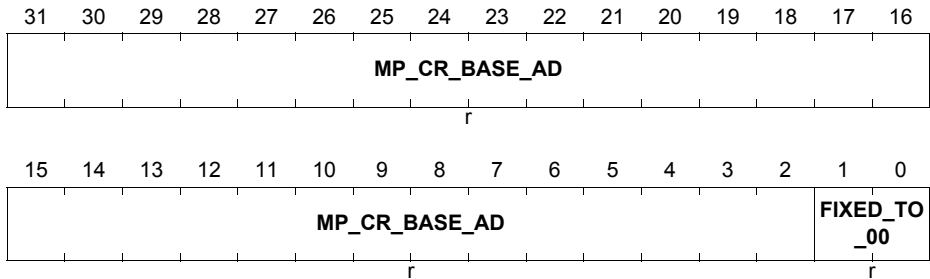
Memory Interface Current Offset Counter Of Main Picture Cb Component Ring Buffer Register
CIFMI_MP_CB_OFFS_CNT_SHD
Memory Interface Current Offset Counter Of Main Picture Cb Component Ring Buffer Register (0090_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
MP_CB_OFFS_CNT	23:2	r	Main Picture Cb Offset Counter Current offset counter of main picture Cb component ring buffer for address generation <i>Note: Soft reset will reset the contents to reset value.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

Memory Interface Base Address Shadow Register For Main Picture Cr Component Ring Register

CIFMI_MP_CR_BASE_AD_SHD

Memory Interface Base Address Shadow Register For Main Picture Cr Component Ring Register (0094_H) Reset Value: 0000 0000_H

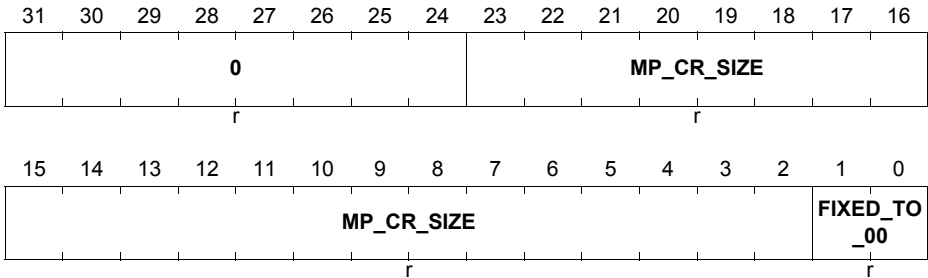


Field	Bits	Type	Description
MP_CR_BASE_AD	31:2	r	Main Picture Cr Base Address Base address of main picture Cr component ring buffer Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B

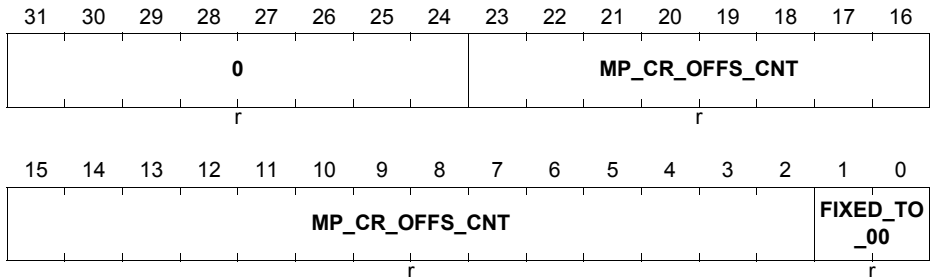
Memory Interface Size Shadow Register Of Main Picture Cr Component Ring Buffer Register

CIFMI_MP_CR_SIZE_SHD

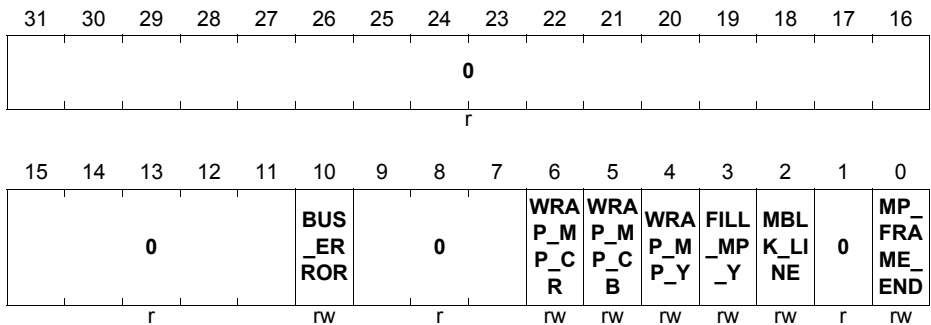
Memory Interface Size Shadow Register Of Main Picture Cr Component Ring Buffer Register (0098_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
MP_CR_SIZE	23:2	r	Main Picture Cr Size Size of main picture Cr component ring buffer Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

Memory Interface Current Offset Counter Of Main Picture Cr Component Ring Buffer Register
CIFMI_MP_CR_OFFS_CNT_SHD
Memory Interface Current Offset Counter Of Main Picture Cr Component Ring Buffer Register (009C_H) Reset Value: 0000 0000_H


Field	Bits	Type	Description
MP_CR_OFFS_CNT	23:2	r	Main Picture Cr Offset Counter Current offset counter of main picture Cr component ring buffer for address generation <i>Note: Soft reset will reset the contents to reset value.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

Memory Interface Interrupt Registers
MI Interrupt Mask ('1': interrupt active, '0': interrupt masked)
CIFMI_IMSC
MI Interrupt Mask '1' interrupt active '0' interrupt masked
(00F8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
BUS_ERROR	10	rw	Bus Error Mask bit for Bus write or read error interrupt (from handshake target interfaces) Reset: 0 _B
WRAP_MP_CR	6	rw	Wrap Main Picture Cr Mask bit for main picture Cr address wrap interrupt Reset: 0 _B
WRAP_MP_CB	5	rw	Wrap Main Picture Cb Mask bit for main picture Cb address wrap interrupt Reset: 0 _B
WRAP_MP_Y	4	rw	Wrap Main Picture Y Mask bit for main picture Y address wrap interrupt Reset: 0 _B

Field	Bits	Type	Description
FILL_MP_Y	3	rw	Fill Main Picture Y Mask bit for fill level interrupt of main picture Y, JPEG or RAW data Reset: 0 _B
MBLK_LINE	2	rw	Macro Block Line Interrupt Mask bit for macroblock line interrupt of main picture (16 lines of Y, 8 lines of Cb and 8 lines of Cr are written into RAM) Reset: 0 _B
MP_FRAME_END	0	rw	Main Picture Frame End Mask main picture end of frame interrupt Reset: 0 _B
0	31:11, 9:7, 1	r	Reserved Read as 0, should be written with 0.

MI Raw Interrupt Status Register

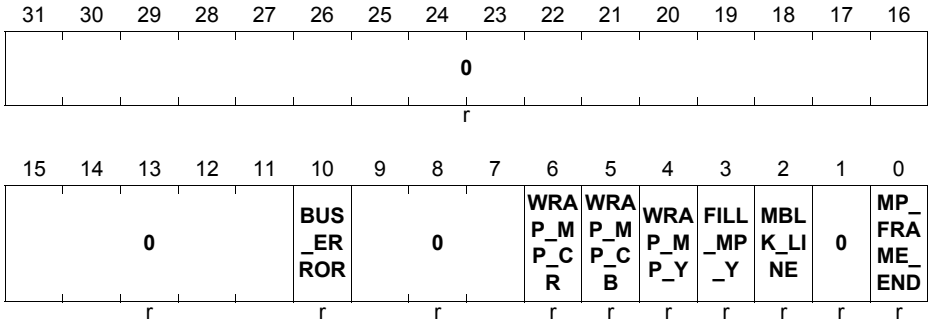
CIFMI_RIS

MI Raw Interrupt Status Register (00FC_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0					BUS_ERROR	0			WRAMP_C_R	WRAMP_C_B	WRAMP_Y	FILL_MP_Y	MBLK_LINE	0	MP_FRAME_END
r					r	r			r	r	r	r	r	r	r

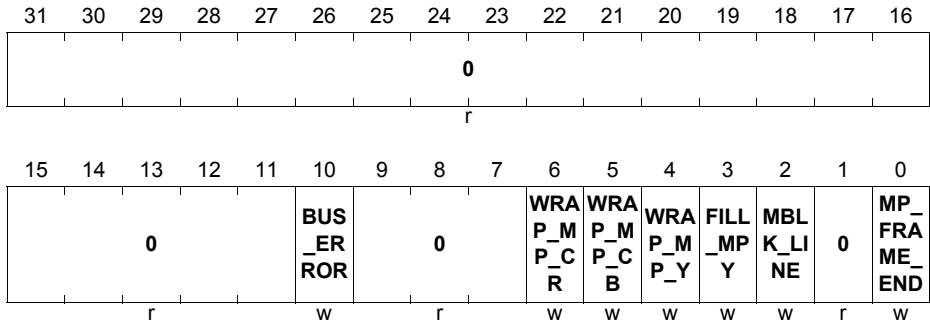
Field	Bits	Type	Description
BUS_ERROR	10	r	Bus Error Raw status of Bus write or read error interrupt (from handshake target interfaces) Reset: 0 _B

Field	Bits	Type	Description
WRAP_MP_CR	6	r	Wrap Main Picture Cr Raw status of main picture Cr address wrap interrupt Reset: 0 _B
WRAP_MP_CB	5	r	Wrap Main Picture Cb Raw status of main picture Cb address wrap interrupt Reset: 0 _B
WRAP_MP_Y	4	r	Wrap Main Picture Y Raw status of main picture Y address wrap interrupt Reset: 0 _B
FILL_MP_Y	3	r	Fill Main Picture Y Raw status of fill level interrupt of main picture Y, JPEG or RAW data Reset: 0 _B
MBLK_LINE	2	r	Macro Block Line Interrupt Raw status of macroblock line interrupt of main picture (16 lines of Y, 8 lines of Cb and 8 lines of Cr are written into RAM, valid only for planar and semi-planar mode) Reset: 0 _B
MP_FRAME_END	0	r	Main Picture Frame End Raw status of main picture end of frame interrupt Reset: 0 _B
0	31:11, 9:7, 1	r	Reserved Read as 0.

MI Masked Interrupt Status Register
CIFMI_MIS
MI Masked Interrupt Status Register (0100_H)
Reset Value: 0000 0000_H


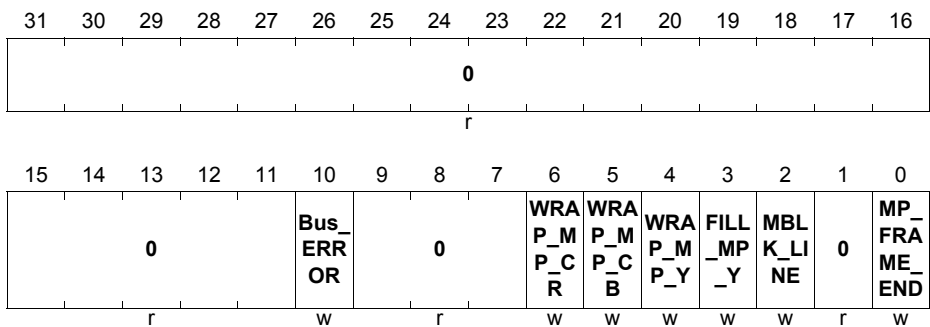
Field	Bits	Type	Description
BUS_ERROR	10	r	Bus Error Masked status of Bus write or read error interrupt (from handshake target interfaces) Reset: 0 _B
WRAP_MP_Cr	6	r	Wrap Main Picture Cr Masked status of main picture Cr address wrap interrupt Reset: 0 _B
WRAP_MP_Cb	5	r	Wrap Main Picture Cb Masked status of main picture Cb address wrap interrupt Reset: 0 _B
WRAP_MP_Y	4	r	Wrap Main Picture Y Masked status of main picture Y address wrap interrupt Reset: 0 _B
FILL_MP_Y	3	r	Fill Main Picture Y Masked status of fill level interrupt of main picture Y, JPEG or RAW data Reset: 0 _B

Field	Bits	Type	Description
MBLK_LINE	2	r	Macro Block Line Interrupt Masked status of macroblock line interrupt of main picture (16 lines of Y, 8 lines of Cb and 8 lines of Cr are written into RAM, valid only for planar and semi-planar mode) Reset: 0 _B
MP_FRAME_END	0	r	Main Picture Frame End Masked status of main picture end of frame interrupt Reset: 0 _B
0	31:11, 9:7, 1	r	Reserved Read as 0.

MI Interrupt Clear Register
CIFMI_ICR
MI Interrupt Clear Register
(0104_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
BUS_ERROR	10	w	Bus Error Clear Bus write or read error interrupt (from handshake target interfaces) Reset: 0 _B
WRAP_MP_CR	6	w	Wrap Main Picture Cr Clear main picture Cr address wrap interrupt Reset: 0 _B
WRAP_MP_CB	5	w	Wrap Main Picture Cb Clear main picture Cb address wrap interrupt Reset: 0 _B
WRAP_MP_Y	4	w	Wrap Main Picture Y Clear main picture Y address wrap interrupt Reset: 0 _B
FILL_MP_Y	3	w	Fill Main Picture Y Clear fill level interrupt of main picture Y, JPEG or RAW data Reset: 0 _B

Field	Bits	Type	Description
MBLK_LINE	2	w	Macro Block Line Interrupt Clear macroblock line interrupt of main picture (16 lines of Y, 8 lines of Cb and 8 lines of Cr are written into RAM, valid only for planar and semi-planar mode) Reset: 0 _B
MP_FRAME_END	0	w	Main Picture Frame End Clear main picture end of frame interrupt Reset: 0 _B
0	31:11, 9:7, 1	r	Reserved Read as 0, should be written with 0.

MI Interrupt Set Register
CIFMI_ISR
MI Interrupt Set Register
(0108_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
Bus_ERROR	10	w	Bus Error Set Bus write or read error interrupt (from handshake target interfaces) Reset: 0 _B
WRAP_MP_CR	6	w	Wrap Main Picture Cr Set main picture Cr address wrap interrupt Reset: 0 _B

Field	Bits	Type	Description
WRAP_MP_CB	5	w	Wrap Main Picture Cb Set main picture Cb address wrap interrupt Reset: 0 _B
WRAP_MP_Y	4	w	Wrap Main Picture Y Set main picture Y address wrap interrupt Reset: 0 _B
FILL_MP_Y	3	w	Fill Main Picture Y Set fill level interrupt of main picture Y, JPEG or RAW data Reset: 0 _B
MBLK_LINE	2	w	Macro Block Line Interrupt Set macroblock line interrupt of main picture (16 lines of Y, 8 lines of Cb and 8 lines of Cr are written into RAM, valid only for planar and semi-planar mode) Reset: 0 _B
MP_FRAME_EN D	0	w	Main Picture Frame End Set main picture end of frame interrupt Reset: 0 _B
0	31:11, 9:7, 1	r	Reserved Read as 0, should be written with 0.

MI Status Register
CIFMI_STATUS
MI Status Register
(010C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
r															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							BUS _WR ITE_ ERR OR	0					MP_ CR_ FIFO _FU LL	MP_ CB_ FIFO _FU LL	MP_ Y_FI FO_ FUL L
r							r	r					r	r	r

Field	Bits	Type	Description
BUS_WRITE_ERROR	8	r	Bus Write Error An Bus error occurred (Bus_error interrupt raised) while writing to the Bus (main/self path) since last clear. Reset: 0 _B
MP_CR_FIFO_FULL	2	r	Main Picture Cr FIFO Full FIFO full flag of Cr FIFO in main path asserted since last clear Reset: 0 _B
MP_CB_FIFO_FULL	1	r	Main Picture Cb FIFO Full FIFO full flag of Cb FIFO in main path asserted since last clear Reset: 0 _B
MP_Y_FIFO_FULL	0	r	Main Picture Y FIFO Full FIFO full flag of Y FIFO in main path asserted since last clear Reset: 0 _B
0	31:9, 7:3	r	Reserved Read as 0.

MI Status Clear Register

CIFMI_STATUS_CLR

MI Status Clear Register

 (0110_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		EP_5_FIF_O_F_ULL	EP_4_FIF_O_F_ULL	EP_3_FIF_O_F_ULL	EP_2_FIF_O_F_ULL	EP_1_FIF_O_F_ULL	0								
r		w	w	w	w	w	r								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							BUS_WRITE_ERROR	0					MP_CR_FIFO_FULL	MP_CB_FIFO_FULL	MP_Y_FIFO_FULL
r							w	r					w	w	w

Field	Bits	Type	Description
EP_5_FIFO_FULL	28	w	Extra Path 5 FIFO Full Clear status of FIFO full flag in extra path Reset: 0 _B
EP_4_FIFO_FULL	27	w	Extra Path 4 FIFO Full Clear status of FIFO full flag in extra path Reset: 0 _B
EP_3_FIFO_FULL	26	w	Extra Path 3 FIFO Full Clear status of FIFO full flag in extra path Reset: 0 _B
EP_2_FIFO_FULL	25	w	Extra Path 2 FIFO Full Clear status of FIFO full flag in extra path Reset: 0 _B
EP_1_FIFO_FULL	24	w	Extra Path 1 FIFO Full Clear status of FIFO full flag in extra path Reset: 0 _B
BUS_WRITE_ERROR	8	w	Bus Write Error Clear status of Bus write error flag Reset: 0 _B
MP_CR_FIFO_FULL	2	w	Main Picture Cr FIFO Full Clear status of Cr FIFO full flag in main path Reset: 0 _B
MP_CB_FIFO_FULL	1	w	Main Picture Cb FIFO Full Clear status of Cb FIFO full flag in main path Reset: 0 _B
MP_Y_FIFO_FULL	0	w	Main Picture Y FIFO Full Clear status of Y FIFO full flag in main path Reset: 0 _B
0	31:29, 23:9, 7:3	r	Reserved Read as 0, should be written with 0.

36.1.3.6 JPEG Encoder Programming Registers

The address of each CIF JPEG encoder register is evaluated as CIF_JPE_BASE + Offset.

JPEG Encoder Control Registers

JPE Command To Start Stream Header Generation Register

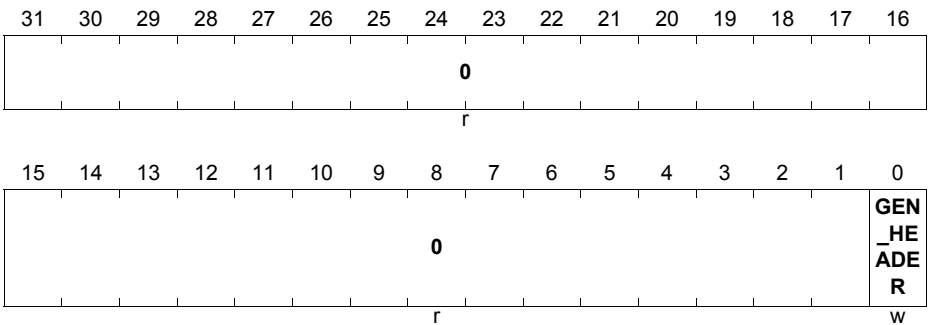
Register 305

CIFJPE_GEN_HEADER

JPE Command To Start Stream Header Generation Register

(0000_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
GEN_HEADER	0	w	Generate Header 1 _B Start command to generate stream header ; auto reset to zero after one clock cycle Reset: 0 _B
0	31:1	r	Reserved Read as 0, should be written with 0.

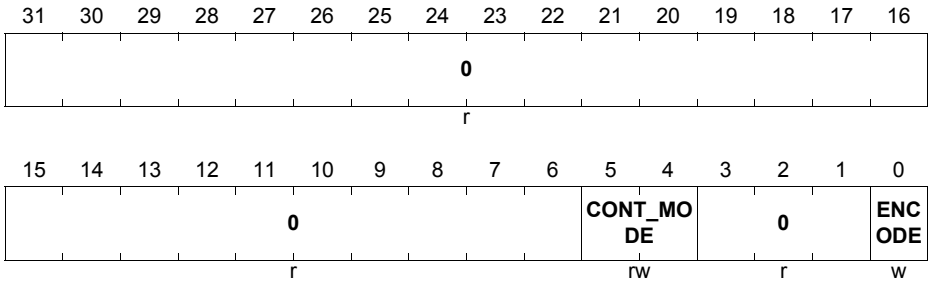
JPE Start Command To Start JFIF Stream Encoding Register

CIFJPE_ENCODE

JPE Start Command To Start JFIF Stream Encoding Register

(0004_H)

Reset Value: 0000 0000_H



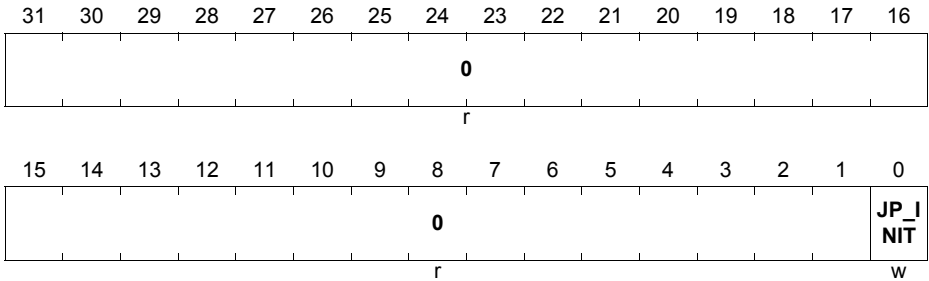
Field	Bits	Type	Description
CONT_MODE	5:4	rw	Encoder continuous mode "00": encoder stops at frame end (corresponds to former behavior)"01": encoder starts automatically to encode the next frame"10": unused"11": encoder first generates next header and then encodes automatically the next frame These settings are checked after encoding one frame. They are not auto-reset by hardware. Reset: 00 _B
ENCODE	0	w	Encode 1 _B Start command to start JFIF stream encoding; auto reset to zero after one clock cycle Reset: 0 _B
0	31:6, 3:1	r	Reserved Read as 0, should be written with 0.

JPE Automatic Configuration Update (INIT) Register

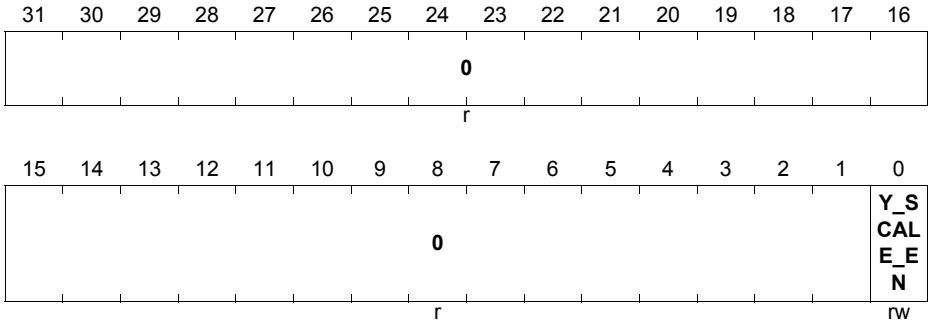
CIFJPE_INIT

**JPE Automatic Configuration Update Register
(0008_H)**

Reset Value: 0000 0000_H



Field	Bits	Type	Description
JP_INIT	0	w	JPEG Init This bit has to be set after “Encode” to start the JPEG encoder. The “Encode” command becomes active either with JP_INIT or with the input signal “CFG_UPD” auto reset to zero after one clock cycle !!! 1 _B Immediate start of JPEG encoder. Reset: 0 _B
0	31:1	r	Reserved Read as 0, should be written with 0.

JPE Y Value Scaling Control Register
CIFJPE_Y_SCALE_EN
JPE Y Value Scaling Control Register(000C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
Y_SCALE_EN	0	rw	Y scale flag 0 _B no Y input scaling 1 _B scaling Y input from [16..235] to [0..255] Reset: 0 _B
0	31:1	r	Reserved Read as 0, should be written with 0.

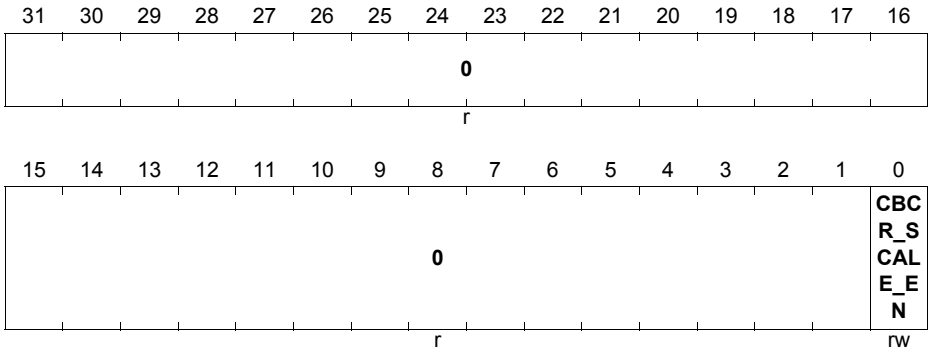
JPE Cb/Cr Value Scaling Control Register

CIFJPE_CBCR_SCALE_EN

JPE Cb/Cr Value Scaling Control Register

(0010_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CBCR_SCALE_EN	0	rw	Cb/Cr scale flag 0 _B no Cb/Cr input scaling 1 _B scaling Cb/Cr input from [16..240] to [0..255] Reset: 0 _B
0	31:1	r	Reserved Read as 0, should be written with 0.

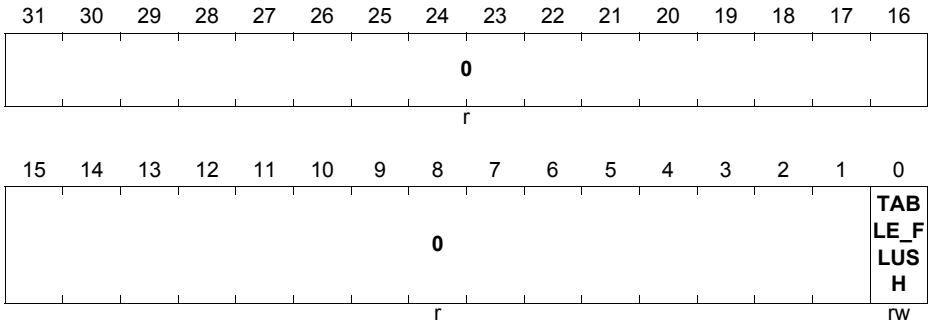
JPE Header Generation Debug Register

CIFJPE_TABLE_FLUSH

JPE Header Generation Debug Register

(0014_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TABLE_FLUSH	0	rw	Header generation debug control flag (controls transmission of last header bytes if the 64 bit output buffer is not completely filled) 0 _B wait for encoded image data to fill output buffer 1 _B immediately transmit last header bytes Reset: 0 _B
0	31:1	r	Reserved Read as 0, should be written with 0.

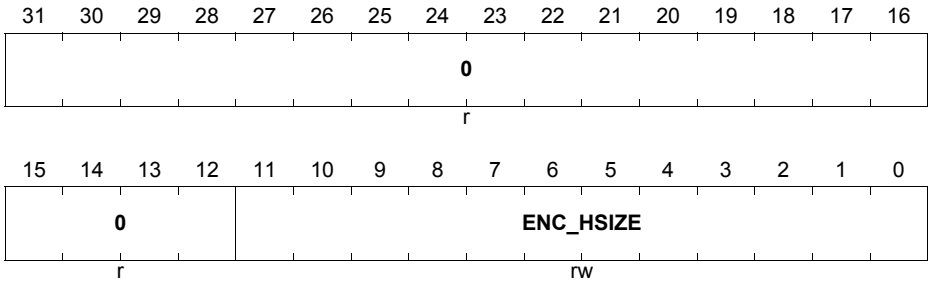
JPEG Codec Horizontal Image Size For Encoding Register

CIFJPE_ENC_HSIZE

JPEG Codec Horizontal Image Size For Encoding Register

(0018_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
ENC_HSIZE	11:0	rw	Horizontal Size JPEG codec horizontal image size for R2B and SGEN blocks Reset: 000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

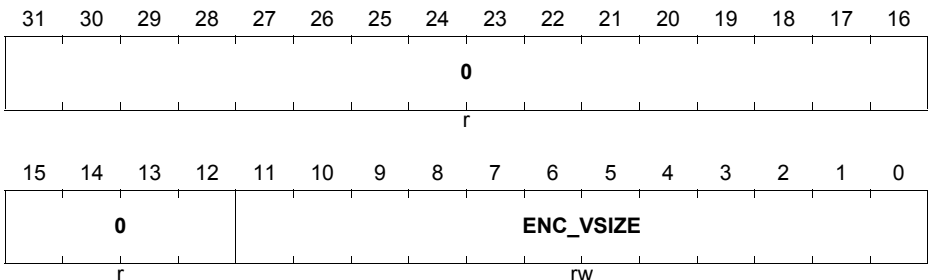
JPEG Codec Vertical Image Size For Encoding Register

CIFJPE_ENC_VSIZE

JPEG Codec Vertical Image Size For Encoding Register

(001C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
ENC_VSIZE	11:0	rw	Vertical Size JPEG codec vertical image size for R2B and SGEN blocks Reset: 000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

JPEG Picture Encoding Format Register

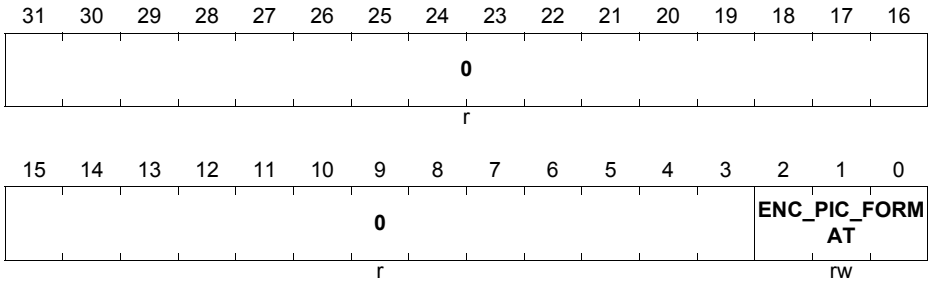
Register 313

CIFJPE_PIC_FORMAT

JPEG Picture Encoding Format Register

(0020_H)

Reset Value: 0000 0001_H



Field	Bits	Type	Description
ENC_PIC_FORMAT	2:0	rw	Picture Encoding Format 001 _B 4:2:2 format 1XX _B 4:0:0 format Reset: 001 _B
0	31:3	r	Reserved Read as 0, should be written with 0.

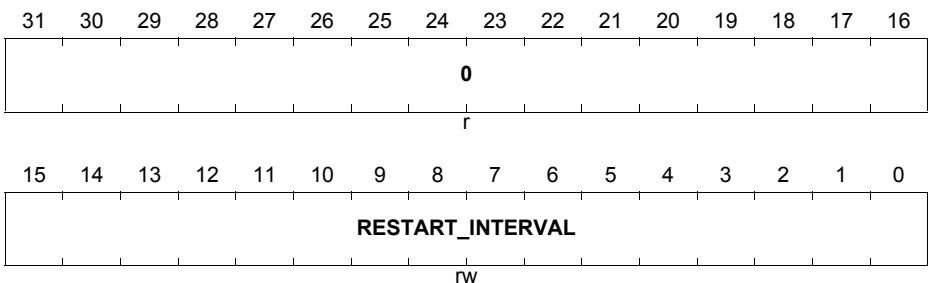
JPE Restart Marker Insertion Register

CIFJPE_RESTART_INTERVAL

JPE Restart Marker Insertion Register

(0024_H)

Reset Value: 0000 0000_H



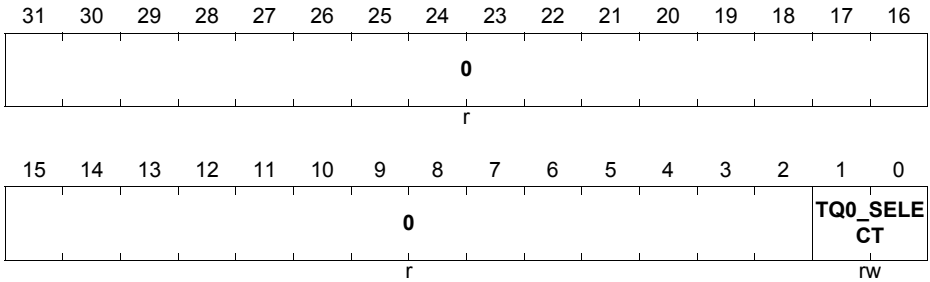
Field	Bits	Type	Description
RESTART_INTERVAL	15:0	rw	Restart Interval No of MCU in reset interval via host Reset: 0000 _H
0	31:16	r	Reserved Read as 0, should be written with 0.

Q- table Selector 0, Quant. Table For Y Component

CIFJPE_TQ_Y_SELECT

**Q- table Selector 0 Quant Table For Y Component
(0028_H)**

Reset Value: 0000 0000_H



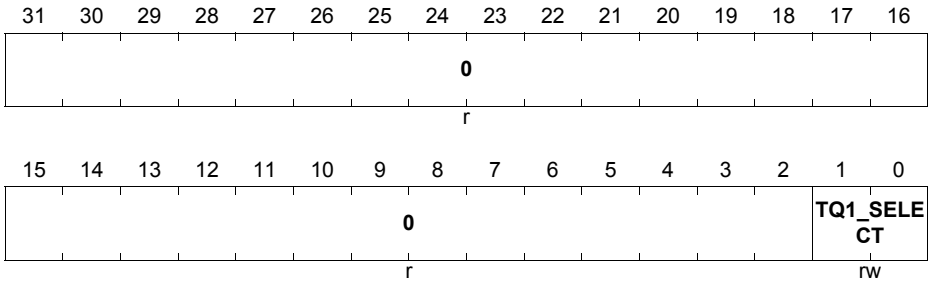
Field	Bits	Type	Description
TQ0_SELECT	1:0	rw	Q-Table Selector Y Q-table selector 0, Quant. For Y component. 00 _B Q-Table 0 01 _B Q-Table 1 10 _B Q-Table 2 11 _B Q-Table 3 Reset: 00 _B
0	31:2	r	Reserved Read as 0, should be written with 0.

Q- table selector 1, quant. table for U component

CIFJPE_TQ_U_SELECT

**Q- table Selector 1, Quant. Table For U Component
(002C_H)**

Reset Value: 0000 0001_H



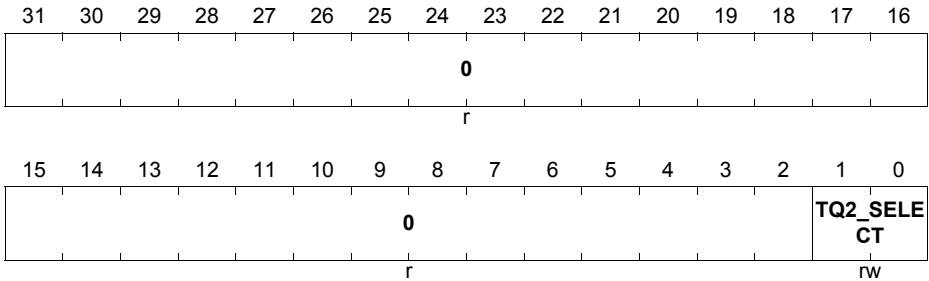
Field	Bits	Type	Description
TQ1_SELECT	1:0	rw	Q-Table Selector U Q-table selector 1, Quant. For U component. 00 _B Q-Table 0 01 _B Q-Table 1 10 _B Q-Table 2 11 _B Q-Table 3 Reset: 00 _B
0	31:2	r	Reserved Read as 0, should be written with 0.

Q- table selector 2, quant. table for V component

CIFJPE_TQ_V_SELECT

**Q- table Selector 2 Quant Table For V Component
(0030_H)**

Reset Value: 0000 0001_H



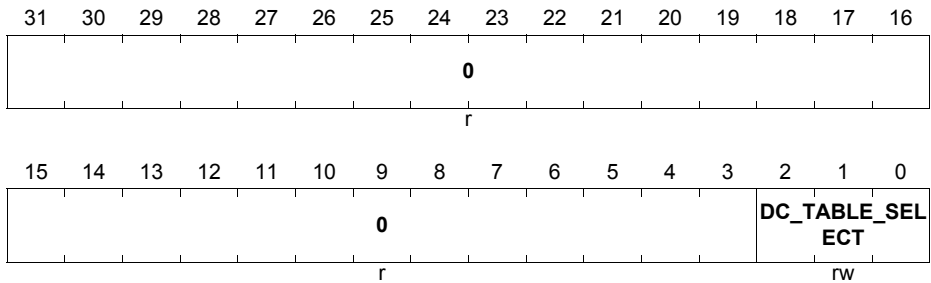
Field	Bits	Type	Description
TQ2_SELECTOR	1:0	rw	Q-Table Selector V Q-table selector 2, Quant. For V component. 00 _B Q-Table 0 01 _B Q-Table 1 10 _B Q-Table 2 11 _B Q-Table 3 Reset: 00 _B
0	31:2	r	Reserved Read as 0, should be written with 0.

JPE Huffman Table Selector For DC Values

CIFJPE_DC_TABLE_SELECT

JPE Huffman Table Selector For DC Values Register
(0034_H)

Reset Value: 0000 0006_H



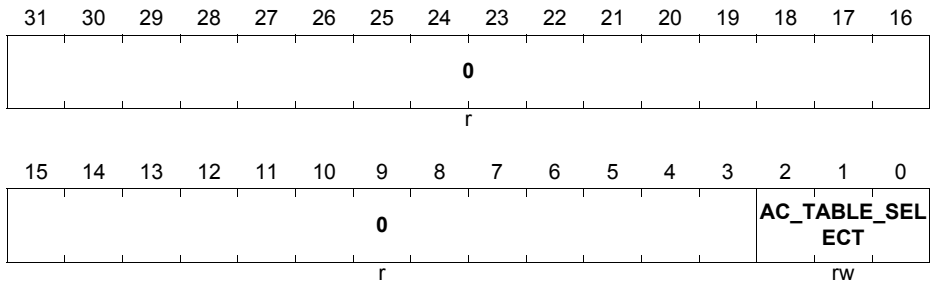
Field	Bits	Type	Description
DC_TABLE_SELECTOR	2:0	rw	DC Table Selector XX0 _B dc table 0; component 0 XX1 _B dc table 1; component 0 X0X _B dc table 0; component 1 X1X _B dc table 1; component 1 0XX _B dc table 0; component 2 1XX _B dc table 1; component 2 Reset: 110 _B
0	31:3	r	Reserved Read as 0, should be written with 0.

JPE Huffman Table Selector For AC Values Register

CIFJPE_AC_TABLE_SELECT

JPE Huffman Table Selector For AC Values Register
(0038_H)

Reset Value: 0000 0006_H



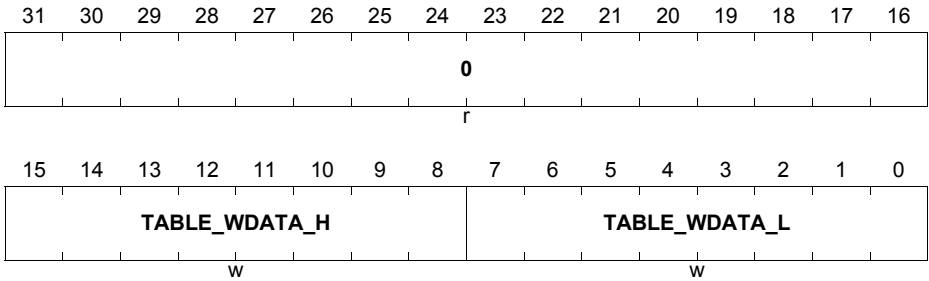
Field	Bits	Type	Description
AC_TABLE_SELECT	2:0	rw	AC Table Selector "xx0" = dc table 0; component 0 "xx1" = dc table 1; component 0 "x0x" = dc table 0; component 1 "x1x" = dc table 1; component 1 "0xx" = dc table 0; component 2 "1xx" = dc table 1; component 2 Reset: 110 _B
0	31:3	r	Reserved Read as 0, should be written with 0.

JPE Table Programming Register

CIFJPE_TABLE_DATA

JPE Table Programming Register (003C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TABLE_WDAT A _H	15:8	w	Table data MSB Reset: 00 _H
TABLE_WDAT A _L	7:0	w	Table data LSB Reset: 00 _H
0	31:16	r	Reserved Read as 0, should be written with 0.

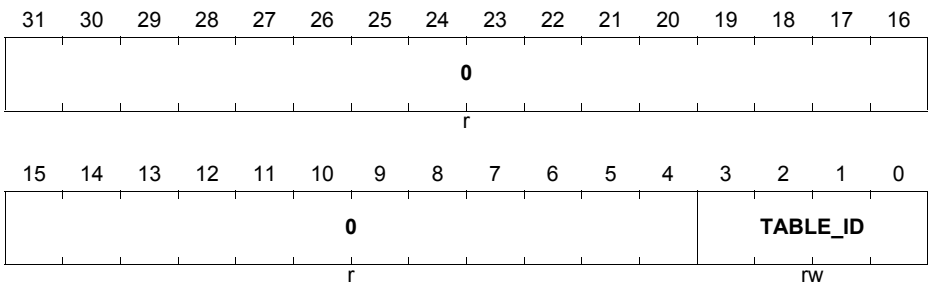
JPE Table Programming Select Register

CIFJPE_TABLE_ID

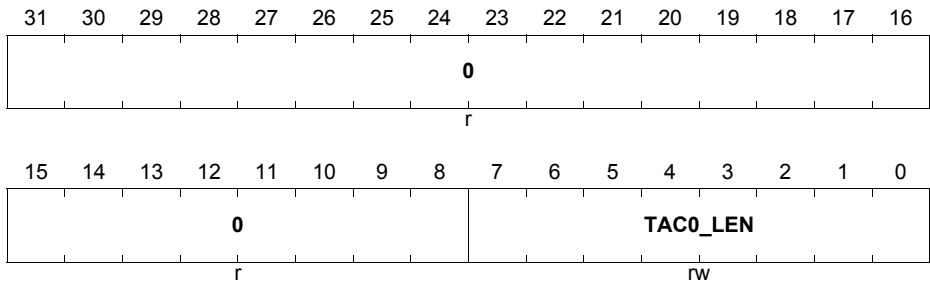
JPE Table Programming Select Register

(0040_H)

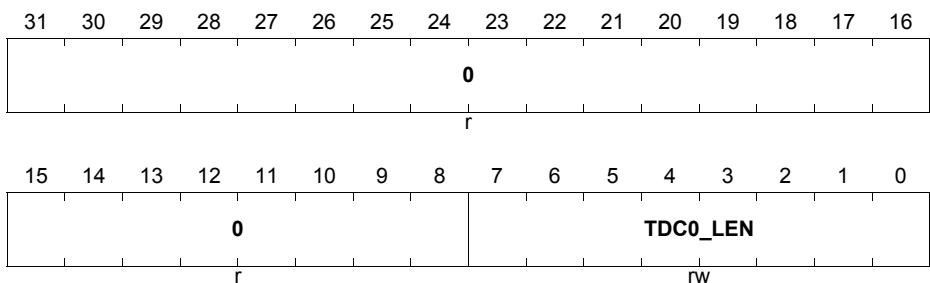
Reset Value: 0000 0000_H



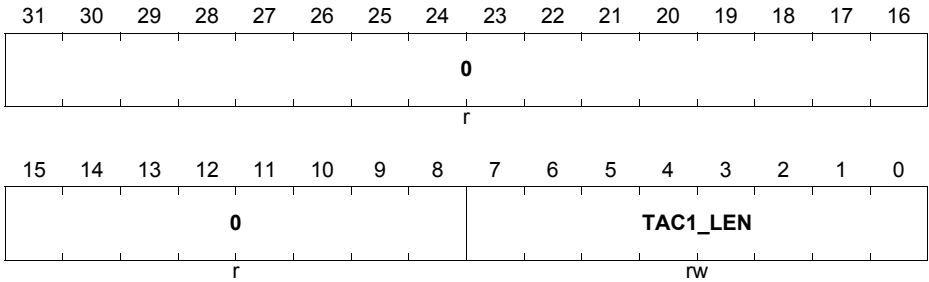
Field	Bits	Type	Description
TABLE_ID	3:0	rw	JPE Table ID Select table according to the following list. Values 8 _H through F _H are reserved for debug purposes. 0 _H Q-Table 0 1 _H Q-Table 1 2 _H Q-Table 2 3 _H Q-Table 3 4 _H VLC DC Table 0 5 _H VLC AC Table 0 6 _H VLC DC Table 1 7 _H VLC AC Table 1 Reset: 0000 _B
0	31:4	r	Reserved Read as 0, should be written with 0.

JPE Huffman AC Table 0 Length Register
CIFJPE_TAC0_LEN
JPE Huffman AC Table 0 Length Register
(0044_H)
Reset Value: 0000 00B_H


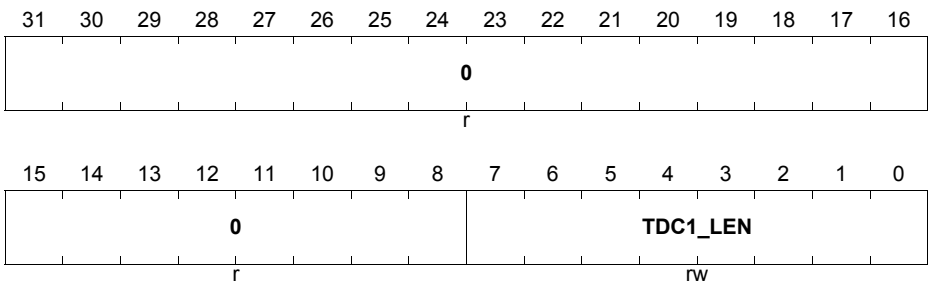
Field	Bits	Type	Description
TAC0_LEN	7:0	rw	AC Table 0 Length Huffman table length for ac0 table Reset: B2 _H
0	31:8	r	Reserved Read as 0, should be written with 0.

JPE Huffman DC Table 0 Length Register
CIFJPE_TDC0_LEN
JPE Huffman DC Table 0 Length Register
(0048_H)
Reset Value: 0000 001C_H


Field	Bits	Type	Description
TDC0_LEN	7:0	rw	DC Table 0 Length Huffman table length for dc0 table Reset: 1C _H
0	31:8	r	Reserved Read as 0, should be written with 0.

JPE Huffman AC Table 1 Length Register
CIFJPE_TAC1_LEN
JPE Huffman AC Table 1 Length Register
(004C_H)
Reset Value: 0000 00B_H


Field	Bits	Type	Description
TAC1_LEN	7:0	rw	AC Table 1 Length Huffman table length for ac1 table Reset: B2 _H
0	31:8	r	Reserved Read as 0, should be written with 0.

JPE Huffman DC Table 1 Length Register
CIFJPE_TDC1_LEN
JPE Huffman DC Table 1 Length Register
(0050_H)
Reset Value: 0000 001C_H


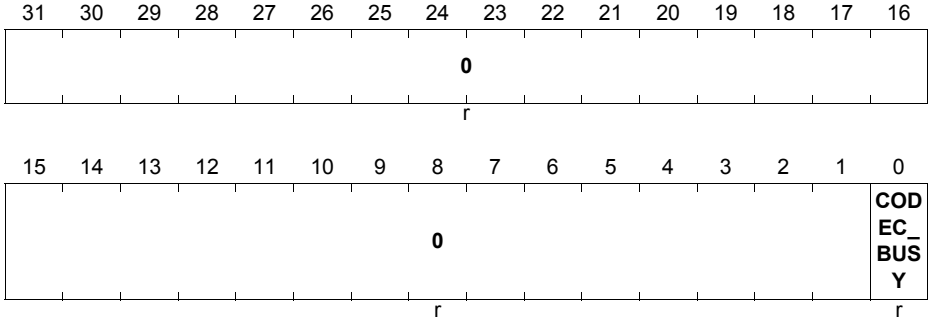
Field	Bits	Type	Description
TDC1_LEN	7:0	rw	DC Table 1 Length Huffman table length for dc1 table Reset: $1C_H$
0	31:8	r	Reserved Read as 0, should be written with 0.

JPE Encoder Status Flag Register

CIFJPE_ENCODER_BUSY

JPE Encoder Status Flag Register (0058_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
CODEC_BUSY	0	r	Codec Busy 0 _B Codec is free (not busy) 1 _B JPEG codec in process Reset: 0 _B
0	31:1	r	Reserved Read as 0.

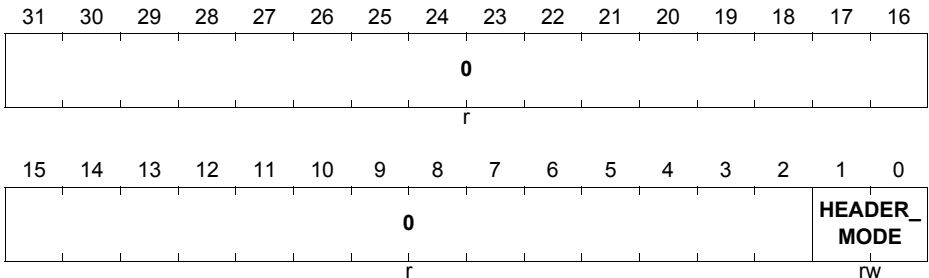
JPE Header Mode Definition Register

CIFJPE_HEADER_MODE

JPE Header Mode Definition Register

(005C_H)

Reset Value: 0000 0000_H

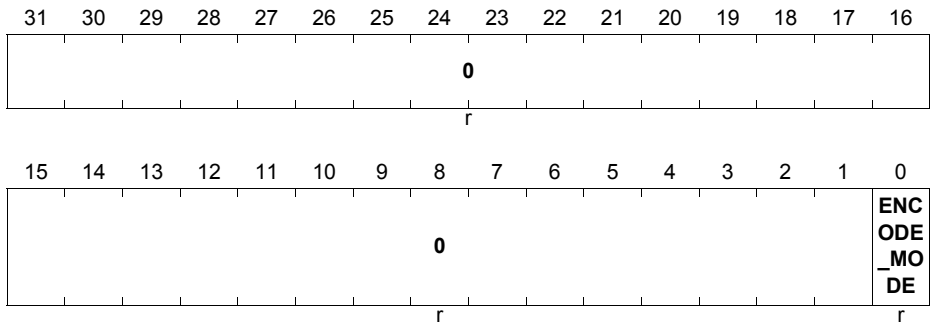


Field	Bits	Type	Description
HEADER_MODE	1:0	rw	Header Mode 00 _B no APPn header 01 _B don't use this setting 10 _B JFIF header 11 _B don't use this setting Reset: 00 _B
0	31:2	r	Reserved Read as 0, should be written with 0.

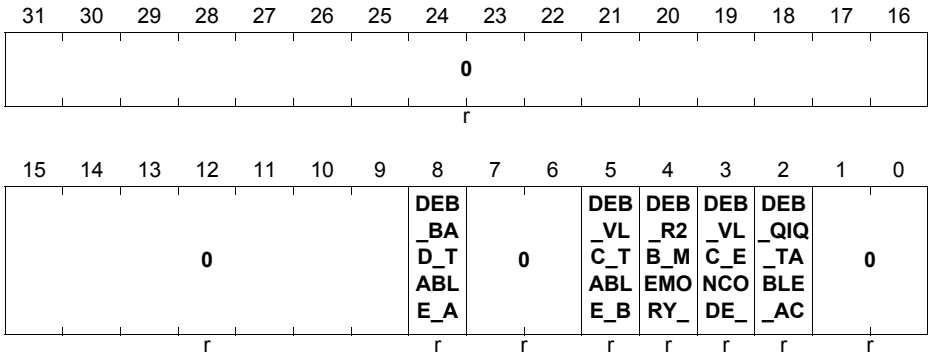
JPE Encode Mode Register

CIFJPE_ENCODE_MODE

JPE Encode Mode Register (0060_H) Reset Value: 0000 0001_H

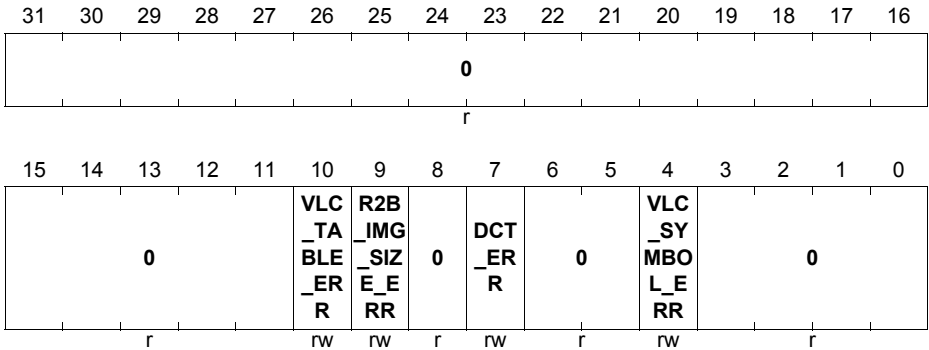


Field	Bits	Type	Description
ENCODE_MODE	0	r	Encode Mode Always 1, because this is the encoder only edition Reset: 1 _B
0	31:1	r	Reserved Read as 0.

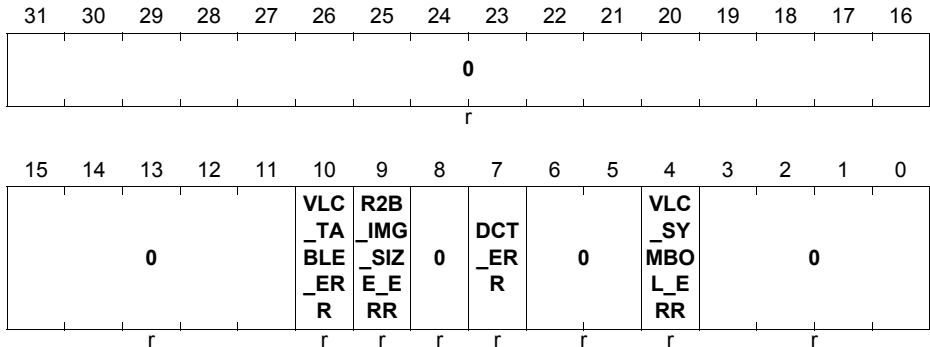
JPE Debug Information Register
CIFJPE_DEBUG
JPE Debug Information Register
(0064_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
DEB_BAD_TABLE_ACCESS	8	r	Debug Bad Table Access Debug signal only (set if an access to the TABLE_DATA or to the TABLE_ID register is performed, when the JPEG_ENCODER is busy. In this case a default handshake acknowledge is generated. Thus the configuration bus is not blocked) Reset: 0 _B
DEB_VLC_TABLE_BUSY	5	r	Debug VLC Table Busy Debug signal only (vlc access to huff-tables) Unit will initialize huff tables internally, busy for ~400 cycles after reset. Reset: 0 _B
DEB_R2B_MEMORY_FULL	4	r	R2B Memory Full Debug signal only (line memory status of r2b) Reset: 0 _B

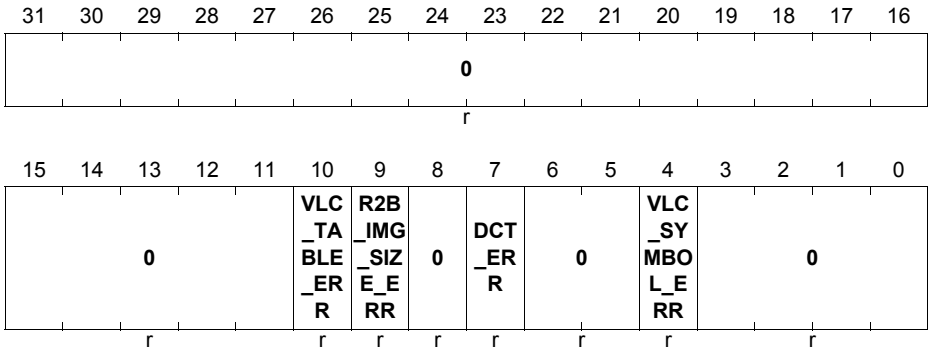
Field	Bits	Type	Description
DEB_VLC_ENCODE_B USY	3	r	VLC Encode Busy Debug signal only (vlc encode processing active) Reset: 0 _B
DEB_QIQ_TABLE_AC C	2	r	QIQ Table Access Debug signal only (QIQ table access) Reset: 0 _B
0	31:9, 7:6, 1:0	r	Reserved Read as 0.

JPEG Encoder Interrupt Registers
JPE Error Interrupt Mask Register
CIFJPE_ERROR_IMR
JPE Error Interrupt Mask Register (0068_H)
Reset Value: 0000 0000_H


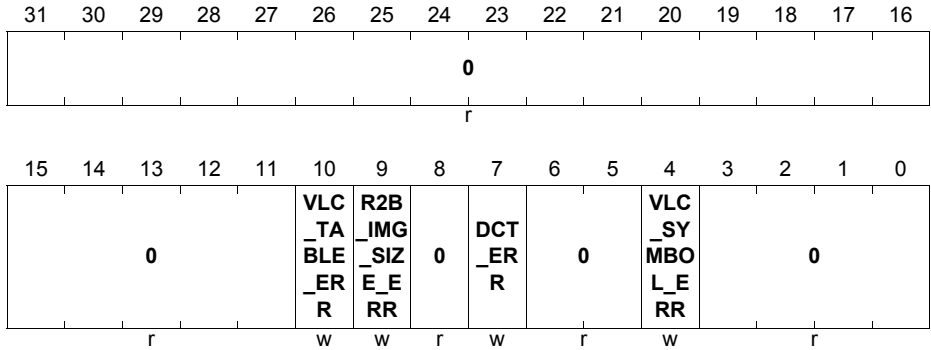
Field	Bits	Type	Description
VLC_TABLE_ERR	10	rw	VLC Table Error 1 _B interrupt is activated (masked in) Reset: 0 _B
R2B_IMG_SIZE_ERR	9	rw	R2B Image Size Error 1 _B interrupt is activated (masked in) Reset: 0 _B
DCT_ERR	7	rw	DC Table Error 1 _B interrupt is activated (masked in) Reset: 0 _B
VLC_SYMBOL_ERR	4	rw	VLC Symbol Error 1 _B interrupt is activated (masked in) Reset: 0 _B
0	31:11, 8, 6:5, 3:0	r	Reserved Read as 0, should be written with 0.

JPE Error Raw Interrupt Status Register
CIFJPE_ERROR_RIS
JPE Error Raw Interrupt Status Register
(006C_H)
Reset Value: 0000 0000_H


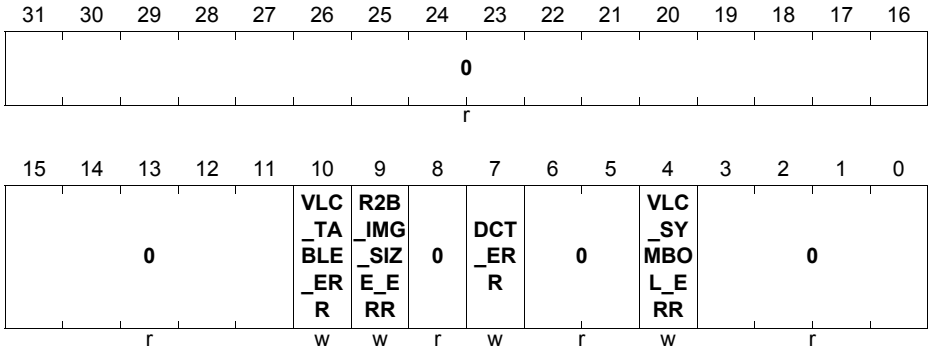
Field	Bits	Type	Description
VLC_TABLE_ERR	10	r	VLC Table Error 1 _B illegal table detected
R2B_IMG_SIZE_ERR	9	r	R2B Image Size Error 1 _B mismatch of predefined h_size and v_size values with calculated values (encode mode)
DCT_ERR	7	r	DC Table Error 1 _B block start mismatch
VLC_SYMBOL_ERR	4	r	VLC Symbol Error 1 _B illegal symbol detected (encoding)
0	31:11, 8, 6:5, 3:0	r	Reserved Read as 0

JPE Error Masked Interrupt Status Register
CIFJPE_ERROR_MIS
**JPE Error Masked Interrupt Status Register
(0070_H)**
Reset Value: 0000 0000_H


Field	Bits	Type	Description
VLC_TABLE_ERR	10	r	VLC Table Error 1 _B illegal table detected Reset: 0 _B
R2B_IMG_SIZE_ERR	9	r	R2B Image Size Error 1 _B mismatch of predefined h_size and v_size values with calculated values (encode mode) Reset: 0 _B
DCT_ERR	7	r	DC Table Error 1 _B block start mismatch Reset: 0 _B
VLC_SYMBOL_ERR	4	r	VLC Symbol Error 1 _B illegal symbol detected (encoding) Reset: 0 _B
0	31:11, 8, 6:5, 3:0	r	Reserved Read as 0.

JPE Error Interrupt Set Register
CIFJPE_ERROR_ISR
JPE Error Interrupt Set Register
(0078_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
VLC_TABLE_ERR	10	w	VLC Table Error 1 _B set error bit, bit is reset to zero after 1 clk Reset: 0 _B
R2B_IMG_SIZE_ERR	9	w	R2B Image Size Error 1 _B set error bit, bit is reset to zero after 1 clk Reset: 0 _B
DCT_ERR	7	w	DC Table Error 1 _B set error bit, bit is reset to zero after 1 clk Reset: 0 _B
VLC_SYMBOL_ERR	4	w	VLC Symbol Error 1 _B set error bit, bit is reset to zero after 1 clk Reset: 0 _B
0	31:11, 8, 6:5, 3:0	r	Reserved Read as 0, should be written with 0.

JPE Error Interrupt Clear Register
CIFJPE_ERROR_ICR
JPE Error Interrupt Clear Register (0074_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
VLC_TABLE_ERR	10	w	VLC Table Error 1 _B set error bit, bit is reset to zero after 1 clk Reset: 0 _B
R2B_IMG_SIZE_ERR	9	w	R2B Image Size Error 1 _B set error bit, bit is reset to zero after 1 clk Reset: 0 _B
DCT_ERR	7	w	DC Table Error 1 _B set error bit, bit is reset to zero after 1 clk Reset: 0 _B
VLC_SYMBOL_ERR	4	w	VLC Symbol Error 1 _B set error bit, bit is reset to zero after 1 clk Reset: 0 _B
0	31:11, 8, 6:5, 3:0	r	Reserved Read as 0, should be written with 0.

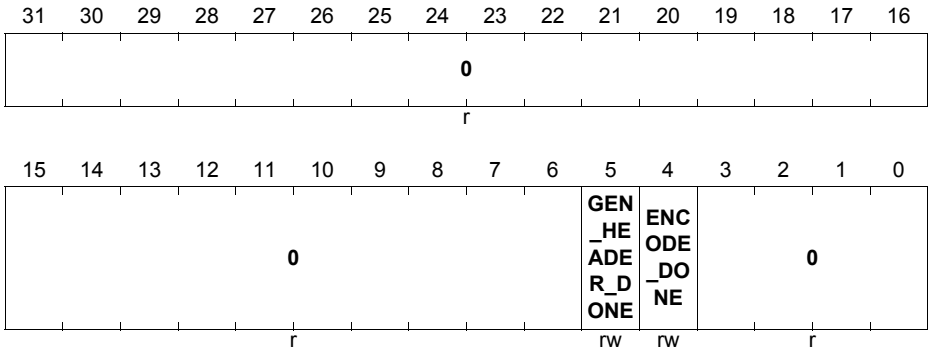
JPEG Status Interrupt Mask Register

CIFJPE_STATUS_IMR

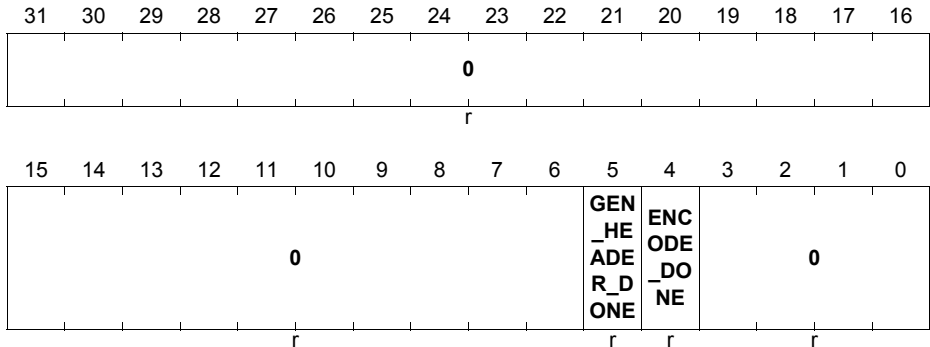
JPEG Status Interrupt Mask Register

(007C_H)

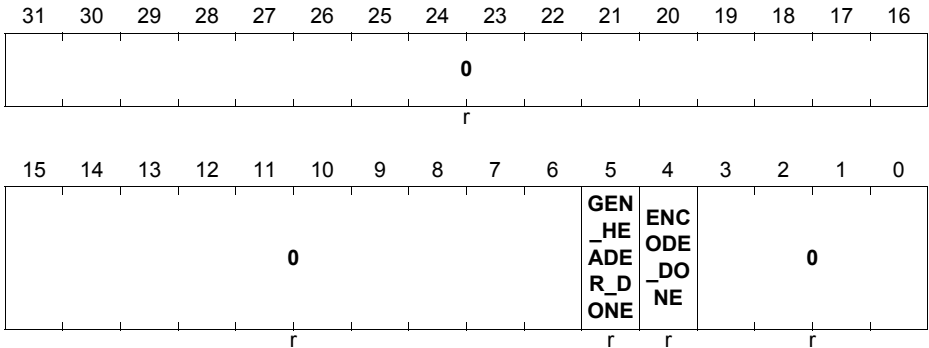
Reset Value: 0000 0000_H



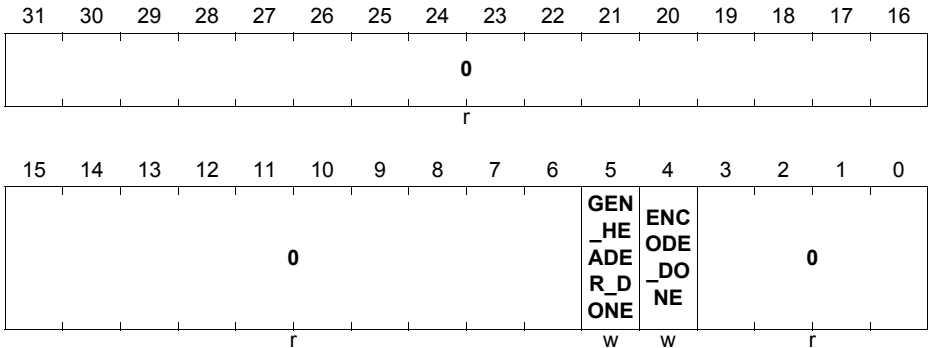
Field	Bits	Type	Description
GEN_HEADER_DONE	5	rw	Header Generation Complete 1 _B interrupt is activated (masked in) Reset: 0 _B
ENCODE_DONE	4	rw	Encoding Complete 1 _B interrupt is activated (masked in) Reset: 0 _B
0	31:6, 3:0	r	Reserved Read as 0, should be written with 0.

JPEG Status Raw Interrupt Status Register
CIFJPE_STATUS_RIS
JPEG Status Raw Interrupt Status Register
(0080_H)
Reset Value: 0000 0000_H


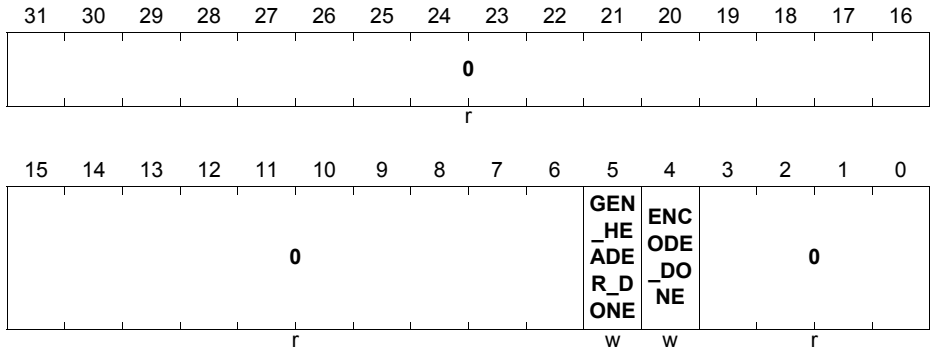
Field	Bits	Type	Description
GEN_HEADER_DONE	5	r	Header Generation Complete 1 _B Stream header generation finished Reset: 0 _B
ENCODE_DONE	4	r	Encoding Complete 1 _B Encode processing finished Reset: 0 _B
0	31:6, 3:0	r	Reserved Read as 0.

JPEG Status Masked Interrupt Status Register
CIFJPE_STATUS_MIS
**JPEG Status Masked Interrupt Status Register
(0084_H)**
Reset Value: 0000 0000_H


Field	Bits	Type	Description
GEN_HEADER_DONE	5	r	Header Generation Complete 1 _B Stream header generation finished Reset: 0 _B
ENCODE_DONE	4	r	Encoding Complete 1 _B Encode processing finished Reset: 0 _B
0	31:6, 3:0	r	Reserved Read as 0.

JPEG Status Interrupt Clear Register
CIFJPE_STATUS_ICR
JPEG Status Interrupt Clear Register
(0088_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
GEN_HEADER_DONE	5	w	Header Generation Complete 1 _B clear status bit, bit is reset to zero after 1 clk Reset: 0 _B
ENCODE_DONE	4	w	Encoding Complete 1 _B clear status bit, bit is reset to zero after 1 clk Reset: 0 _B
0	31:6, 3:0	r	Reserved Read as 0, should be written with 0.

JPEG Status Interrupt Set Register
CIFJPE_STATUS_ISR
JPEG Status Interrupt Set Register (008C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
GEN_HEADER_DONE	5	w	Header Generation Complete 1 _B set error bit, bit is reset to zero after 1 clk Reset: 0 _B
ENCODE_DONE	4	w	Encoding Complete 1 _B set error bit, bit is reset to zero after 1 clk Reset: 0 _B
0	31:6, 3:0	r	Reserved Read as 0, should be written with 0.

36.1.3.7 Security Watchdog Programming Registers

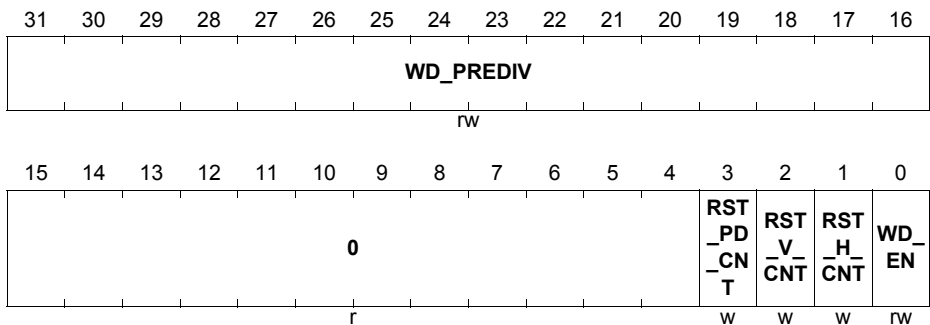
The address of each CIF Security Watchdog register is evaluated as CIF_WATCHDOG_BASE + Offset.

Watchdog Configuration Registers

Watchdog Control Register

CIFWD_CTRL

Watchdog Control Register (0000_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
WD_PREDIV	31:16	rw	Watchdog Counter Predivider A value of 0 means that the Watchdog Counters are increased with every CIF clock cycle. Every other value N leads to an increment at every N+1th cycle. Reset: 00 _H
RST_PD_CNT	3	w	Reset Predivider Counter 0 _B nothing happens 1 _B reset counter
RST_V_CNT	2	w	Reset Vertical Counter 0 _B nothing happens 1 _B reset counter
RST_H_CNT	1	w	Reset Horizontal Counter 0 _B nothing happens 1 _B reset counter

Field	Bits	Type	Description
WD_EN	0	rw	Enable Security Watchdog 0 _B unit is disabled 1 _B enables the watchdog unit
0	15:4	r	Reserved Read as 0, should be written with 0.

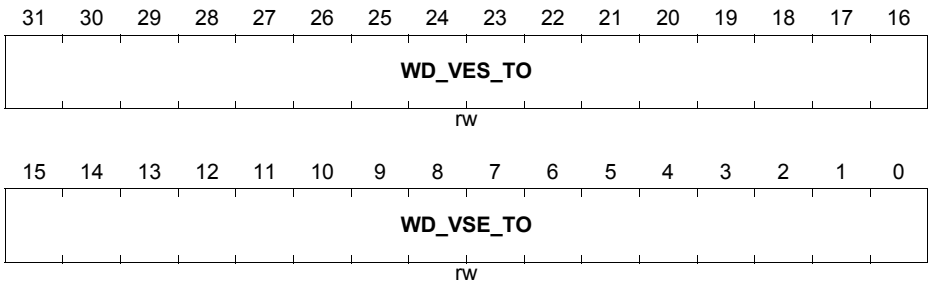
Watchdog Vertical Timeout Register

CIFWD_V_TIMEOUT

Watchdog Vertical Timeout Register

(0004_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
WD_VES_T O	31:16	rw	Watchdog Vertical End Start Timeout A value of 0 means that the Timeout is disabled. Every other value leads to an interrupt trigger when the vertical counter reaches the value in the vertical End Start phase. Reset: 00 _H
WD_VSE_T O	15:0	rw	Watchdog Vertical Start End Timeout A value of 0 means that the Timeout is disabled. Every other value leads to an interrupt trigger when the vertical counter reaches the value in the vertical Start End phase. Reset: 00 _H

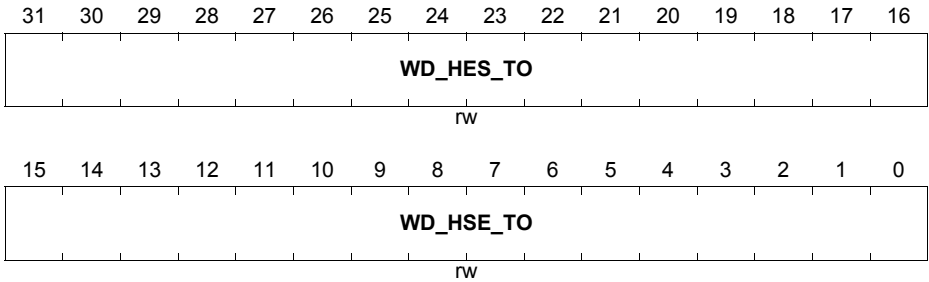
Watchdog Horizontal Timeout Register

CIFWD_H_TIMEOUT

Watchdog Horizontal Timeout Register

(0008_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
WD_HES_TO	31:16	rw	Watchdog Horizontal End Start Timeout A value of 0 means that the Timeout is disabled. Every other value leads to an interrupt trigger when the horizontal counter reaches the value in the horizontal End Start phase. Reset: 00 _H
WD_HSE_TO	15:0	rw	Watchdog Horizontal Start End Timeout A value of 0 means that the Timeout is disabled. Every other value leads to an interrupt trigger when the horizontal counter reaches the value in the horizontal Start End phase. Reset: 00 _H

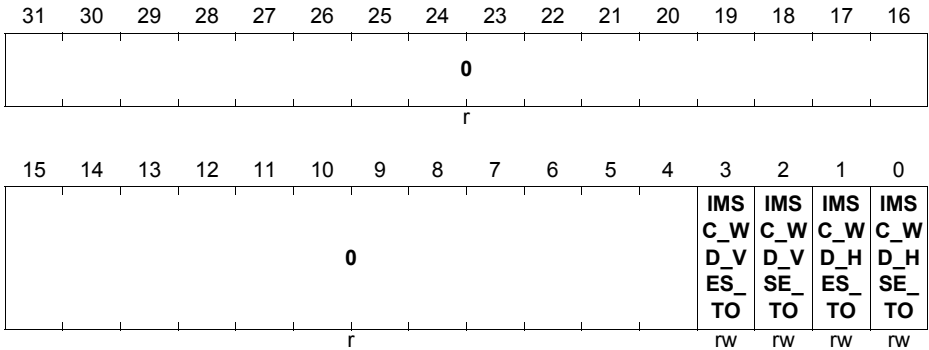
Watchdog Interrupt Registers

Watchdog Interrupt Mask Register

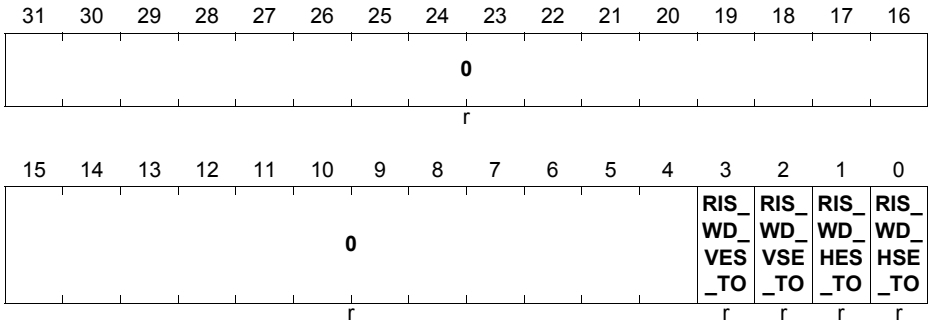
CIFWD_IMSC

Watchdog Interrupt Mask Register (000C_H)

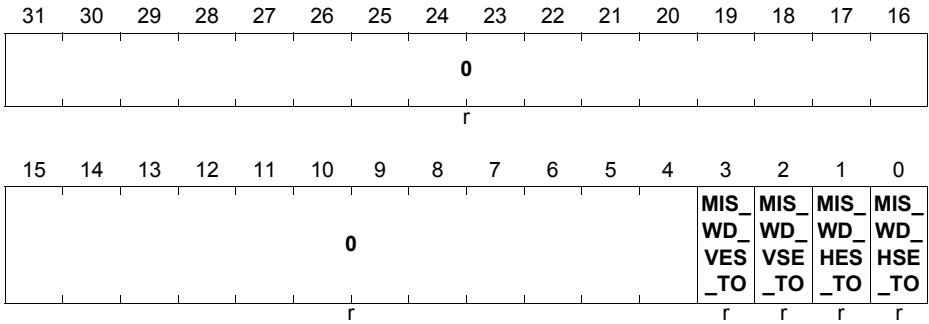
Reset Value: 0000 0000_H



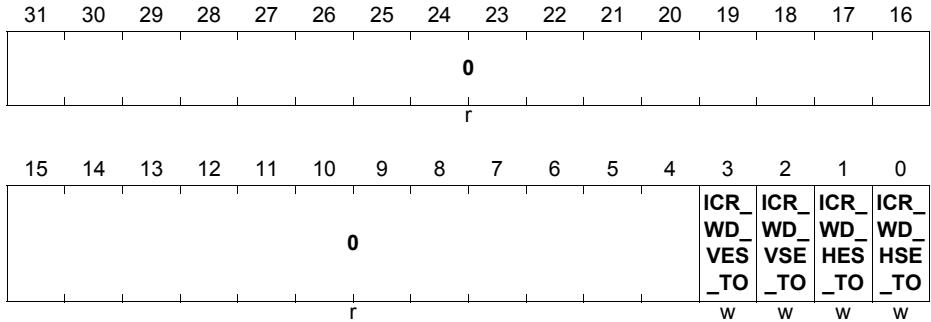
Field	Bits	Type	Description
IMSC_WD_VES_TO	3	rw	Vertical End Start Timeout Enable interrupt (1) or mask out (0) Reset: 0 _B
IMSC_WD_VSE_TO	2	rw	Vertical Start End Timeout Enable interrupt (1) or mask out (0) Reset: 0 _B
IMSC_WD_HES_TO	1	rw	Horizontal End Start Timeout Enable interrupt (1) or mask out (0) Reset: 0 _B
IMSC_WD_HSE_TO	0	rw	Horizontal Start End Timeout Enable interrupt (1) or mask out (0) Reset: 0 _B
0	31:4	r	Reserved Read as 0, should be written with 0.

Watchdog Raw Interrupt Status Register
CIFWD_RIS
Watchdog Raw Interrupt Status Register
(0010_H)
Reset Value: 0000 0000_H


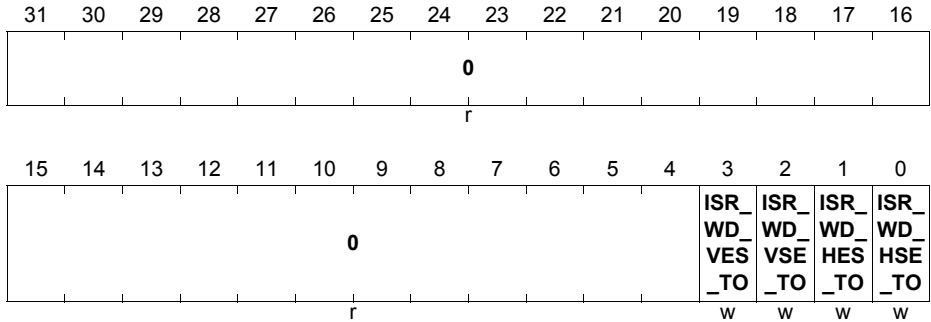
Field	Bits	Type	Description
RIS_WD_VES_TO	3	r	Vertical End Start Timeout The counter reached the programmed timeout. Reset: 0 _B
RIS_WD_VSE_TO	2	r	Vertical Start End Timeout The counter reached the programmed timeout. Reset: 0 _B
RIS_WD_HES_TO	1	r	Horizontal End Start Timeout The counter reached the programmed timeout. Reset: 0 _B
RIS_WD_HSE_TO	0	r	Horizontal Start End Timeout The counter reached the programmed timeout. Reset: 0 _B
0	31:4	r	Reserved Read as 0.

Watchdog Masked Interrupt Status Register
CIFWD_MIS
**Watchdog Masked Interrupt Status Register
(0014_H)**
Reset Value: 0000 0000_H


Field	Bits	Type	Description
MIS_WD_VES_TO	3	r	Vertical End Start Timeout The counter reached the programmed timeout. Reset: 0 _B
MIS_WD_VSE_TO	2	r	Vertical Start End Timeout The counter reached the programmed timeout. Reset: 0 _B
MIS_WD_HES_TO	1	r	Horizontal End Start Timeout The counter reached the programmed timeout. Reset: 0 _B
MIS_WD_HSE_TO	0	r	Horizontal Start End Timeout The counter reached the programmed timeout. Reset: 0 _B
0	31:4	r	Reserved Read as 0.

Watchdog Interrupt Clear Register
CIFWD_ICR
Watchdog Interrupt Clear Register (0018_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
ICR_WD_VES_T O	3	w	Vertical End Start Timeout 0 _B nothing happens 1 _B clear interrupt
ICR_WD_VSE_T O	2	w	Vertical Start End Timeout 0 _B nothing happens 1 _B clear interrupt
ICR_WD_HES_T O	1	w	Horizontal End Start Timeout 0 _B nothing happens 1 _B clear interrupt
ICR_WD_HSE_T O	0	w	Horizontal Start End Timeout 0 _B nothing happens 1 _B clear interrupt
0	31:4	r	Reserved Read as 0, should be written with 0.

Watchdog Interrupt Set Register
CIFWD_ISR
Watchdog Interrupt Set Register
(001C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
ISR_WD_VES_TO	3	w	Vertical End Start Timeout 0 _B nothing happens 1 _B set interrupt
ISR_WD_VSE_TO	2	w	Vertical Start End Timeout 0 _B nothing happens 1 _B set interrupt
ISR_WD_HES_TO	1	w	Horizontal End Start Timeout 0 _B nothing happens 1 _B set interrupt
ISR_WD_HSE_TO	0	w	Horizontal Start End Timeout 0 _B nothing happens 1 _B set interrupt
0	31:4	r	Reserved Read as 0, should be written with 0.

36.1.3.8 ISP Image Stabilization Registers

The address of each CIF image stabilization register is evaluated as CIF_IS_BASE + Offset.

Image Stabilization Control Registers

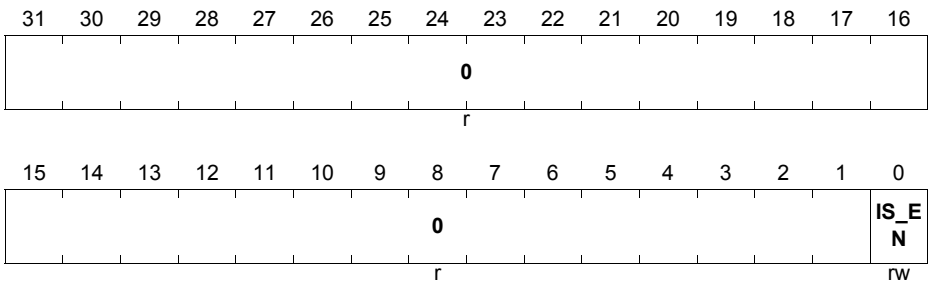
ISP Image Stabilization Control Register

CIFISPIS_CTRL

ISP Image Stabilization Control Register

(0000_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
IS_EN	0	rw	Image Stabilization Enable 0 _B image stabilization switched off 1 _B image stabilization switched on Reset: 0 _B
0	31:1	r	Reserved Read as 0, should be written with 0.

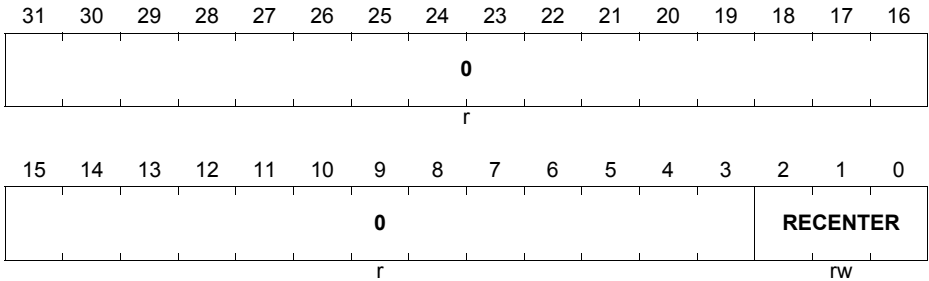
ISP Image Stabilization Recenter Register

CIFISPIS_RECENTER

ISP Image Stabilization Recenter Register

(0004_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RECENTER	2:0	rw	Recenter 000 _B recenter feature switched off For all other values recentering is active (cur_h/v_offs-H/V_OFFS)/2 pöower(recenter) Reset: 000 _B
0	31:3	r	Reserved Read as 0, should be written with 0.

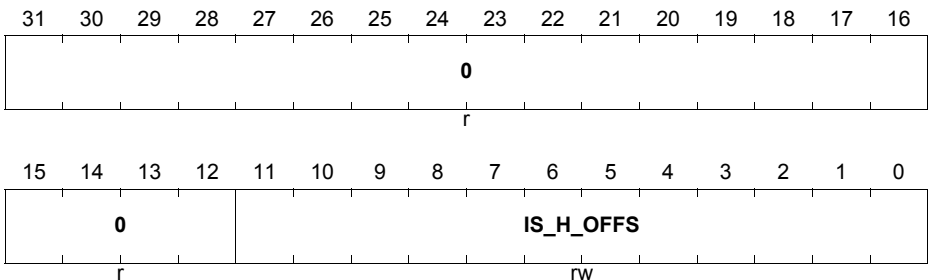
ISP Image Stabilization Horizontal Offset Of Output Window Register

CIFISPIS_H_OFFS

ISP Image Stabilization Horizontal Offset Of Output Window Register

(0008_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
IS_H_OFFS	11:0	rw	Horizontal Picture Offset Horizontal picture offset in pixel Reset: 0000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

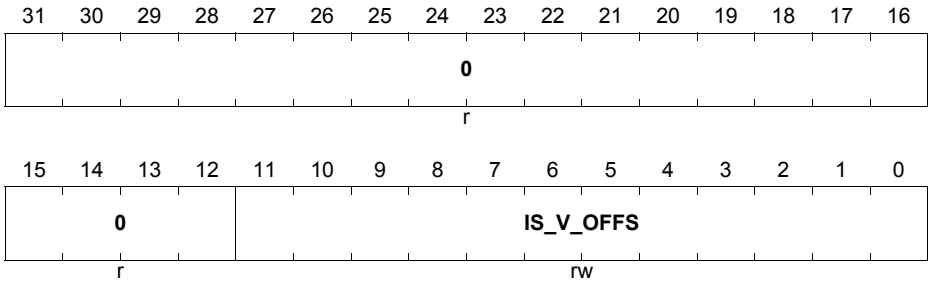
ISP Image Stabilization Vertical Offset Of Output Window Register

CIFISPIS_V_OFFS

ISP Image Stabilization Vertical Offset Of Output Window Register

(000C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
IS_V_OFFS	11:0	rw	Vertical Picture Offset Vertical picture offset in lines Reset: 000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

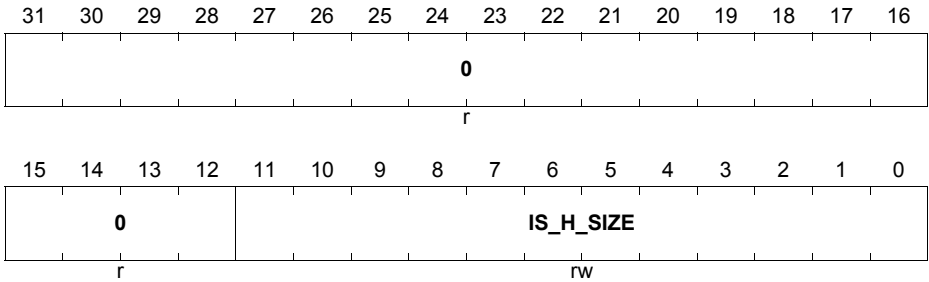
ISP Image Stabilization Output Horizontal Picture Size Register

CIFISPIS_H_SIZE

ISP Image Stabilization Output Horizontal Picture Size Register

(0010_H)

Reset Value: 0000 0A28_H



Field	Bits	Type	Description
IS_H_SIZE	11:0	rw	<p>Horizontal Picture Size Horizontal picture size in pixel Only even numbers are accepted, because complete quadruples of YUYV(YCbYCr) are needed for the following modules. If an odd size is programmed the value will be truncated to even size. If ISP_MODE is set to 001_B (ITU-R BT.656 YUV) 010_B (ITU-R BT.601 YUV) 011_B (ITU-R BT.601 Bayer RGB) 101_B (ITU-R BT.656 Bayer RGB) Reset: A28_H</p>
0	31:12	r	<p>Reserved Read as 0, should be written with 0.</p>

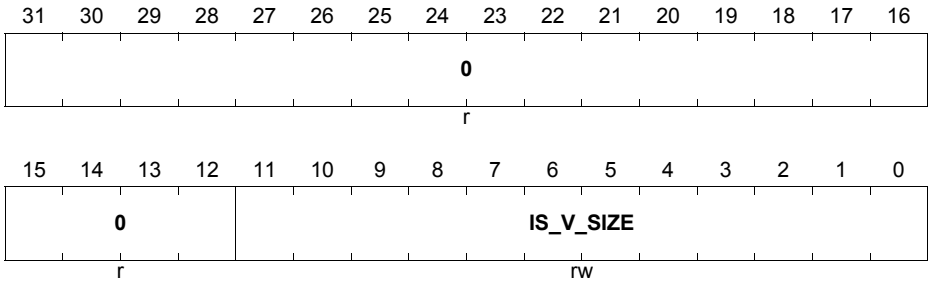
ISP Image Stabilization Output Vertical Picture Size Register

CIFISPIS_V_SIZE

ISP Image Stabilization Output Vertical Picture Size Register

(0014_H)

Reset Value: 0000 0800_H



Field	Bits	Type	Description
IS_V_SIZE	11:0	rw	Vertical Picture Size Vertical picture size in lines Reset: 800 _H
0	31:12	r	Reserved

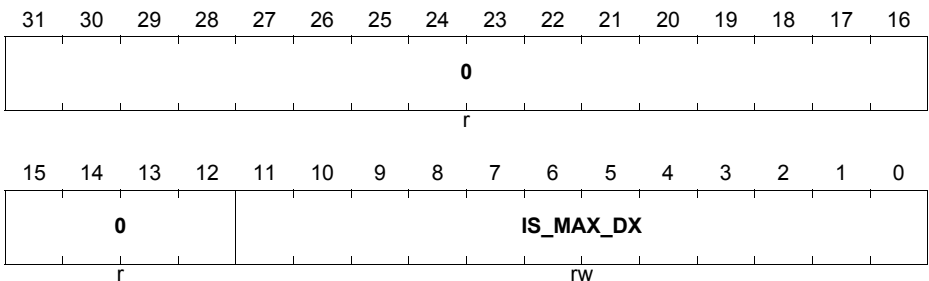
ISP Image Stabilization Maximum Horizontal Displacement Register

CIFISPIS_MAX_DX

ISP Image Stabilization Maximum Horizontal Displacement Register

(0018_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
IS_MAX_DX	11:0	rw	Maximum Horizontal Displacement Maximum allowed accumulated horizontal displacement in pixels Reset: 000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

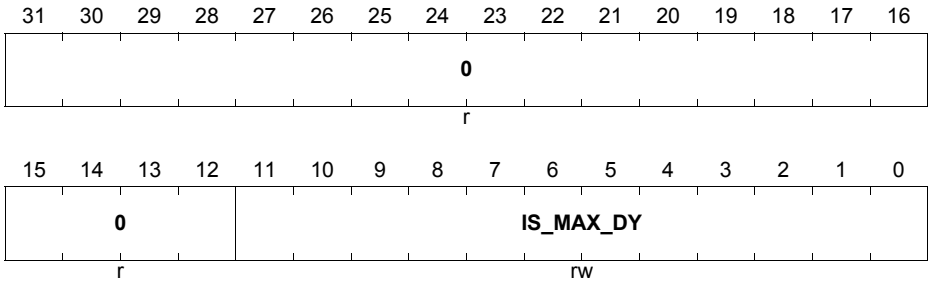
ISP Image Stabilization Maximum Vertical Displacement Register

CIFISPIS_MAX_DY

ISP Image Stabilization Maximum Vertical Displacement Register

(001C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
IS_MAX_DY	11:0	rw	Maximum Vertical Displacement Maximum allowed accumulated vertical displacement in lines Reset: 000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

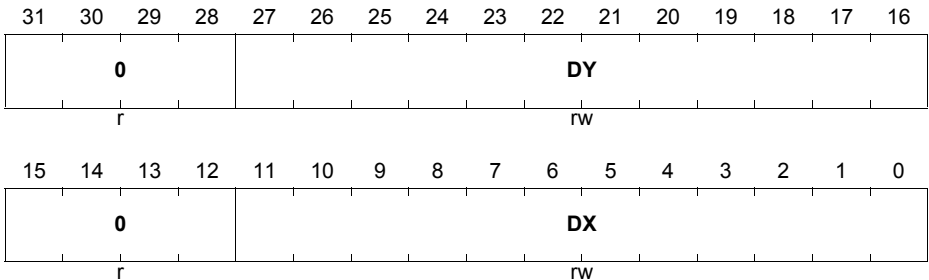
ISP Image Stabilization Camera Displacement Register

CIFISPIS_DISPLACE

ISP Image Stabilization Camera Displacement Register

(0020_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
DY	27:16	rw	Camera Displacement ISP_IS will compensate for vertical camera displacement of DY pixels in the next frame Reset: 000 _H
DX	11:0	rw	Camera Displacement ISP_IS will compensate for horizontal camera displacement of DX pixels in the next frame Reset: 000 _H
0	31:28, 15:12	r	Reserved Read as 0, should be written with 0.

Image Stabilization Shadow Registers

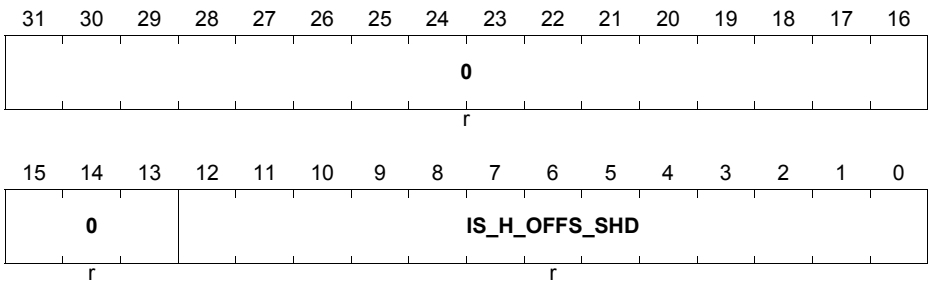
ISP Image Current Horizontal Offset Of Output Window (Shadow Register) Register

CIFISPIS_H_OFFS_SHD

SP Image Current Horizontal Offset Of Output Window Shadow Register

(0024_H)

Reset Value: 0000 0000_H

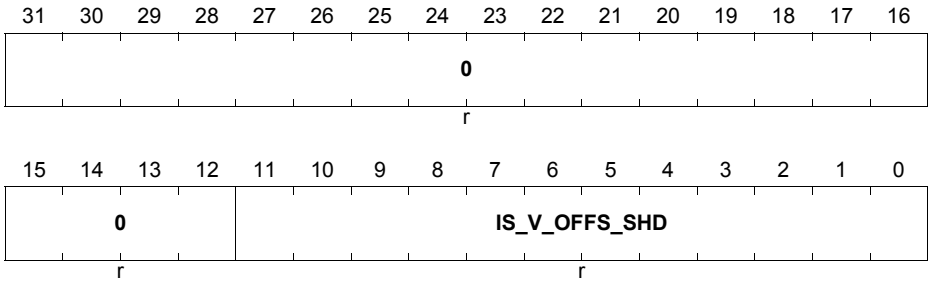


Field	Bits	Type	Description
IS_H_OFFS_S HD	12:0	r	Horizontal Picture Offset Current horizontal picture offset in lines Reset: 0000 _H
0	31:13	r	Reserved Read as 0

ISP Image Current Vertical Offset Of Output Window (Shadow Register) Register

CIFISPIS_V_OFFS_SHD

ISP Image Current Vertical Offset Of Output Window Shadow Register
(0028_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
IS_V_OFFS_SHD	11:0	r	Vertical Picture Offset Current vertical picture offset in lines Reset: 000 _H
0	31:12	r	Reserved Read as 0.

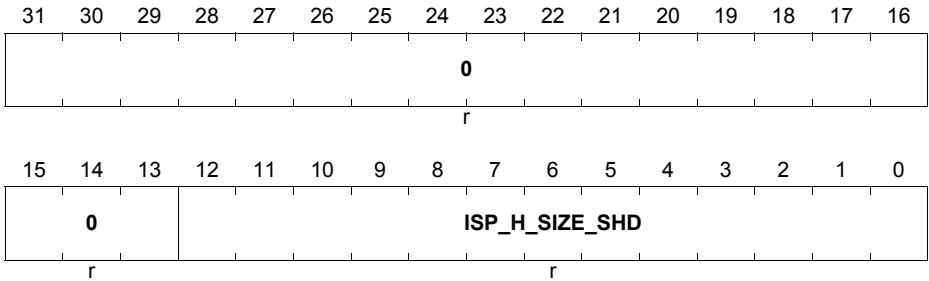
ISP Image Current Output Horizontal Picture Size (Shadow Register) Register

CIFISPIS_H_SIZE_SHD

ISP Image Current Output Horizontal Picture Size Shadow Register

(002C_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
ISP_H_SIZE_S HD	12:0	r	Horizontal Picture Size Current horizontal picture size in pixel Reset: 0000 _H
0	31:13	r	Reserved Read as 0.

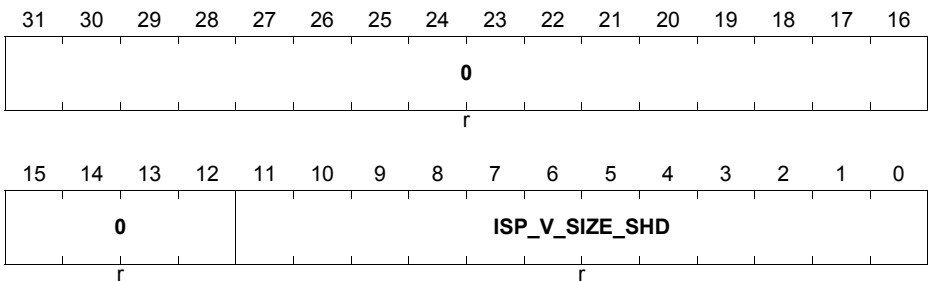
ISP Image Current Output Vertical Picture Size (Shadow Register) Register

CIFISPIS_V_SIZE_SHD

ISP Image Current Output Vertical Picture Size Shadow Register

(0030_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
ISP_V_SIZE_SH D	11:0	r	Vertical Picture Size Current vertical picture size in lines Reset: 000 _H
0	31:12	r	Reserved Read as 0.

36.1.3.9 Extra Path Programming Registers

The address of each CIF Extra Path register is evaluated as CIF_EXTRA_PATH_BASE + Offset.

Extra Path Error Registers

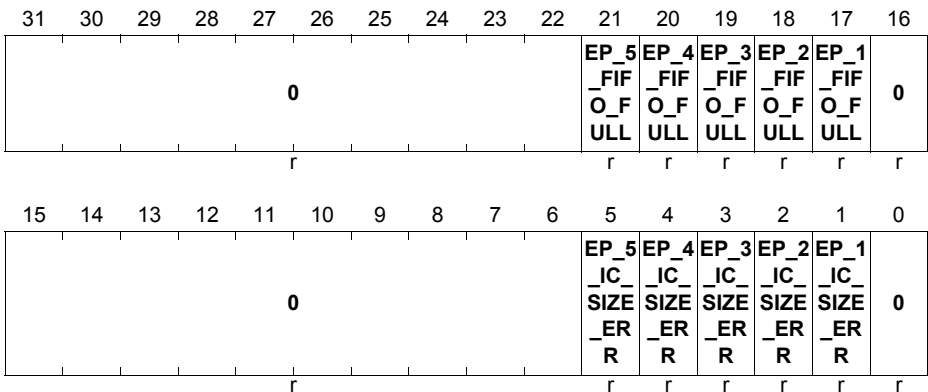
Extra Path Status Error Register

CIFMIEP_STA_ERR

Extra Path Error Register

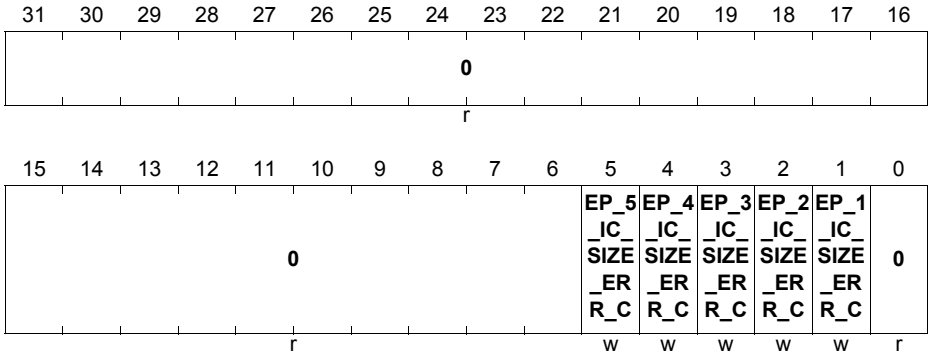
(0000_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
EP_5_FIFO_FULL	21	r	Extra Path 5 FIFO Full FIFO full flag of FIFO in extra path asserted since last clear Reset: 0 _B
EP_4_FIFO_FULL	20	r	Extra Path 4 FIFO Full FIFO full flag of FIFO in extra path asserted since last clear Reset: 0 _B
EP_3_FIFO_FULL	19	r	Extra Path 3 FIFO Full FIFO full flag of FIFO in extra path asserted since last clear Reset: 0 _B

Field	Bits	Type	Description
EP_2_FIFO_FULL	18	r	Extra Path 2 FIFO Full FIFO full flag of FIFO in extra path asserted since last clear Reset: 0 _B
EP_1_FIFO_FULL	17	r	Extra Path 1 FIFO Full FIFO full flag of FIFO in extra path asserted since last clear Reset: 0 _B
EP_5_IC_SIZE_ERR	5	r	Size error is generated in Extra Path 5 image cropping submodule Reset: 0 _B
EP_4_IC_SIZE_ERR	4	r	Size error is generated in Extra Path 4 image cropping submodule Reset: 0 _B
EP_3_IC_SIZE_ERR	3	r	Size error is generated in Extra Path 3 image cropping submodule Reset: 0 _B
EP_2_IC_SIZE_ERR	2	r	Size error is generated in Extra Path 2 image cropping submodule Reset: 0 _B
EP_1_IC_SIZE_ERR	1	r	Size error is generated in Extra Path 1 image cropping submodule Reset: 0 _B
0	31:22, 16:6, 0	r	Reserved Read as 0.

Extra Path Status Error Clear Register
CIFMIEP_STA_ERR_CLR
Extra Path Error Clear Register
(0004_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
EP_5_IC_SIZE_ERR_CLR	5	w	Size error is cleared Reset: 0 _B
EP_4_IC_SIZE_ERR_CLR	4	w	Size error is cleared Reset: 0 _B
EP_3_IC_SIZE_ERR_CLR	3	w	Size error is cleared Reset: 0 _B
EP_2_IC_SIZE_ERR_CLR	2	w	Size error is cleared Reset: 0 _B
EP_1_IC_SIZE_ERR_CLR	1	w	Size error is cleared Reset: 0 _B
0	31:6, 0	r	Reserved Read as 0, should be written with 0.

Memory Interface Extra Path Interrupt Registers

MI Extra Path Interrupt Mask ('1': interrupt active, '0': interrupt masked)

CIFMIEP_IMSC

MI Extra Path Interrupt Mask '1': interrupt active, '0': interrupt masked
(0008_H) Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												MBLK_LINE_EP_5	WRAP_P_5	FILL_EP_5	FRAME_END_EP_5
												rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBLK_LINE_EP_4	WRAP_P_4	FILL_EP_4	FRAME_END_EP_4	MBLK_LINE_EP_3	WRAP_P_3	FILL_EP_3	FRAME_END_EP_3	MBLK_LINE_EP_2	WRAP_P_2	FILL_EP_2	FRAME_END_EP_2	MBLK_LINE_EP_1	WRAP_P_1	FILL_EP_1	FRAME_END_EP_1
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Field	Bits	Type	Description
MBLK_LINE_EP_5	19	rw	Macro Block Line Interrupt Extra Path 5 Mask bit for macroblock line interrupt of Extra Path 5 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_5	18	rw	Wrap Extra Path 5 Mask bit for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_5	17	rw	Fill Extra Path 5 Mask bit for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_5	16	rw	Extra Path 5 Frame End Mask Extra Path end of frame interrupt Reset: 0 _B

Field	Bits	Type	Description
MBLK_LINE_EP_4	15	rw	Macro Block Line Interrupt Extra Path 4 Mask bit for macroblock line interrupt of Extra Path 4 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_4	14	rw	Wrap Extra Path 4 Mask bit for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_4	13	rw	Fill Extra Path 4 Mask bit for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_4	12	rw	Extra Path 4 Frame End Mask Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_3	11	rw	Macro Block Line Interrupt Extra Path 3 Mask bit for macroblock line interrupt of Extra Path 3 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_3	10	rw	Wrap Extra Path 3 Mask bit for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_3	9	rw	Fill Extra Path 3 Mask bit for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_3	8	rw	Extra Path 3 Frame End Mask Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_2	7	rw	Macro Block Line Interrupt Extra Path 2 Mask bit for macroblock line interrupt of Extra Path 2 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_2	6	rw	Wrap Extra Path 2 Mask bit for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_2	5	rw	Fill Extra Path 2 Mask bit for fill level interrupt of Extra Path Reset: 0 _B

Field	Bits	Type	Description
FRAME_END_EP_2	4	rw	Extra Path 2 Frame End Mask Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_1	3	rw	Macro Block Line Interrupt Extra Path 1 Mask bit for macroblock line interrupt of Extra Path 1 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_1	2	rw	Wrap Extra Path 1 Mask bit for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_1	1	rw	Fill Extra Path 1 Mask bit for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_1	0	rw	Extra Path 1 Frame End Mask Extra Path end of frame interrupt Reset: 0 _B
0	31:20	r	Reserved Read as 0, should be written with 0.

MI Extra Path Raw Interrupt Status Register

CIFMIEP_RIS

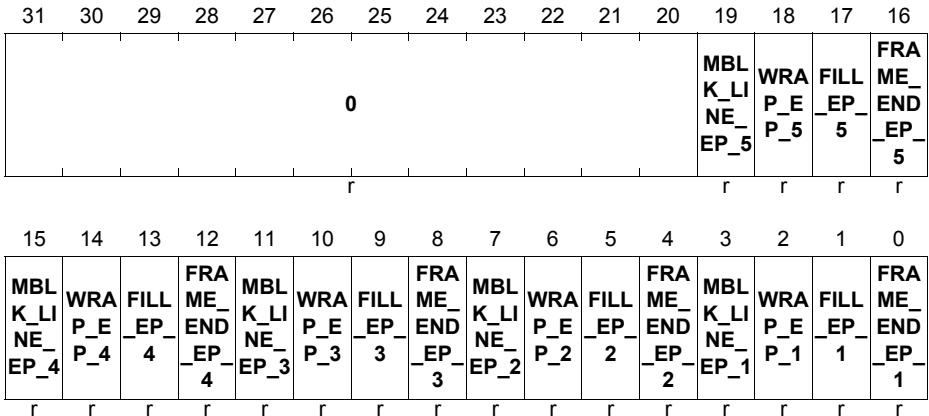
MI Extra Path Raw Interrupt Status Register

(000C_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												MBLK_LI NE_EP_5	WRA P_5	FILL _EP_5	FRA ME _EP_5
r												r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBLK_LI NE_EP_4	WRA P_4	FILL _EP_4	FRA ME _EP_4	MBLK_LI NE_EP_3	WRA P_3	FILL _EP_3	FRA ME _EP_3	MBLK_LI NE_EP_2	WRA P_2	FILL _EP_2	FRA ME _EP_2	MBLK_LI NE_EP_1	WRA P_1	FILL _EP_1	FRA ME _EP_1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Field	Bits	Type	Description
MBLK_LINE_EP_5	19	r	Macro Block Line Interrupt Extra Path 5 Raw status for macroblock line interrupt of Extra Path 5 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_5	18	r	Wrap Extra Path 5 Raw status for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_5	17	r	Fill Extra Path 5 Raw status for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_5	16	r	Extra Path 5 Frame End Raw status picture end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_4	15	r	Macro Block Line Interrupt Extra Path 4 Raw status for macroblock line interrupt of Extra Path 4 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_4	14	r	Wrap Extra Path 4 Raw status for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_4	13	r	Fill Extra Path 4 Raw status for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_4	12	r	Extra Path 4 Frame End Raw status Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_3	11	r	Macro Block Line Interrupt Extra Path 3 Raw status for macroblock line interrupt of Extra Path 3 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_3	10	r	Wrap Extra Path 3 Raw status for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_3	9	r	Fill Extra Path 3 Raw status for fill level interrupt of Extra Path Reset: 0 _B

Field	Bits	Type	Description
FRAME_END_E P_3	8	r	Extra Path 3 Frame End Raw status Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP _2	7	r	Macro Block Line Interrupt Extra Path 2 Raw status for macroblock line interrupt of Extra Path 2 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_2	6	r	Wrap Extra Path 2 Raw status for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_2	5	r	Fill Extra Path 2 Raw status for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_E P_2	4	r	Extra Path 2 Frame End Raw status Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP _1	3	r	Macro Block Line Interrupt Extra Path 1 Raw status for macroblock line interrupt of Extra Path 1 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_1	2	r	Wrap Extra Path 1 Raw status for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_1	1	r	Fill Extra Path 1 Raw status for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_E P_1	0	r	Extra Path 1 Frame End Raw status Extra Path end of frame interrupt Reset: 0 _B
0	31:20	r	Reserved Read as 0.

MI Extra Path Masked Interrupt Status Register
CIFMIEP_MIS
**MI Extra Path Masked Interrupt Status Register
(0010_H)**
Reset Value: 0000 0000_H


Field	Bits	Type	Description
MBLK_LINE_EP_5	19	r	Macro Block Line Interrupt Extra Path 5 Masked status for macroblock line interrupt of Extra Path 5 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_5	18	r	Wrap Extra Path 5 Masked status for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_5	17	r	Fill Extra Path 5 Masked status for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_5	16	r	Extra Path 5 Frame End Masked status picture end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_4	15	r	Macro Block Line Interrupt Extra Path 4 Masked status for macroblock line interrupt of Extra Path 4 (16 lines are written into RAM) Reset: 0 _B

Field	Bits	Type	Description
WRAP_EP_4	14	r	Wrap Extra Path 4 Masked status for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_4	13	r	Fill Extra Path 4 Masked status for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_4	12	r	Extra Path 4 Frame End Masked status Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_3	11	r	Macro Block Line Interrupt Extra Path 3 Masked status for macroblock line interrupt of Extra Path 3 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_3	10	r	Wrap Extra Path 3 Masked status for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_3	9	r	Fill Extra Path 3 Masked status for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_3	8	r	Extra Path 3 Frame End Masked status Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_2	7	r	Macro Block Line Interrupt Extra Path 2 Masked status for macroblock line interrupt of Extra Path 2 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_2	6	r	Wrap Extra Path 2 Masked status for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_2	5	r	Fill Extra Path 2 Masked status for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_2	4	r	Extra Path 2 Frame End Masked status Extra Path end of frame interrupt Reset: 0 _B

Field	Bits	Type	Description
MBLK_LINE_EP_1	3	r	Macro Block Line Interrupt Extra Path 1 Masked status for macroblock line interrupt of Extra Path 1 (16 lines are written into RAM) Reset: 0 _B
WRAP_EP_1	2	r	Wrap Extra Path 1 Masked status for Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_1	1	r	Fill Extra Path 1 Masked status for fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_1	0	r	Extra Path 1 Frame End Masked status Extra Path end of frame interrupt Reset: 0 _B
0	31:20	r	Reserved Read as 0.

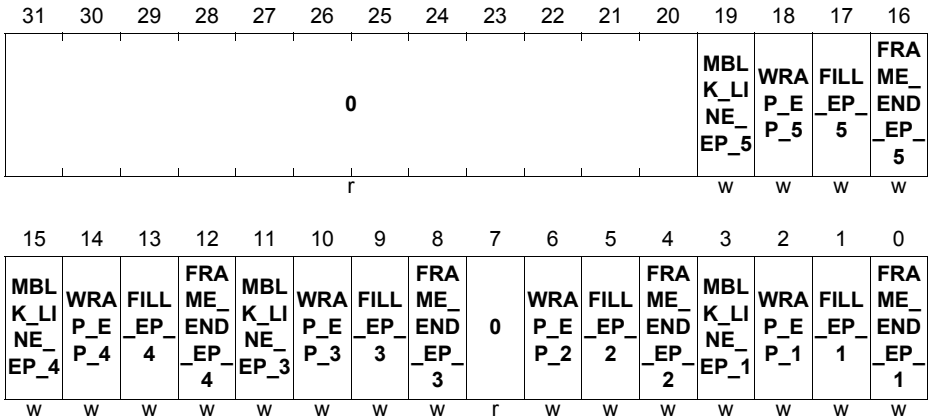
MI Extra Path Interrupt Clear Register

CIFMIEP_ICR

MI Extra Path Interrupt Clear Register

(0014_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
MBLK_LINE_EP_5	19	w	Macro Block Line Interrupt Extra Path 5 Clear macroblock line interrupt of Extra Path 5 Reset: 0 _B
WRAP_EP_5	18	w	Wrap Extra Path 5 Clear Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_5	17	w	Fill Extra Path 5 Clear fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_5	16	w	Extra Path 5 Frame End Clear picture end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_4	15	w	Macro Block Line Interrupt Extra Path 4 Clear macroblock line interrupt of Extra Path 4 Reset: 0 _B
WRAP_EP_4	14	w	Wrap Extra Path 4 Clear Extra Path address wrap interrupt Reset: 0 _B

Field	Bits	Type	Description
FILL_EP_4	13	w	Fill Extra Path 4 Clear fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_4	12	w	Extra Path 4 Frame End Clear Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_3	11	w	Macro Block Line Interrupt Extra Path 3 Clear macroblock line interrupt of Extra Path 3 Reset: 0 _B
WRAP_EP_3	10	w	Wrap Extra Path 3 Clear Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_3	9	w	Fill Extra Path 3 Clear fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_3	8	w	Extra Path 3 Frame End Clear Extra Path end of frame interrupt Reset: 0 _B
WRAP_EP_2	6	w	Wrap Extra Path 2 Clear Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_2	5	w	Fill Extra Path 2 Clear fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_2	4	w	Extra Path 2 Frame End Clear Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_1	3	w	Macro Block Line Interrupt Extra Path 1 Clear macroblock line interrupt of Extra Path 1 Reset: 0 _B
WRAP_EP_1	2	w	Wrap Extra Path 1 Clear Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_1	1	w	Fill Extra Path 1 Clear fill level interrupt of Extra Path Reset: 0 _B

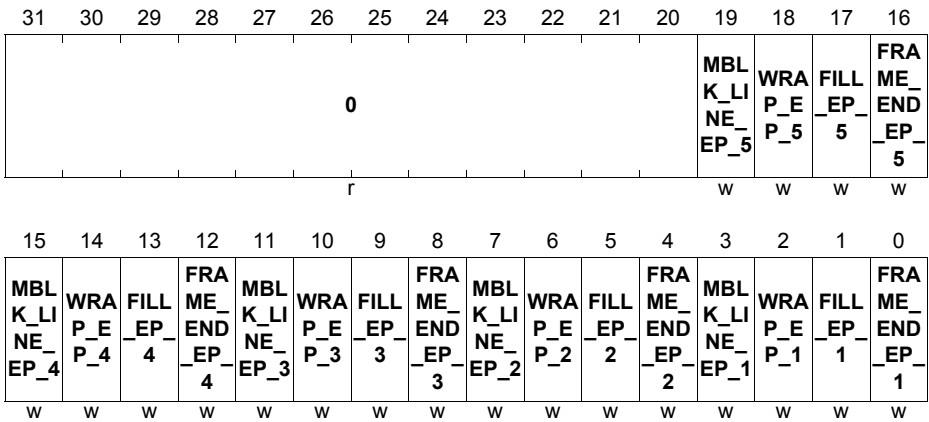
Field	Bits	Type	Description
FRAME_END_EP_1	0	w	Extra Path 1 Frame End Clear Extra Path end of frame interrupt Reset: 0 _B
0	31:20, 7	r	Reserved Read as 0, should be written with 0.

MI Extra Path Interrupt Set Register

CIFMIEP_ISR

MI Extra Path Interrupt Set Register (0018_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
MBLK_LINE_EP_5	19	w	Macro Block Line Interrupt Extra Path 5 Set macroblock line interrupt of Extra Path 5 Reset: 0 _B
WRAP_EP_5	18	w	Wrap Extra Path 5 Set Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_5	17	w	Fill Extra Path 5 Set fill level interrupt of Extra Path Reset: 0 _B

Field	Bits	Type	Description
FRAME_END_E P_5	16	w	Extra Path 5 Frame End Set picture end of frame interrupt Reset: 0 _B
MBLK_LINE_EP _4	15	w	Macro Block Line Interrupt Extra Path 4 Set macroblock line interrupt of Extra Path 4 Reset: 0 _B
WRAP_EP_4	14	w	Wrap Extra Path 4 Set Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_4	13	w	Fill Extra Path 4 Set fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_E P_4	12	w	Extra Path 4 Frame End Set Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP _3	11	w	Macro Block Line Interrupt Extra Path 3 Set macroblock line interrupt of Extra Path 3 Reset: 0 _B
WRAP_EP_3	10	w	Wrap Extra Path 3 Set Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_3	9	w	Fill Extra Path 3 Set fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_E P_3	8	w	Extra Path 3 Frame End Set Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP _2	7	w	Macro Block Line Interrupt Extra Path 2 Set macroblock line interrupt of Extra Path 2 Reset: 0 _B
WRAP_EP_2	6	w	Wrap Extra Path 2 Set Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_2	5	w	Fill Extra Path 2 Set fill level interrupt of Extra Path Reset: 0 _B

Field	Bits	Type	Description
FRAME_END_EP_2	4	w	Extra Path 2 Frame End Set Extra Path end of frame interrupt Reset: 0 _B
MBLK_LINE_EP_1	3	w	Macro Block Line Interrupt Extra Path 1 Set macroblock line interrupt of Extra Path 1 Reset: 0 _B
WRAP_EP_1	2	w	Wrap Extra Path 1 Set Extra Path address wrap interrupt Reset: 0 _B
FILL_EP_1	1	w	Fill Extra Path 1 Set fill level interrupt of Extra Path Reset: 0 _B
FRAME_END_EP_1	0	w	Extra Path 1 Frame End Set Extra Path end of frame interrupt Reset: 0 _B
0	31:20	r	Reserved Read as 0, should be written with 0.

Memory Interface Extra Path Control Registers

Memory Interface Extra Path Control Register

CIFMIEP_z_CTRL (z=1-5)

Memory Interface Extra Path z Control Register

 (z*0100_H)

 Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								EP_WRITE_FORMAT	INIT_OFF_SET_EN	INIT_BAS_EEN	0				
r								rw	rw	rw	r				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							BYT_ES_WAP	0						EP_ENABLE	
r							rw	r						rw	

Field	Bits	Type	Description
EP_WRITE_FOR MAT	23:22	rw	<p>Extra Path Write Format Defines how Extra Path data is written to memory.</p> <p><i>Note: Programmed value becomes effective immediately. So write to the register only if no picture data is sent to the path.</i></p> <p>00_B RAW & data mode (8 bit) 01_B RAW 8 bit 10_B RAW & data mode (greater 8 up to 16 bit) 11_B YCbCr 16 bit; YCbCr data is handled interleaved as 16 bit data in extra paths. Reset: 00_B</p>
INIT_OFFSET_E N	21	rw	<p>Init Offset Counter Enable Enables updating of the offset counters shadow registers for Extra Path to the programmed register init values. MI_EP_OFFS_CNT_INIT -> MI_EP_OFFS_CNT_SHD The update will be executed either when a forced software update occurs (in register MI_EP_n_INIT bit cfg_upd = 1) or when an automatic config update signal arrives at the MI input port. So only the corresponding shadow registers are affected. After a picture skip has been performed init_offset_en selects between skip restart and skip init mode (see bit skip in register MI_EP_INIT). Reset: 0_B</p>

Field	Bits	Type	Description
INIT_BASE_EN	20	rw	<p>Init Base Address Enable Enables updating of the base address and buffer size shadow registers for picture to the programmed register init values. MI_EP_BASE_AD_INIT -> MI_EP_BASE_AD_SHD MI_EP_SIZE_INIT -> MI_EP_SIZE_SHD The update will be executed either when a forced software update occurs (in register MI_EP_n_INIT bit <code>cfg_upd = 1</code>) or when an automatic config update signal arrives at the MI input port. So only the corresponding shadow registers are affected. Reset: 0_B</p>
BYTE_SWAP	7	rw	<p>Byte Swap Enable Enables change of byte order of the 32 bit output word at write port. <i>Note: Programmed value becomes effective immediately. So write to the register only if no picture data is sent.</i> 0_B no byte mirroring 1_B byte order is mirrored but the bit order within one byte doesn't change Reset: 0_B</p>
EP_ENABLE	0	rw	<p>Enables enable ep picture data path Programmed value becomes effective (visible in control shadow register) after a soft reset, a forced software update or an automatic config update. Affects MI_IN and MI_OUT module. Reset: 0_B</p>
0	31:24, 19:8, 6:1	r	<p>Reserved Read as 0, should be written with 0.</p>

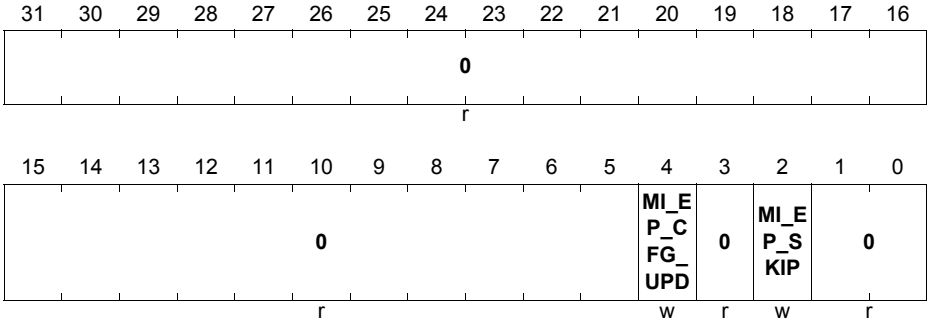
Memory Interface Control Register For Address Init And Skip Function Register

CIFMIEP_z_INIT (z=1-5)

Memory Interface Extra Path z Control Register For Address Init And Skip Function Register

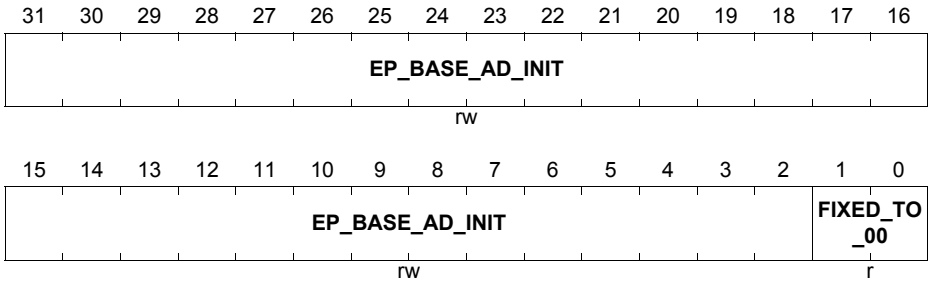
(z*0100_H+4_H)

Reset Value: 0000 0000_H

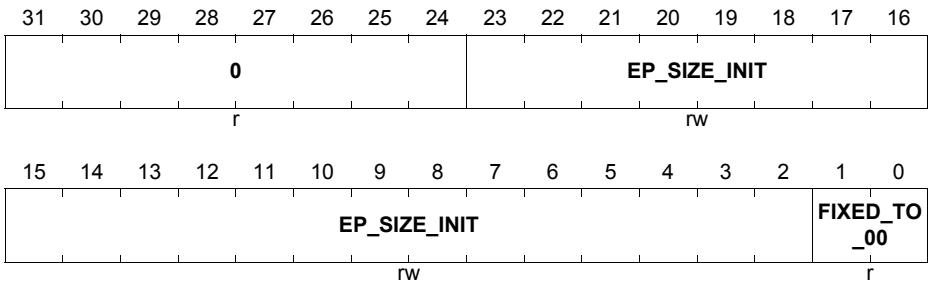


Field	Bits	Type	Description
MI_EP_CFG_UPD	4	w	Forced Configuration Update Leads to an immediate update of the shadow registers. Depending on the two init enable bits in the MI_EP_n_CTRL register (init_offset_en and init_base_en) the offset counter, base address and buffer size shadow registers are also updated. Reset: 0 _B

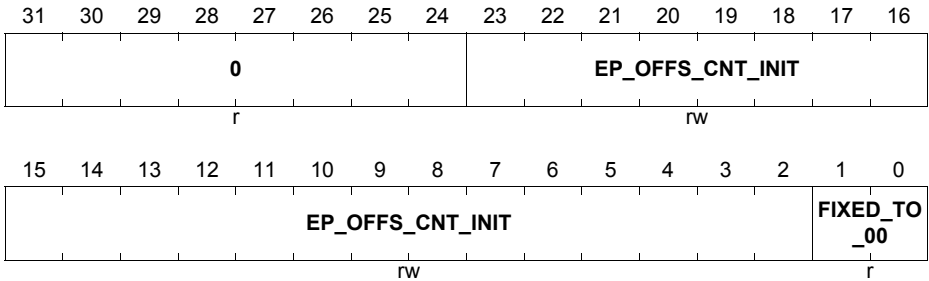
Field	Bits	Type	Description
MI_EP_SKIP	2	w	<p>Skip Picture</p> <p>Skip of current or next starting main picture: Aborts writing of main picture image data of the current frame to RAM (after the current burst transmission has been completed). Further picture data up to the end of the current frame are discarded. No further macroblock line interrupt (mblk_line), no wrap around interrupt for picture (wrap_ep_n) and no fill level interrupt (fill_ep_n) are generated. Skip does not affect the generation of the extra path frame end interrupt (ep_frame_end). The byte counter (register MI_EP_BYTE_CNT) is not affected. It produces the correct number of RAW data bytes at the end of the current (skipped) frame. After a skip has been performed the offset counter for the main picture at the start of the following frame are set depending on the bit init_offset_en in register MI_CTRL:</p> <p>A) Skip restart mode (init_offset_en = 0) The offset counters of the Extra Path are restarted at the old start value of the previous skipped frame.</p> <p>B) Skip init mode (init_offset_en = 1) The offset counters of the Extra Path are initialized with the register contents of the offset counter init registers without any additional forced software update or automatic config update.</p> <p>Reset: 0_B</p>
0	31:5, 3, 1:0	r	<p>Reserved</p> <p>Read as 0, should be written with 0.</p>

Memory Interface Base Address for Extra Path Buffer Register
CIFMIEP_z_BASE_AD_INIT (z=1-5)
Memory Interface Base Address For Extra Path z Data Buffer Register
 (z*0100_H+8_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
EP_BASE_AD_INIT	31:2	rw	Extra Path Base Address Init Base address of Extra Path ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update. <i>Note: Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to "00" (word aligned value). Reset: 00 _B

Memory Interface Size Of Extra Path Buffer Register
CIFMIEP_k_SIZE_INIT (k=1-5)
Memory Interface Size of Extra Path k Data Buffer Register
 $(k \cdot 0100_H + 0C_H)$
Reset Value: 0000 0000_H


Field	Bits	Type	Description
EP_SIZE_INIT	23:2	rw	Extra Path Size Init Size of Extra Path ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update. <i>Note: Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0, should be written with 0.

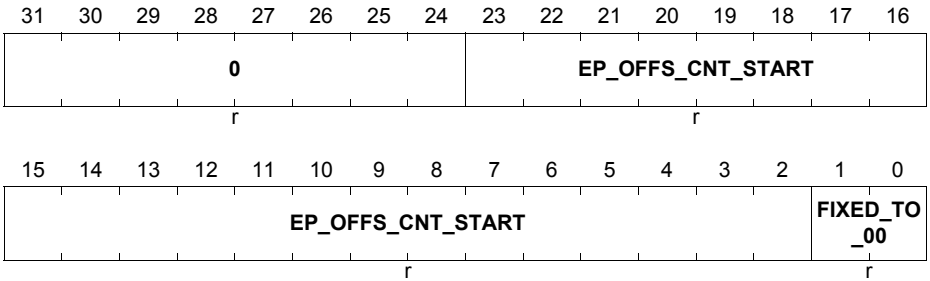
Memory Interface Offset Counter Init Value For Extra Path Buffer Register
CIFMIEP_z_OFFS_CNT_INIT (z=1-5)
Memory Interface Offset Counter Init Value For Extra Path z Buffer Register
(z*0100_H+10_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
EP_OFFS_CNT_INIT	23:2	rw	Extra Path Offset Counter Init Offset counter init value of Extra Path ring buffer. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update. <i>Note: Set control bit <code>init_base_en</code> before updating so that a forced or automatic update can take effect. Check exceptional handling in skip modes.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to "00" (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0, should be written with 0.

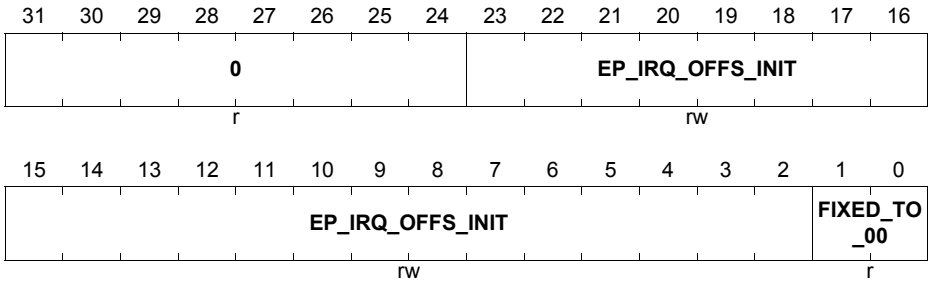
Memory Interface Offset Counter Start Value for Extra Path Register

CIFMIEP_z_OFFS_CNT_START (z=1-5)

Memory Interface Offset Counter Start Value For Extra Path z Register
 ($z \cdot 0100_H + 14_H$) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
EP_OFFS_CNT_START	23:2	r	Extra Path Offset Counter Start Offset counter value which points to the start address of the previously processed picture. Updated at frame end. Note: A soft reset resets the contents to the reset value. Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

Memory Interface Fill Level Interrupt Offset Value For Extra Path Data Register
CIFMIEP_z_IRQ_OFFS_INIT (z=1-5)
Memory Interface Fill Level Interrupt Offset Value For Extra Path z Register
(z*0100_H+18_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
EP_IRQ_OFFS_INIT	23:2	rw	Extra Path Y IRQ Offset Init Reaching this programmed value by the current offset counter for addressing Extra Path buffer leads to generation of fill level interrupt fill_ep_y. Programmed value becomes effective (visible in corresponding shadow register) after a soft reset, a forced software update or an automatic config update. Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to "00" (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0, should be written with 0.

Extra Path Memory Interface Shadow Registers

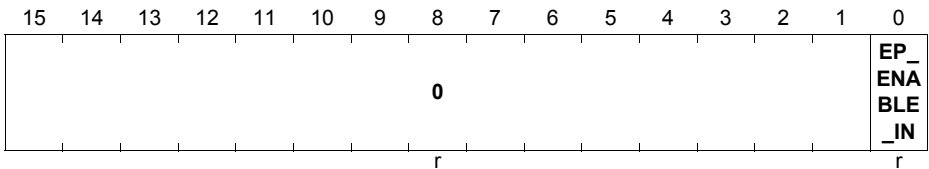
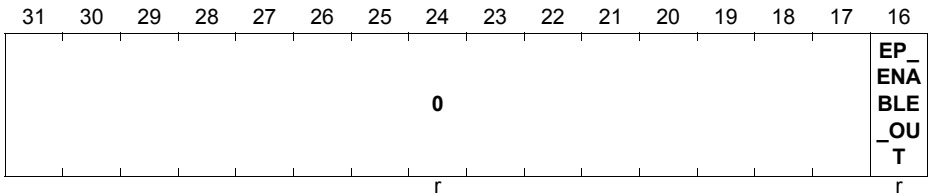
Memory Interface Extra Path Control Internal Shadow Register

CIFMIEP_z_CTRL_SHD (z=1-5)

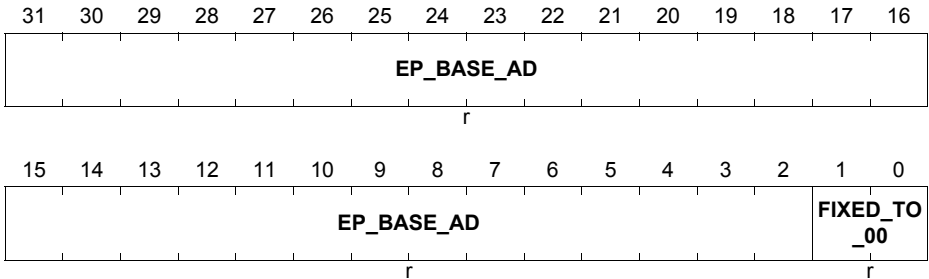
Memory Interface Extra Path z Control Internal Shadow Register

(z*0100_H+1C_H)

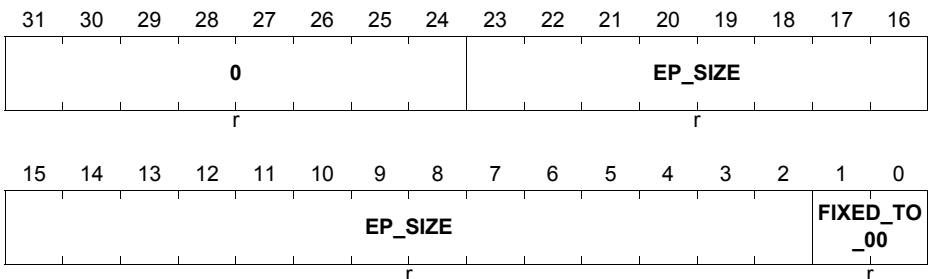
Reset Value: 0000 0000_H



Field	Bits	Type	Description
EP_ENABLE_OUT	16	r	Extra Path Out Enable Extra Path used in module MI_OUT Reset: 0 _B
EP_ENABLE_IN	0	r	Extra Path In Enable Extra Path data used in module MI_IN Reset: 0 _B
0	31:17, 15:1	r	Reserved Read as 0.

Memory Interface Base Address Shadow Register for Extra Path Buffer Register
CIFMIEP_z_BASE_AD_SHD (z=1-5)
Memory Interface Base Address Shadow Register for Extra Path z Buffer Register
 (z*0100_H+20_H) **Reset Value: 0000 0000_H**


Field	Bits	Type	Description
EP_BASE_AD	31:2	r	Extra Path Base Address Base address of Extra Path ring buffer. Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B

Memory Interface Size Shadow Register of Extra Path Buffer Register
CIFMIEP_k_SIZE_SHD (k=1-5)
Memory Interface Size Shadow Register of Extra Path k Buffer Register
 (k*0100_H+24_H) **Reset Value: 0000 0000_H**


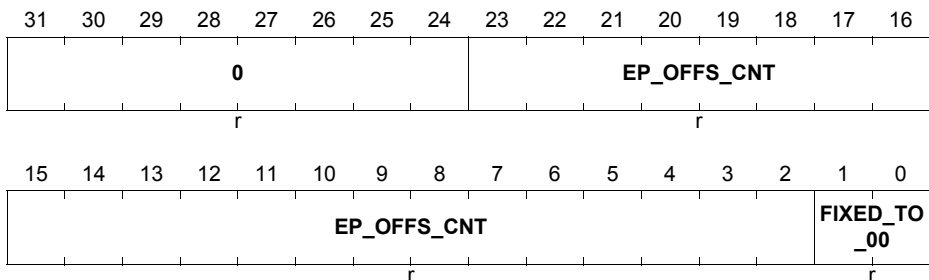
Field	Bits	Type	Description
EP_SIZE	23:2	r	Extra Path Size Size of Extra Path ring buffer. Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

Memory Interface Current Offset Counter of Extra Path Buffer Register

CIFMIEP_z_OFFS_CNT_SHD (z=1-5)

Memory Interface Current Offset Counter of Extra Path z Buffer Register

 $(z * 0100_H + 28_H)$

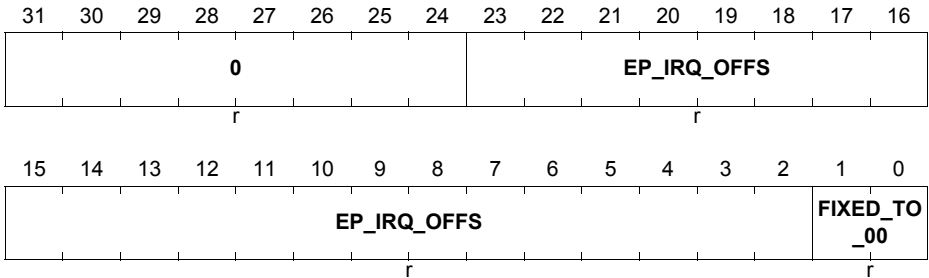
 Reset Value: 0000 0000_H


Field	Bits	Type	Description
EP_OFFS_CNT	23:2	r	Extra Path Y Offset Counter Current offset counter of Extra Path ringbuffer for address generation. <i>Note: Soft reset will reset the contents to reset value.</i> Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to “00” (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

Memory Interface Shadow Register of Fill Level Interrupt Offset Value For Extra Path Register

CIFMIEP_z_IRQ_OFFS_SHD (z=1-5)

Memory Interface Shadow Register of Fill Level Interrupt Offset Value For Extra Path z Register
 (z*0100_H+2C_H) Reset Value: 0000 0000_H



Field	Bits	Type	Description
EP_IRQ_OFFS	23:2	r	Extra Path IRQ Offset Reaching this offset value by the current offset counter for addressing Extra Path leads to generation of fill level interrupt fill_ep_y. Reset: 0 _D
FIXED_TO_00	1:0	r	Bits [1:0] are set to "00" (word aligned value). Reset: 00 _B
0	31:24	r	Reserved Read as 0.

Extra Path Image Cropping Control Registers

The address of each CIF Extra Path Image Cropping register is evaluated as CIF_EP_IC_BASE + Offset

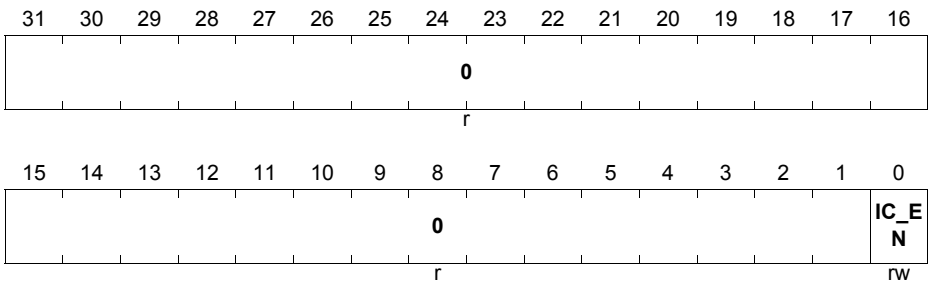
Extra Path Image Cropping Control Register

CIFEP_z_IC_CTRL (z=1-5)

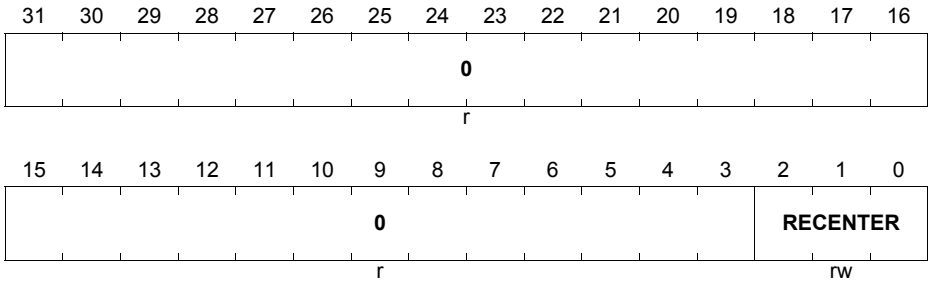
Extra Path z Image Cropping Control Register

(z*100_H-100_H)

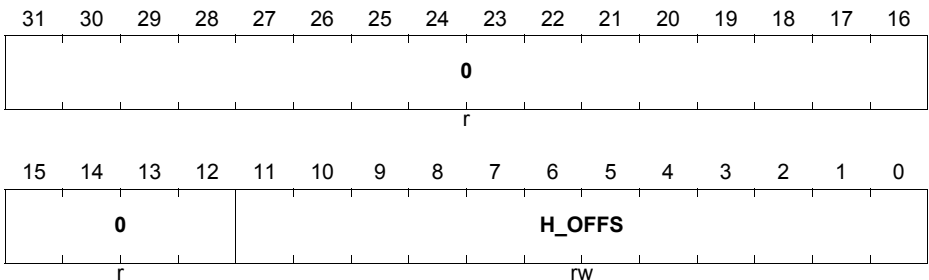
Reset Value: 0000 0000_H



Field	Bits	Type	Description
IC_EN	0	rw	Image Cropping Enable 0 _B image cropping switched off 1 _B image cropping switched on Reset: 0 _B
0	31:1	r	Reserved Read as 0.

Extra Path Image Cropping Recenter Register
CIFEP_z_IC_RECENTER (z=1-5)
Extra Path z Image Cropping Recenter Register
(z*100_H-0FC_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RECENTER	2:0	rw	Recenter 000 _B recenter feature switched off For all other values recentering is active (cur_h/v_offs-H/V_OFFS)/2 power(recenter) Reset: 000 _B
0	31:3	r	Reserved Read as 0, should be written with 0.

Extra Path Image Cropping Horizontal Offset of Output Window Register
CIFEP_z_IC_H_OFFS (z=1-5)
Extra Path z Image Cropping Horizontal Offset of Output Window Register
(z*100_H-0F8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
H_OFFS	11:0	rw	Horizontal Picture Offset Horizontal picture offset in pixel Reset: 0000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

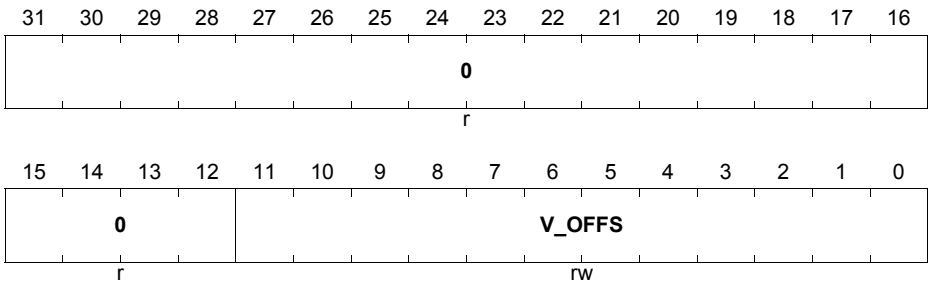
Extra Path Image Cropping Vertical Offset Of Output Window Register

CIFEP_z_IC_V_OFFS (z=1-5)

Extra Path z Image Cropping Vertical Offset Of Output Window Register

(z*100_H-0F4_H)

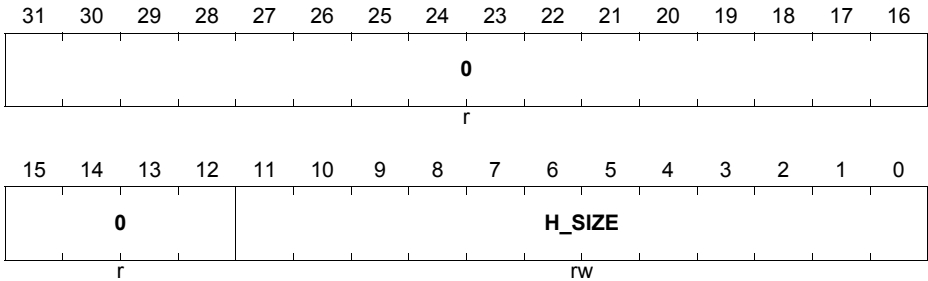
Reset Value: 0000 0000_H



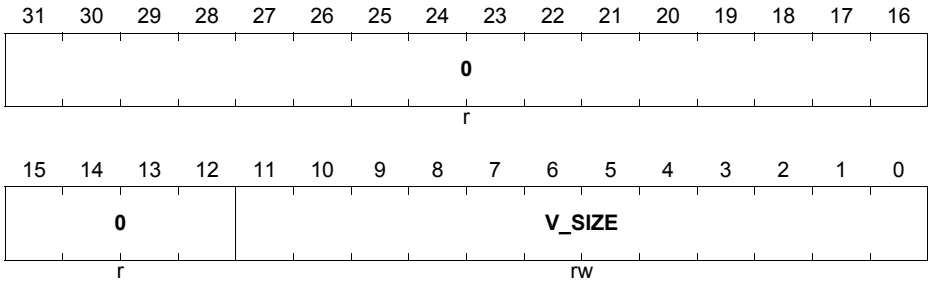
Field	Bits	Type	Description
V_OFFS	11:0	rw	Vertical Picture Offset Vertical picture offset in lines Reset: 000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

Extra Path Image Cropping Output Horizontal Picture Size Register
CIFEP_z_IC_H_SIZE (z=1-5)
Extra Path z Image Cropping Output Horizontal Picture Size Register

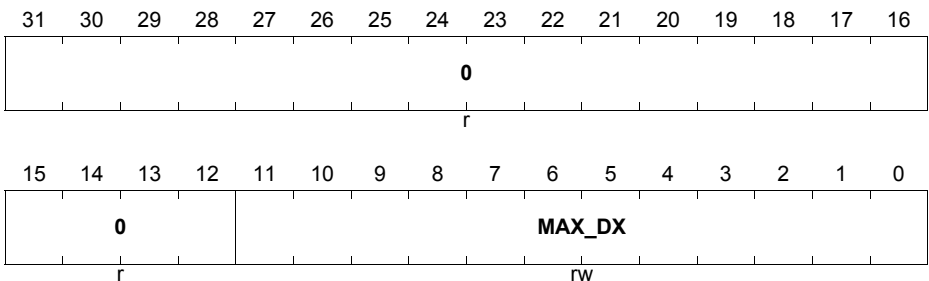
 ($z \cdot 100_{\text{H}} - 0\text{F0}_{\text{H}}$)

 Reset Value: **0000 0A28_H**


Field	Bits	Type	Description
H_SIZE	11:0	rw	Horizontal Picture Size Horizontal picture size in pixel Only even numbers are accepted, because complete quadruples of YUYV(YCbYCr) are needed for the following modules. If an odd size is programmed the value will be truncated to even size. If ISP_MODE is set to 001 _B (ITU-R BT.656 YUV) 010 _B (ITU-R BT.601 YUV) 011 _B (ITU-R BT.601 Bayer RGB) 101 _B (ITU-R BT.656 Bayer RGB) Reset: A28 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

Extra Path Image Cropping Output Vertical Picture Size Register
CIFEP_k_IC_V_SIZE (k=1-5)
Extra Path k Image Cropping Output Vertical Picture Size Register
(k*100_H-0EC_H)
Reset Value: 0000 0800_H


Field	Bits	Type	Description
V_SIZE	11:0	rw	Vertical Picture Size Vertical picture size in lines Reset: 800 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

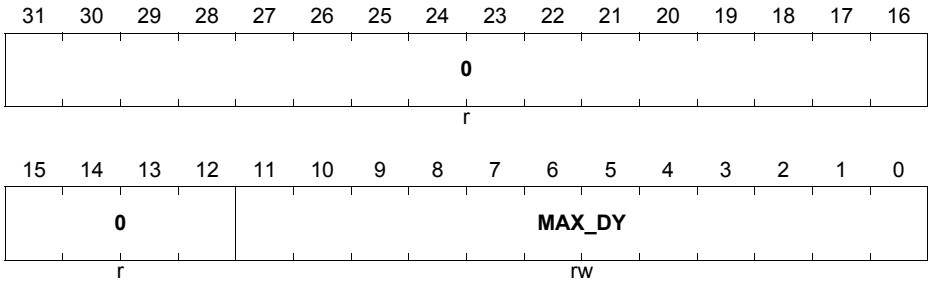
Extra Path Image Cropping Maximum Horizontal Displacement Register
CIFEP_k_IC_MAX_DX (k=1-5)
Extra Path k Image Cropping Maximum Horizontal Displacement Register
(k*100_H-0E8_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
MAX_DX	11:0	rw	Maximum Horizontal Displacement Maximum allowed accumulated horizontal displacement in pixels Reset: 000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

Extra Path Image Cropping Maximum Vertical Displacement Register

CIFEP_z_IC_MAX_DY (z=1-5)

Extra Path z Image Cropping Maximum Vertical Displacement Register
 (z*100_H-0E4_H) **Reset Value: 0000 0000_H**

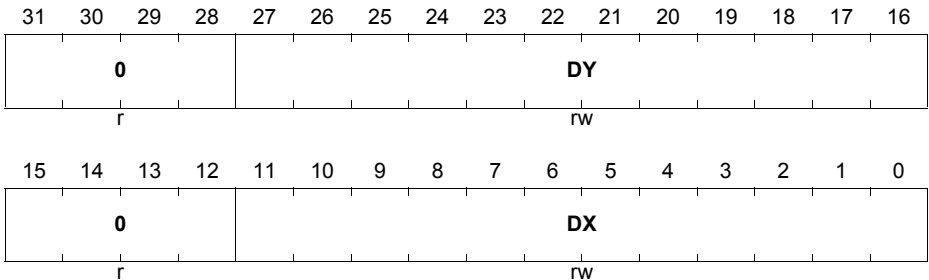


Field	Bits	Type	Description
MAX_DY	11:0	rw	Maximum Vertical Displacement Maximum allowed accumulated vertical displacement in lines Reset: 000 _H
0	31:12	r	Reserved Read as 0, should be written with 0.

Extra Path Image Cropping Camera Displacement Register

CIFEP_z_IC_DISPLACE (z=1-5)

Extra Path z Image Cropping Camera Displacement Register
 (z*100_H-0E0_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
DY	27:16	rw	Camera Displacement ISP_IS will compensate for vertical camera displacement of DY pixels in the next frame Reset: 000 _H
DX	11:0	rw	Camera Displacement Will compensate for horizontal camera displacement of DX pixels in the next frame Reset: 000 _H
0	31:28, 15:12	r	Reserved Read as 0, should be written with 0.

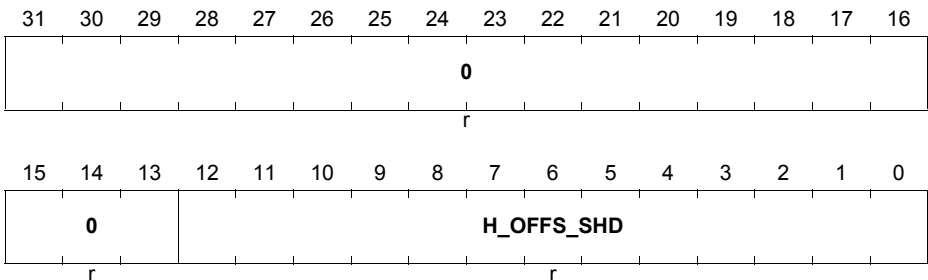
Extra Path Image Cropping Shadow Registers

The address of each CIF Extra Path Image Cropping Shadow register is evaluated as
CIF_EP_IC_BASE + Offset

Extra Path Image Cropping Current Horizontal Offset of Output Window Shadow Register

CIFEP_k_IC_H_OFFS_SHD (k=1-5)

Extra Path k Image Current Horizontal Offset Of Output Window Shadow Register
(k*100_H-0DC_H) Reset Value: 0000 0000_H



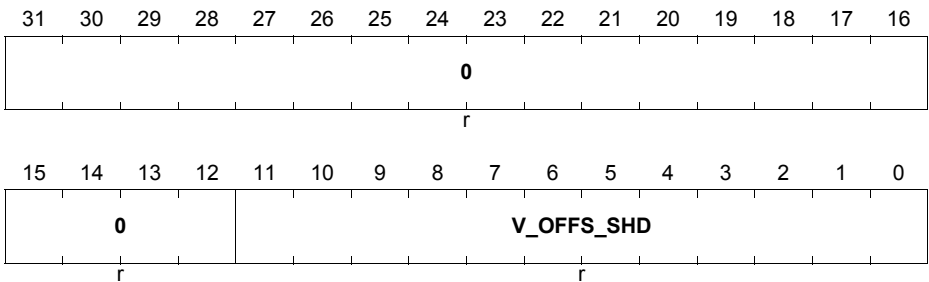
Field	Bits	Type	Description
H_OFFS_S HD	12:0	r	Horizontal Picture Offset Current horizontal picture offset in lines Reset: 0000 _H

Field	Bits	Type	Description
0	31:13	r	Reserved Read as 0.

Extra Path Image Cropping Current Vertical Offset Of Output Window Shadow Register

CIFEP_k_IC_V_OFFS_SHD (k=1-5)

Extra Path k Image Current Vertical Offset Of Output Window Shadow Register
(k*100_H-0D8_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
V_OFFS_SHD	11:0	r	Vertical Picture Offset Current vertical picture offset in lines Reset: 000 _H
0	31:12	r	Reserved Read as 0.

Extra Path Image Cropping Current Output Horizontal Picture Size Shadow

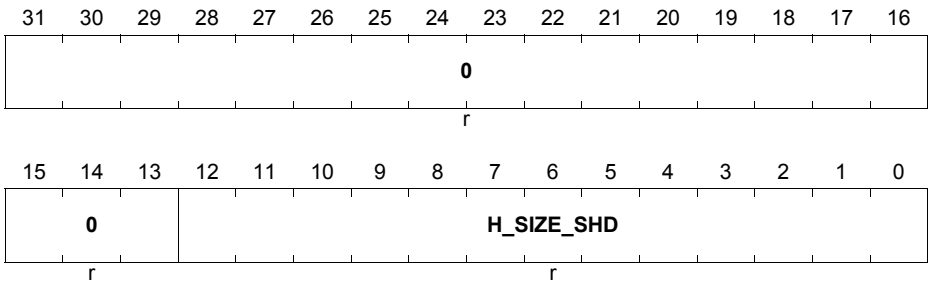
Register

CIFEP_k_IC_H_SIZE_SHD (k=1-5)

Extra Path k Image Current Output Horizontal Picture Size Shadow Register

(k*100_H-0D4_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
H_SIZE_SHD	12:0	r	Horizontal Picture Size Current horizontal picture size in pixel Reset: 0000 _H
0	31:13	r	Reserved Read as 0.

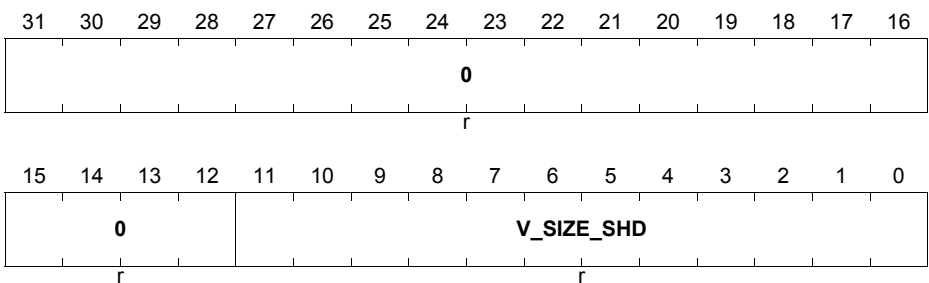
Extra Path Image Cropping Current Output Vertical Picture Size Shadow Register

CIFEP_k_IC_V_SIZE_SHD (k=1-5)

Extra Path k Image Current Output Vertical Picture Size Shadow Register

(k*100_H-0D0_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
V_SIZE_SHD	11:0	r	Vertical Picture Size Current vertical picture size in lines Reset: 000 _H
0	31:12	r	Reserved Read as 0.

36.1.3.10 Debug Path Programming Registers

The address of each CIF Debug Path register is evaluated as CIF_DEBUG_PATH_BASE + Offset.

Debug Path Control Registers

Debug Path Control Register

CIFDP_CTRL

Debug Path Control Register

(0000_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								UDS	UDS	UDS	UDS	UDS	UDS	UDS	UDS
								8	7	6	5	4	3	2	1
r								rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSC	LNC	FNC	0	RST	RST	RST	RST	0				DP_SEL		DP	
_EN	_EN	_EN		_PD	_TS	_LN	_FN	C				SEL		_EN	
rw	rw	rw	r	w	w	w	w	r				rw		rw	

Field	Bits	Type	Description
UDS8	23	rw	Enable/disable sending of User Defined Symbol 8 0 _B disable 1 _B enable Reset: 0 _B
UDS7	22	rw	Enable/disable sending of User Defined Symbol 7 0 _B disable 1 _B enable Reset: 0 _B
UDS6	21	rw	Enable/disable sending of User Defined Symbol 6 0 _B disable 1 _B enable Reset: 0 _B
UDS5	20	rw	Enable/disable sending of User Defined Symbol 5 0 _B disable 1 _B enable Reset: 0 _B

Field	Bits	Type	Description
UDS4	19	rw	Enable/disable sending of User Defined Symbol 4 0 _B disable 1 _B enable Reset: 0 _B
UDS3	18	rw	Enable/disable sending of User Defined Symbol 3 0 _B disable 1 _B enable Reset: 0 _B
UDS2	17	rw	Enable/disable sending of User Defined Symbol 2 0 _B disable 1 _B enable Reset: 0 _B
UDS1	16	rw	Enable/disable sending of User Defined Symbol 1 0 _B disable 1 _B enable Reset: 0 _B
TSC_EN	15	rw	Enable/disable Timestamp Counter 0 _B disable 1 _B enable Reset: 0 _B
LNC_EN	14	rw	Enable/disable Line Number Counter 0 _B disable 1 _B enable Reset: 0 _B
FNC_EN	13	rw	Enable/disable Frame Number Counter 0 _B disable 1 _B enable Reset: 0 _B
RST_PD	11	w	Reset Predivider Counter 1 _B Writing to this register bit sets the Counter to zero Reset: 0 _B
RST_TSC	10	w	Reset Timestamp Counter 1 _B Writing to this register bit sets the Counter to zero Reset: 0 _B
RST_LNC	9	w	Reset Line Number Counter 1 _B Writing to this register bit sets the Counter to zero Reset: 0 _B

Field	Bits	Type	Description
RST_FNC	8	w	Reset Frame Number Counter 1 _B Writing to this register bit sets the Counter to zero Reset: 0 _B
DP_SEL	3:1	rw	Select Source Path which will be transferred over the Debug Interface 000 _B Main Path 001 _B Extra Path 1 010 _B Extra Path 2 011 _B Extra Path 3 100 _B Extra Path 4 101 _B Extra Path 5 Reset: 0 _B
DP_EN	0	rw	Debug Path Enable 0 _B disabled 1 _B enabled Reset: 0 _B
0	31:24, 12, 7:4	r	Reserved Read as 0, should be written with 0.

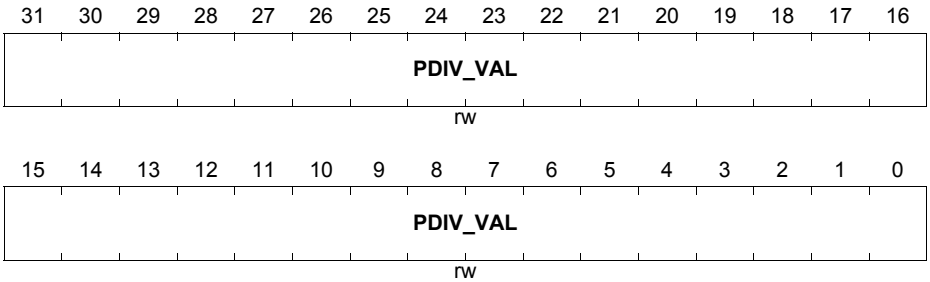
Debug Path Predivider Control Register

CIFDP_PDIV_CTRL

Debug Path Predivider Control Register

(0004_H)

Reset Value: 0000 0000_H



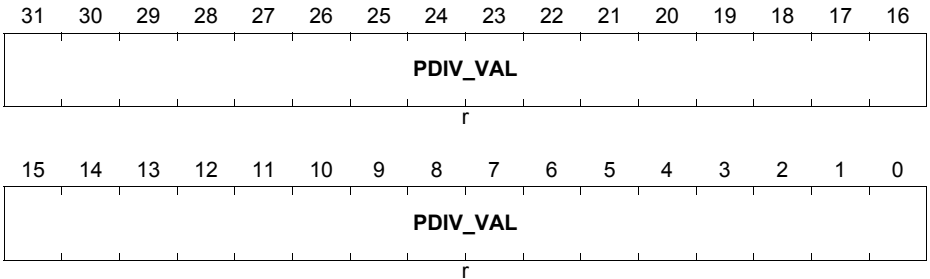
Field	Bits	Type	Description
PDIV_VAL	31:0	rw	If the Debug Path and the Timestamp Counter are enabled, the timestamp counter will be increased with every pdiv_val+1 CIF module clock cycle. Reset: 0 _B

Debug Path Predivider Counter Status Register

CIFDP_PDIV_STAT

**Debug Path Predivider Counter Status Register
(000C_H)**

Reset Value: 0000 0000_H



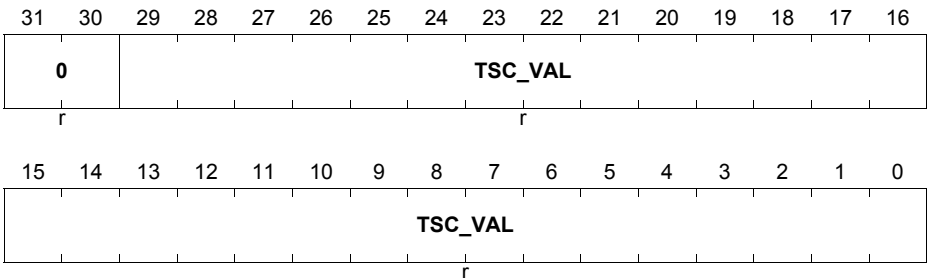
Field	Bits	Type	Description
PDIV_VAL	31:0	r	Returns the current value of the Predivider Counter. Reset: 0 _B

Debug Path Timestamp Counter Status Register

CIFDP_TSC_STAT

**Debug Path Timestamp Counter Status Register
(0010_H)**

Reset Value: 0000 0000_H



Field	Bits	Type	Description
TSC_VAL	29:0	r	Returns the current value of the Timestamp Counter. Reset: 0 _B
0	31:30	r	Reserved Read as 0, should be written with 0.

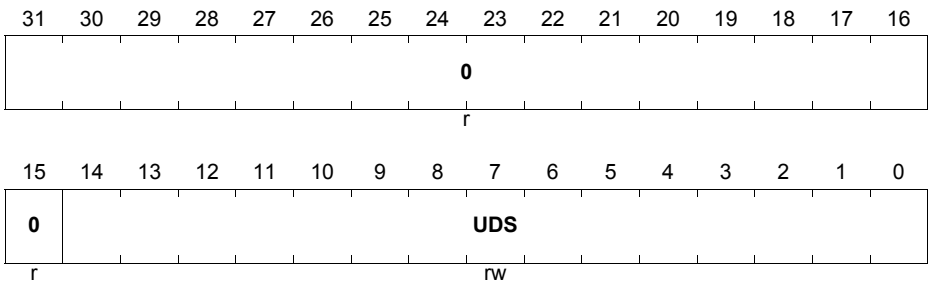
Debug Path User Defined Symbols Registers

User Defined Symbol x Register

CIFDP_UDS_x (x=1-8)

Debug Path User Defined Symbol x Register

 $(0010_H + x*4_H)$

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
UDS	14:0	rw	User Defined Symbol which may be inserted into debug stream Reset: 000 _H
0	31:15	r	Reserved Read 0, should be written with 0.

36.1.4 CIF Module Implementation

This section describes the implementation and integration details of the CIF module.

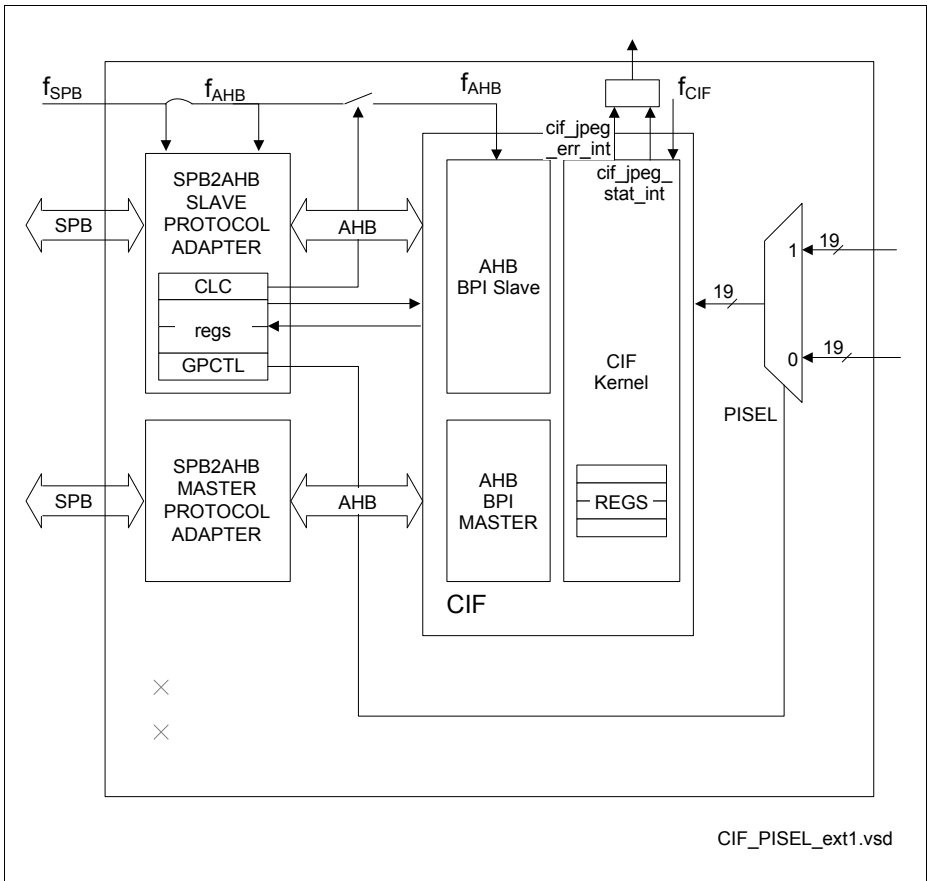


Figure 36-20 Overview of the SPB2AHB adapter

*Note: Normally only one pin mapping of CIF signals to pins is available, selected by **CIFBBB_GPCTL**. PISEL default value of 0. The second selection is tied to low.*

36.1.4.1 BPI_SPB Module Registers

This section describes the registers implemented in the slave (SPB2AHB) adapter component.

In this specification release it is assumed that the BPI_SPB registers will use reserved addresses in the module address space, and not separate address page. This issue is subject of discussion and may change.

System Registers

Figure 36-21 shows all registers associated with the BPI_SPB module, configured for one kernel. The Offset Address in the following BPI_SPB register table and in the BPI_SPB register descriptions are proposals. In a standard 32 bit peripheral the CLC is mapped to offset address 00h, module ID to 08h.

BPI_SPB Registers Overview

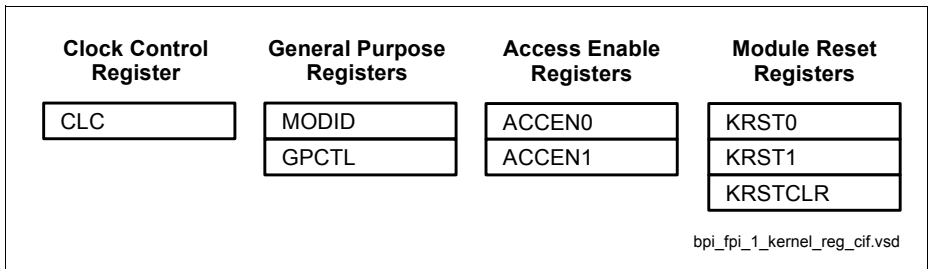


Figure 36-21 BPI_SPB Registers

Standard Registers

The adapter will implement the following standard registers as defined in the specification for the SPB BPI:

- MODID, Module ID¹⁾
- CLC, Clock Control
- ACCEN0, Access Enable 0
- ACCEN1, Access Enable 1
- KRST0, Kernel Reset 0
- KRST1, Kernel Reset 1
- KRSTCLR, Kernel Reset Clear
- GPCTL Register

Instantiation of the CLC register is controlled by a VHDL generic. Instantiation of the KRST0, KRST1, and KRSTCLR registers is controlled by a second VHDL generic. The MODID, ACCEN0 and ACCEN1 registers will always be present.

If a particular register is not instantiated then any associated output ports on the adapter entity will be driven to a constant logic value.

1) It is possible to set the fields of the module ID register without altering the adapter RTL by setting VHDL generics on the adapter instantiation

Kernel Reset

If a kernel reset is requested, the adapter will synchronously assert the internal kernel reset output and error any ongoing accesses on the SPB bus not addressed to the adapter's internal SFRs.

The method of using the kernel reset output from the adapter to initialise the associated AHB module is module specific and outside the scope of this specification.

Any AHB accesses in progress will be synchronously terminated.

GPCTL

There is a single adapter specific SFR implemented in the adaptor.

The SFR contains 32 general purpose read/write control bits without ENDINIT protection. All the bits are connected to output ports on the SPB2AHB entity. The SFR will implement no safety requirements and is not be suitable for controlling hardware related to safety functions. The SFR supports, byte, half word and word transactions only. All other transactions are rejected with an SPB error termination.

Note: In the CIF module, this register is reserved.

Principle of Operation

When the controlling state machine detects an SPB access, it compares the address to the address of the SFR. This address is configurable via VHDL generic when instantiating the adaptor. In the case that the address matches the SFR address, the normal state machine transitions are interrupted and the access is not passed to the AHB bus.

If the access opcode is not SDTB, SDTH or SDTW, then the access is terminated with an SPB error. If the opcode is supported, the SFR data will be returned on the SPB bus if the transaction is a read or the appropriate bits of the register will be updated if the access is a write.

Clock Control Register (CLC)

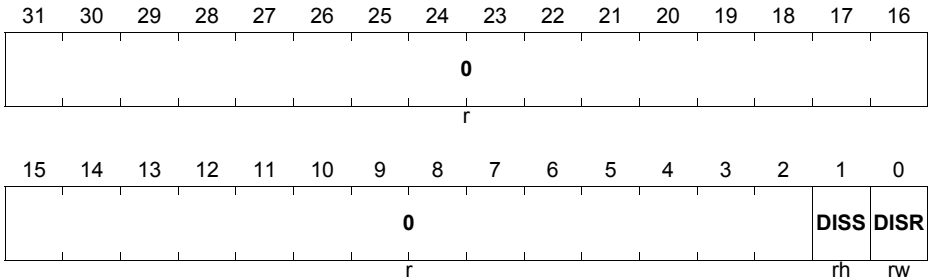
The Clock Control Register, CLC, acts globally and allows the complete AHB module to be disabled to reduce power consumption when the module is not required. When the module is disabled ($DISS=1_B$), only register accesses to the adapter register address space are permitted. All other accesses to the module address space will be errored.

CIFBBB_CLC

Clock Control Register

(00_H)

Reset Value: 0000 0003_H



Field	Bits	Type	Description
DISR	0	rw	Module Disable Request Bit Used for enable/disable control of the module.
DISS	1	rh	Module Disable Status Bit Bit indicates the current status of the module.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Module Identification Register (ID)

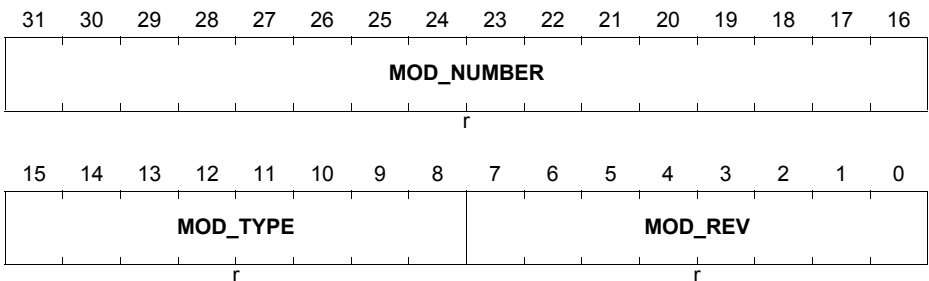
The identification register allows the programmer version-tracking of the module. The table below shows the identification register which is implemented in the DMA module.

CIFBBB_MODID

Module Identification Register

(04_H)

Reset Value: 00B3 C0XX_H



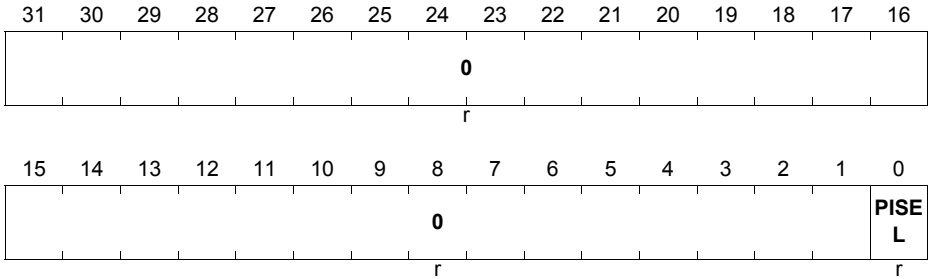
Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number This bit field defines the module revision number. The value of a module revision starts with 01H (first revision).
MOD_TYPE	[15:8]	r	Module Type The bit field is set to C0H which defines the module as a 32-bit module.
MOD_NUMBER	[31:16]	r	Module Number Value This bit field defines a module identification number.

General Purpose Control Register

CIFBBB_GPCTL

General Purpose Control Register (08_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
PISEL	0	r	Port Input Select Bit Selects between two pin mappings of the CIF interface. 0 _B pin mapping 0 selected (default) 1 _B pin mapping 1 selected
0	[31:1]	r	reserved Reads 0. Should be written with 0.

Access Enable Register 0

The Access Enable Register 0 controls write access for transactions to registers with the on chip bus master TAG ID 000000B to 011111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). The adapter is prepared for an 6 bit TAG ID. The registers ACCEN0 / ACCEN1 are providing one enable bit for each possible 6 bit TAG ID encoding.

Mapping of TAG IDs to ACCEN0.ENx: EN0 -> TAG ID 000000B, EN1 -> TAG ID 000001B, ... ,EN31 -> TAG ID 011111B.

CIFBBB_ACCEN0

Access Enable Register 0

(0C_H)

Reset Value: FFFF FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN3	EN3	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN2	EN1	EN1	EN1	EN1
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN1	EN1	EN1	EN1	EN1	EN1	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
5	4	3	2	1	0										
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Field	Bits	Type	Description
ENn (n = 0-31)	n	rW	<p>Access Enable for Master TAG ID n</p> <p>This bit enables write access to the module register addresses for transactions with the Master TAG ID n</p> <p>0_B Write access will not be executed. Read accesses will be executed.</p> <p>1_B Write and read accesses will be executed</p>

Access Enable Register 1

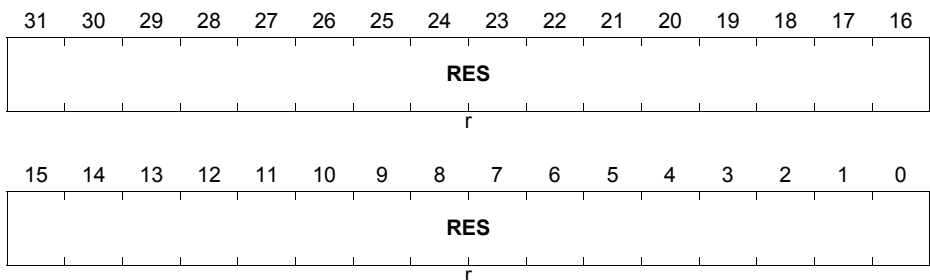
The Access Enable Register 1 controls write access for transactions with the on chip bus master TAG ID 100000B to 111111B (see On Chip Bus chapter for the products TAG ID <-> master peripheral mapping). These tags are not used in this system and so no programmable bits are provided.

CIFBBB_ACCEN1

Access Enable Register 1

(10_H)

Reset Value: 0000 0000_H



Field	Bits	Type	Description
RES	31:0	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 0 (KRST0)

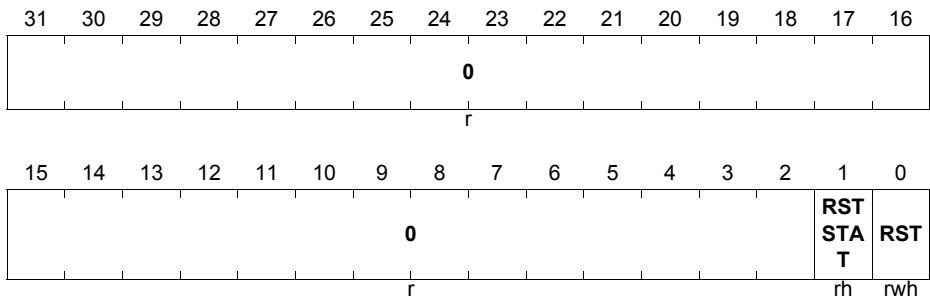
The Kernel Reset function is used to synchronously reset the AHB module. To activate the kernel reset, it is necessary to set the RST bits by writing with 1_B in both Kernel Reset Registers. The RST bit will be re-set by the adapter with the end of the adapter kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to 1_B in the same clock cycle the RST bit is reset. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLE.CLR register bit.

During the execution of the kernel reset until RSTSTAT is set, write accesses to the module registers will result in an error acknowledge. Adapter registers can still be accessed.

CIFBBB_KRST0
Kernel Reset Register 0

 (14_H)

 Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. 0 _B No kernel reset was requested 1 _B A kernel reset was requested The RST bit will be cleared (reset to 0 _B) b after the kernel reset was executed.
RSTSTAT	1	rh	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set after the execution of a kernel reset in the same clock cycle both reset bits. 0 _B No kernel reset was executed 1 _B Kernel reset was executed This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register.
0	[31:2]	r	Reserved Read as 0; should be written with 0.

Kernel Reset Register 1 (KRST1)

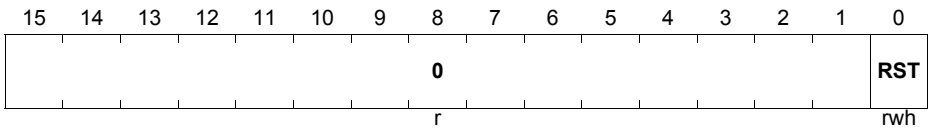
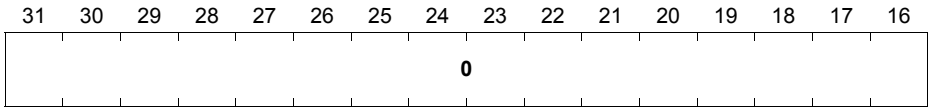
The Kernel Reset Register 1 is used to reset the related module kernel. Kernel registers related to the Debug Reset (Class 1) are not influenced. To reset a module kernel it is necessary to set the RST bits by writing with '1' in both Kernel Reset registers. The RST bit will be re-set by the BPI with the end of the BPI kernel reset sequence.

CIFBBB_KRST1

Kernel Reset Register 1

(18_H)

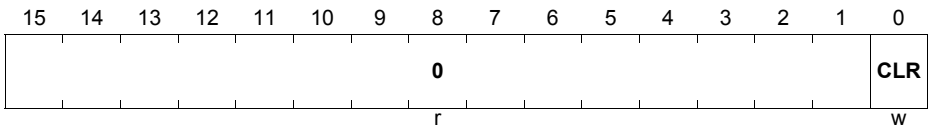
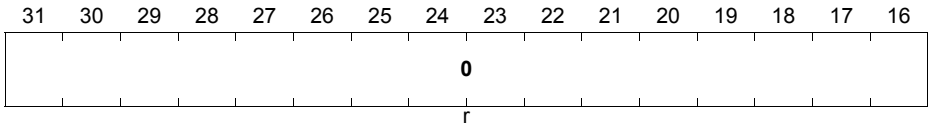
Reset Value: 0000 0000_H



Field	Bits	Type	Description
RST	0	rwh	<p>Kernel Reset</p> <p>This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers is set.</p> <p>0_B No kernel reset was requested</p> <p>1_B A kernel reset was requested</p> <p>The RST bit will be cleared (re-set to 0_B) after the kernel reset was executed.</p>
0	[31:1]	r	<p>Reserved</p> <p>Read as 0_B; should be written with 0_B.</p>

Kernel Reset Status Clear Register (KRSTCLR)

The Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

CIFBBB_KRSTCLR
Kernel Reset Status Clear Register (1C_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT Read always as 0 _B .
0	[31:1]	r	Reserved Read as 0 _B ; should be written with 0 _B .

36.1.5 CIF Programming Hints

36.1.5.1 Configuration and Shadow Registers

Most of the configuration registers inside the CIF hold static programming information (e.g. horizontal and vertical synchronization polarities at the sensor interface). But, in some cases it is required to dynamically re-program the active processing data path with new parameters (e.g. for zooming by re-defining the window). All of these new parameters must not be changed during the processing of a frame, but between frames or at the end of a frame.

Re-programming is done by a CPU attached to the Bus slave interface. Because it can not be ensured that the complete re-programming can be done during the vertical blanking period between two frames shadow registers have been introduced. These shadow registers hold the configuration values that are actually used for processing. Their corresponding configuration registers (e.g. ISP_OUT_H_OFFSETS_SHD and ISP_OUT_H_OFFSETS) store the new values until the configuration update is triggered.

For initial programming after reset or for complete re-programming of the CIF (e.g. changing from viewfinder mode to snapshot mode and vice versa) there exists a `cfg_upd` trigger bit in the main control register of each sub-module making use of shadow registers. Writing a 1 into this field copies the configuration register value into the shadow register. Mostly the main control register also has a shadow register. Other control bits set during the same write transfer as the `cfg_upd` bit are also copied to the shadow register.

Another method of shadow register update is used for dynamically re-programming the data path (see [Figure 36-22 “Automatic Data Path Re-Programming” on Page 36-239](#)). In this case the ISP, which is always the first part of the pipeline, triggers the configuration update for the whole pipeline by setting the `gen_cfg_upd` bit in the ISP main control register `ISP_CTRL`. This causes a configuration update pulse to be sent through the pipeline together with the last pixel of a frame. Each sub-module of the active processing pipeline updates its shadow registers after processing the last pixel of a frame and sends the configuration update pulse to the next stage of the pipeline.

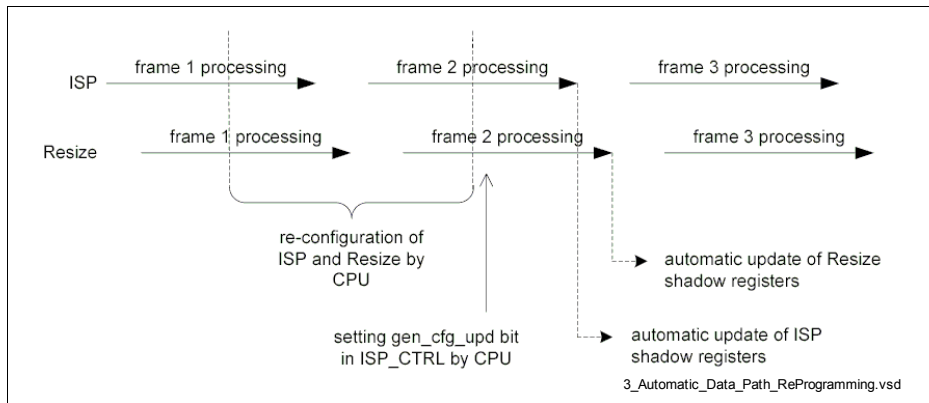


Figure 36-22 Automatic Data Path Re-Programming

Some configuration registers exist that can be changed during processing without use of shadow registers. These are for example adjustment registers in the ISP or the color processing unit which have immediate influence of the processing in terms of pixel calculations. But, no processing errors or unknown formats will result in re-programming of these register. So it is safe to modify them at any time.

36.1.5.2 General Setup for Operation

The basic configuration of the CIF must be done in idle mode. This mode can be entered by either:

- Applying the asynchronous system reset

- Software reset which is combined with the asynchronous system reset
- Applying software resets to all CIF sub-modules (refer to VI_IRCL register)
- Disabling the ISP by setting `isp_en` in `ISP_CTRL` to 0 and waiting for “isp_off” IRQ

In idle mode the following sub-modules must be configured: ISP, memory interface and control unit for image capturing to memory.

The configuration steps, which are performed by register accesses via the Bus slave interface, are the following:

- Set CIF to idle state
- Image acquisition configuration:
 - Set the input acquisition window depending on the attached camera device (`ISP_ACQ_H_OFFS`, `ISP_ACQ_V_OFFS`, `ISP_ACQ_H_SIZE`, `ISP_ACQ_V_SIZE`)
 - Define synchronization signal mode and polarity, data sample edge referring to the sensor clock (`ISP_ACQ_PROP`), and number of frames to be captured (`ISP_NR_FRAMES`)
- Image windowing setup:
 - Define the ISP output window (`ISP_OUT_H_OFFS`, `ISP_OUT_V_OFFS`, `ISP_OUT_H_SIZE`, `ISP_OUT_V_SIZE`)
- Immediate ISP configuration update (`cfg_upd = 1` in `ISP_CTRL`)
- Data path configuration:
 - Depending on the use case the data path must be configured for the desired operation (`VI_DPCL`)
- Main picture memory buffer configuration:
 - Define luminance (Y) buffer base address and size in system memory (`MI_MP_Y_BASE_ADDR`, `MI_MP_Y_SIZE`)
 - Define Y data start offset relative to the Y buffer (`MI_MP_Y_OFFS`)
 - Define chrominance blue (C_B) buffer base address and size in system memory (`MI_MP_CB_BASE_ADDR`, `MI_MP_CB_SIZE`)
 - Define C_B data start offset relative to the C_B buffer (`MI_MP_CB_OFFS`)
 - Define chrominance red (C_R) buffer base address and size in system memory (`MI_MP_CR_BASE_ADDR`, `MI_MP_CR_SIZE`)
 - Define C_R data start offset relative to the C_R buffer (`MI_MP_CR_OFFS`)
- Immediate memory interface configuration update:
 - Define to update base and offset shadow registers (`init_base = 1` and `init_offset = 1` in `MI_INIT`)
 - Update shadow registers for video (in) and Bus part (out) (`cfg_upd_mi_in = 1` and `cfg_upd_mi_out = 1` in `MI_CTRL`)
- Enable/disable interrupts (`ISP_IMSC`, `MI_IMSC`)

Now the basic setup of all necessary registers is done.

For programming a specific use case the data path needs to be configured in the right way (please refer to chapter **“Use Case Description” on Page 36-262**). Additionally not

active sub-modules can be switched off as described in chapter **“Power Management”** on Page 36-265.

36.1.5.3 Start-Stop Programming

Data capturing controlled by the ISP

Data capturing is controlled by the ISP only. All the other sub-modules in the processing pipeline automatically stop processing if no video data is available.

The CIF starts and stops video data capturing in a frame synchronized way. So the time of enabling data capturing is not important as the ISP waits for the beginning of a frame before sampling data. The same applies for stopping video data capturing. Regardless of the current pixel position in the frame or outside the frame the current frame is always captured completely before disabling the ISP.

The ISP contains an input acquisition and an output formatting unit. Both units are able to start and stop video processing frame synchronously. For video capturing both units must be enabled, but for disabling capturing at least one of them must be switched off.

Note: It is highly recommended to use the output formatter of the ISP for controlling video processing, because this unit is able to generate an “isp off” interrupt based on the ISP output frame after windowing.

Additionally it is possible to specify the number of frames to be captured by writing this number into ISP_NR_FRAMES. Continuous capturing is enabled by writing a 0 into this register.

So to enable the ISP input_en in ISP_CTRL must be set to 1 first. At this point the ISP starts capturing data but does not provide them at its output interface. The output interface is part of the output formatter which also does the output windowing. It is enabled setting isp_en in ISP_CTRL to 1.

To stop video capturing it is sufficient to disable the ISP output formatter by setting isp_en in ISP_CTRL to 0, or if a certain number of frames to be captured have been specified, the ISP automatically disables its output formatter. Then the “isp off” interrupt is generated after putting out the last pixel of the last frame.

Note: After stopping data capturing at the ISP the processing pipeline is still working for a while because of the pipeline delay of some pixel lines. This delay depends on the mode of operation. The maximum delay is about 11 lines.

The pipeline is clean after either applying the software reset to the involved sub-modules (resulting in loss of data in memory), or it must be waited until the complete frame has been written into system memory. The latter is done by waiting for the “frame end” interrupt of the output unit.

36.1.5.4 Abort of Processing

Complete abort of processing can be either done by applying the asynchronous system reset to the CIF, the software reset for the entire CIF or by applying the software resets to all involved sub-modules which is the preferred method.

Frame Skip

For the main picture data path another way of discarding pixel data exist: frame skipping. This can be used if e.g. the video encoding device can not keep up with the frame rate provided by the CIF. In this case the system controller has to set the skip bit in the MI_INIT register of the memory interface.

For frame skipping the memory interface discards all remaining pixels of the current frame at its main picture input ports. It only completes the current burst transfer and flushes all main picture internal data FIFOs. Resetting of the related offset data pointers for the buffers depends on the value of `init_offset` in `MI_CTRL`. If `init_offset` is set to 1 the buffer offset counters are reset to the programmed values in `MI_MP_Y/CB/CR_OFFS`. If `init_offset` is set to 0 the offset counters for all buffers are set to the beginning of the current (skipped) frame.

Frame skipping does not affect "frame end" interrupt generation.

Handling Picture Size Error

Aborting the ISP processing becomes necessary, if an input picture size violation is detected by the hardware. In this case the interrupt `PIC_SIZE_ERR` is raised, and the ISP does not generate a frame end interrupt, even if the picture size is corrected. To recover from this condition, a software reset of the ISP is necessary. After a reset the picture size needs to be programmed to the appropriate values and processing can be started again.

To find the reason behind the picture size error the register `ISP_ERR` might be helpful. This register provides the information about where exactly the size error occurred.

36.1.5.5 Interrupt Handling

The CIF provides two programmable level active interrupt outputs driven by the ISP and memory interface sub-modules. Interrupt control and status registers are located locally in their originating sub-modules. All sub-module masked interrupts are ORed together to drive the interrupt line.

Five interrupt registers are provided for each interrupt line per block able to generate interrupts:

- Interrupt mask register (IMSC)
- Raw interrupt status register (RIS)
- Masked interrupt status register (MIS)

- Interrupt clear register (ICR)
- Interrupt set register (ISR)

The IMSC register defines if the events actually generate an interrupt. If a mask bit is set, the corresponding event is allowed to generate an interrupt. Per default all mask bits are set to 0 to prevent interrupt generation.

The RIS register shows the status of all sub-module internal events without effect of the mask register IMSC.

The MIS register shows the status of the events allowed to trigger an interrupt (MIS = RIS AND IMSC). If an interrupt has been thrown the MIS register must be read for locating the accountable event.

The ICR register is used for clearing interrupts by resetting the event. Writing a 1 into a interrupt bit resets the respective event. Reading of this register always returns "00_H".

The ISR register is used to trigger events for debug and test purposes. Writing a 1 into an interrupt bit activates the associated event and an interrupt will be thrown if the mask register is set accordingly. Reading this register always returns "00_H".

ISP Events

The ISP generates events at several positions during input and output data formatting. **Figure 36-23** shows the trigger points for the following events:

- ISP returns to idle state (isp_off)
- Output frame end reached (out_frame_end)
- Input frame end reached (in_frame_end)

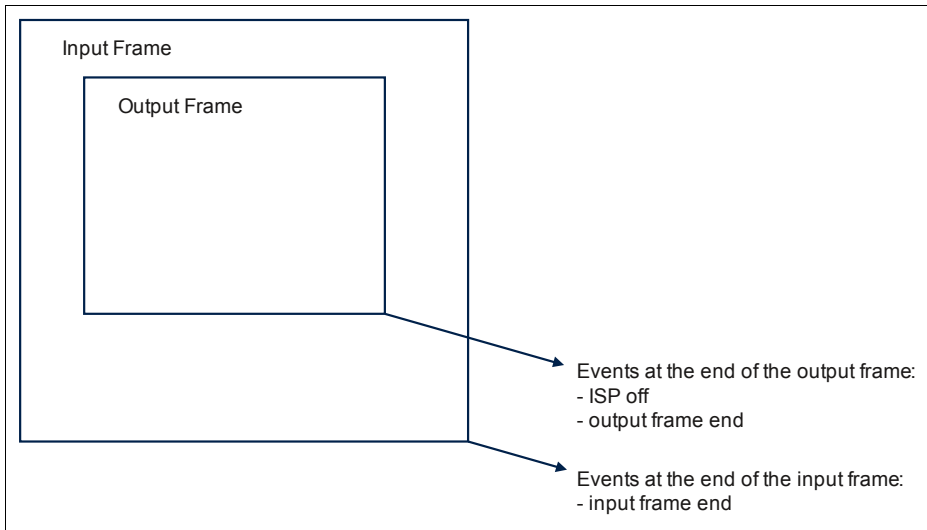


Figure 36-23 Frame Based ISP Events

Additionally the following error events may be generated:

- Input picture size violation detected (`pic_err`)
- Loss of data detected during an active (`data_loss`)

These two events are immediately triggered as they occur. For more information on ISP events please refer to chapter **“ISP Programming” on Page 36-246**.

Memory Interface (MI) Events

The MI provides events from several internal sources (see **Figure 36-24 “Memory Interface Events” on Page 36-245**) supporting video encoding as well as signalling error and transfer end conditions:

- Bus write error detected (`write_err`)
- Main picture buffer wrap-around occurred for Y, C_b , C_r buffers (`wrp_m_y`, `wrp_m_cb`, `wrp_m_cr`)
- Programmed fill level reached in main Y buffer (`fill_m_y`)
- Next macro-block line stored (`mblk_line`)
- Last pixel of image or RAW data stream is written into memory (`frame_end`)

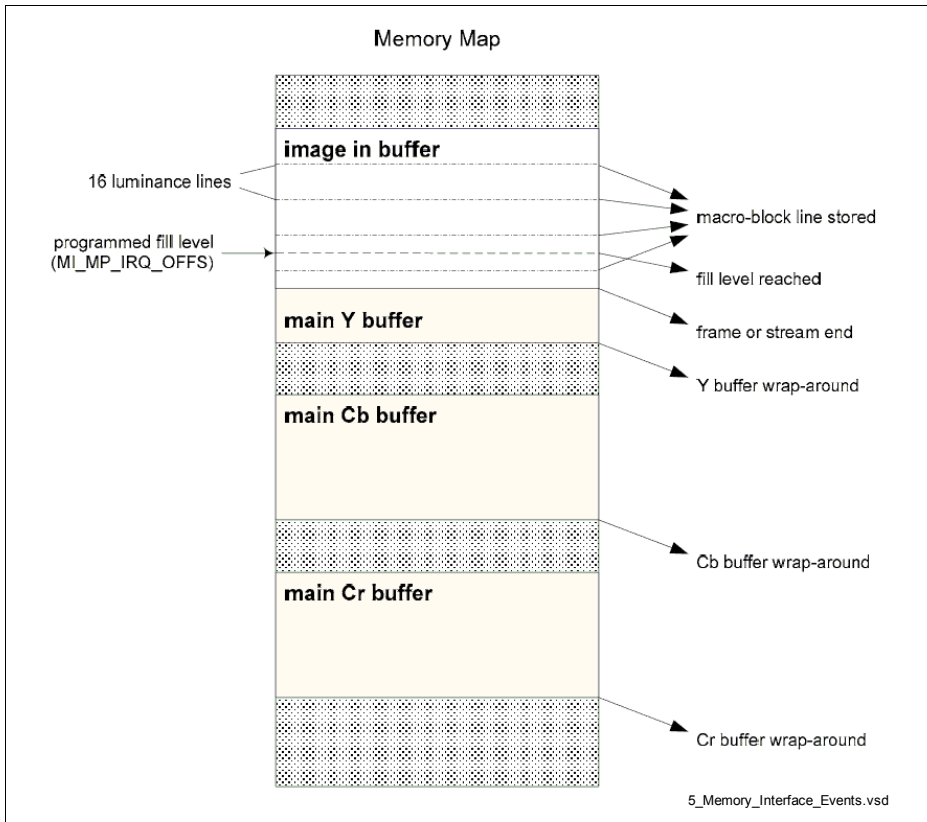


Figure 36-24 Memory Interface Events

The “macro block line” and “frame end” events not only depend on the Y buffer, but also on C_b and C_r buffers because all pixel components have to be written to memory before triggering the appropriate event. For “frame end” this synchronization works for all supported YC_bC_r formats (4:2:2, 4:2:0). The “macro-block line” event relies on 16 luminance and 8 chrominance lines, which are in YC_bC_r 4:2:0 format.

For better debugging in MI an additional status register is implemented. The MI shows status information of the FIFOs in the MI_STATUS register and if a read or write Bus error occurred. This registers can be cleared with the according MI_STATUS_CLR register. Usage of this register should be as follows: If an error interrupt was triggered from the sensor FIFO of the ISP (DATA_LOSS interrupt), then the MI_STATUS register should be checked to find out the reason for the error. One reason could be the latency of the Bus bus, which results in FIFO full bits set in the MI_STATUS register.

36.1.5.6 Reset Handling

Three types of resets are supported by the CIF:

- Asynchronous system reset
- Software reset for the entire CIF
- Independent software resets for each sub-module

The asynchronous reset is a low active system reset for resetting all flip-flops inside the CIF. After an asynchronous reset no processing, status, or configuration information is kept.

The same behaviour as for “asynchronous system reset” is true for the software reset bit of the entire CIF. All status and configuration register information is reset. After applying this reset the software must wait at least 10 module clock cycles before the CIF can be reprogrammed.

The independent software resets for each sub-module are meant for just resetting processing and status information, but not configuration register content. So, the software reset is first choice if a processing error occurs and the system should be re-initialized without need to fully re-program the device. It is triggered by single bits for all CIF sub-modules in VI_IRCL.

36.1.5.7 Programming Guide

This chapter gives a detailed overview about programming of the particular CIF sub-modules.

ISP Programming

The ISP sub-module contains input acquisition, output formatting, as well as status and error interrupt generation.

An overview of the internal processing blocks of the ISP is given in the figure **“Block Diagram of the ISP Sub Module” on Page 36-11** .

Error Handling

If an input picture size violation was detected and reported (interrupt) by the ISP, then the ISP needs to be reseted (soft-reset) in order to do the processing of the following frames. Without reset the frame end interrupt will not work correctly for the following frames, even if the picture size is correct then.

Input Acquisition

The input formatter is responsible for sampling data from the sensor device and providing it to the other blocks in the processing pipelines.

As the clock provided by the image sensor is used for data sampling, the sample edge can be selected depending on the phase shift between data and pixel clock. If `sample_edge` in `ISP_ACQ_PROP` is set to 1 the rising edge of the pixel clock is used,

otherwise the falling edge. The sample edge selection depends on the phase relationship between pixel clock and image data.

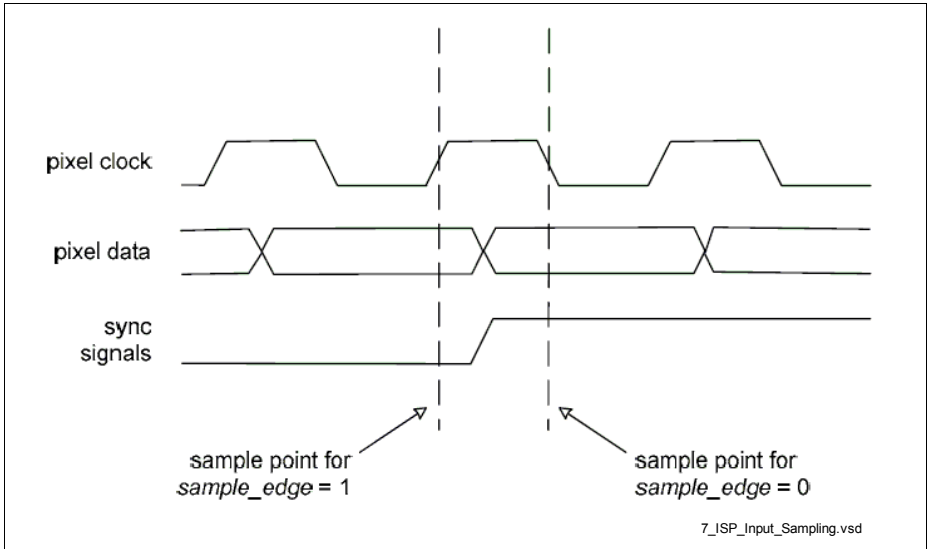


Figure 36-25 ISP Input Sampling

The input acquisition has to be adapted to the specifics of the attached sensor device in terms of image resolution, color space format, black pixel regions, etc.

In addition please make sure that the sensor device delivers at least 2 lines of vertical blanking and at least 5 pixels horizontal blanking in its video timing. This is needed for the CIF ISP to function properly.

The input acquisition window has to be defined using the registers

ISP_ACQ_H_OFFS,
 ISP_ACQ_V_OFFS,
 ISP_ACQ_H_SIZE,
 ISP_ACQ_V_SIZE.

In addition the sample window can be further cropped for the following processing chains by the registers

ISP_OUT_H_OFFS,
 ISP_OUT_V_OFFS,
 ISP_OUT_H_SIZE,
 ISP_OUT_V_SIZE.

This is necessary, because processing blocks must not be fed with images containing black pixel areas. These areas have to be cut off before the image data is transferred to the CIF's ISP.

Furthermore the polarity of the horizontal and vertical synchronization signals in ITU-R BT.601 mode have to be defined using `hsync_pol` and `vsync_pol` in `ISP_ACQ_PROP`.

Limited support for capturing of interlaced video is available. So either odd or even, or both fields can be sampled (`field_sel` in `ISP_ACQ_PROP`).

In the `ISP_CTRL` programming register the operating mode for the ISP must be selected with `isp_mode` (also refer to the following chapters).

Input formatter enabling and disabling is described in chapter [“Start-Stop Programming” on Page 36-241](#).

Output Formatting and Image Stabilization

At the output of the ISP sub module the image data may be cropped a third time by using the registers

`ISP_IS_H_OFFS`,

`ISP_IS_V_OFFS`,

`ISP_IS_H_SIZE`,

`ISP_IS_V_SIZE`.

These registers should be used, if digital zoom features shall be implemented in the system software.

Figure 36-26 illustrates the different possible regions which can be defined by the corresponding register settings. Some of these regions may have the same size and offset (settings), if a differentiation is not needed.

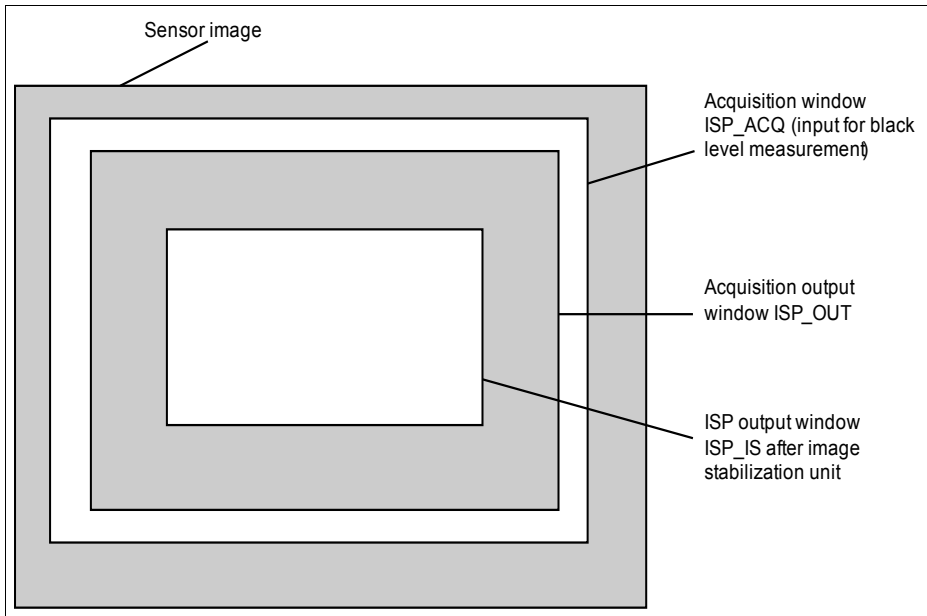


Figure 36-26 Possible cropping regions of ISP

In addition the output cropping module is designed to support image stabilization features for video capturing.

For image stabilization a sub frame must be chosen from the input frame, because a certain border around the output image is needed to have some margin for correcting the position of the image. Based on externally generated camera global motion data, image stabilization offers the functionality to compensate for that camera motion by moving the chosen sub frame across the input frame according to the signalled camera motion. The information source for global motion may be a motion sensor or extracted from the image content by a video encoder. The global motion vector has to be written into the configuration register `ISP_IS_DISPLACE` by software for each frame. To prevent the sub frame from running out of the input frame, the sub frame is re-centered at a programmable rate that is proportional to its distance from the center of the input frame. The rate is programmed by setting the register `ISP_IS_RECENTER` with appropriate values.

In addition, the maximum allowed displacement of the sub window is programmable by using `ISP_IS_MAX_DX` and `ISP_IS_MAX_DY` registers. The output picture size is using shadow registers. If the programmed size can't be reached due to incorrect register programming or wrong input formats, a picture size error interrupt is triggered.

In standard mode (image stabilization switched off: `is_en=0` in `ISP_IS_CTRL`), the registers `ISP_IS_H_OFFS`, `ISP_IS_V_OFFS`, `ISP_IS_H_SIZE`, and `ISP_IS_V_SIZE` are used to cut a programmable sub frame out of the input frame as described above.

If image stabilization is switched on, the horizontal and vertical offset of the chosen sub frame is automatically updated. This automatic update according to camera displacements `dx` and `dy` programmed to the `ISP_IS_DISPLACE` register is implemented as described below.

Instead of using the values programmed to the configuration registers `ISP_IS_H_OFFS` and `ISP_IS_V_OFFS`, (which are now used to configure the initial position of the chosen sub frame) two new internal registers `cur_h_offs` and `cur_v_offs` are used to dynamically choose the sub frame sent to the output interface. These are updated with each frame in the following way:

if `IS_RECENTER != 0`

$$\text{cur_h_offs} = (\text{cur_h_offs} - \text{dx}) - ((\text{cur_h_offs} - \text{IS_H_OFFS}) / 2^{\text{IS_RECENTER}})$$

else

$$\text{cur_h_offs} = (\text{cur_h_offs} - \text{dx})$$

`cur_h_offs` is clipped to the range

`[IS_H_OFFS - IS_MAX_DX ... IS_H_OFFS + IS_MAX_DX]`.

The system software has to ensure that `IS_H_OFFS` is greater or equal `IS_MAX_DX` and that `IS_H_SIZE + IS_H_OFFS + IS_MAX_DX` does not exceed the horizontal size of the input frame. Recommended value for `IS_MAX_DX` is equal to `IS_H_OFFS`.

if `IS_RECENTER != 0`

$$\text{cur_v_offs} = (\text{cur_v_offs} - \text{dy}) - ((\text{cur_v_offs} - \text{IS_V_OFFS}) / 2^{\text{IS_RECENTER}})$$

else

$$\text{cur_v_offs} = (\text{cur_v_offs} - \text{dy})$$

`cur_v_offs` is clipped to the range

`[IS_V_OFFS - IS_MAX_DY ... IS_V_OFFS + IS_MAX_DY]`.

The system software has to ensure that `IS_V_OFFS` is greater or equal `IS_MAX_DY` and that `IS_V_SIZE + IS_V_OFFS + IS_MAX_DY` does not exceed the vertical size of the input frame. Recommended value for `IS_MAX_DY` is equal to `IS_V_OFFS`.

The system software has to ensure that the dx and dy values in ISP_IS_DISPLACE register are always up to date. Especially when no movement was detected the values have to be set to zero.

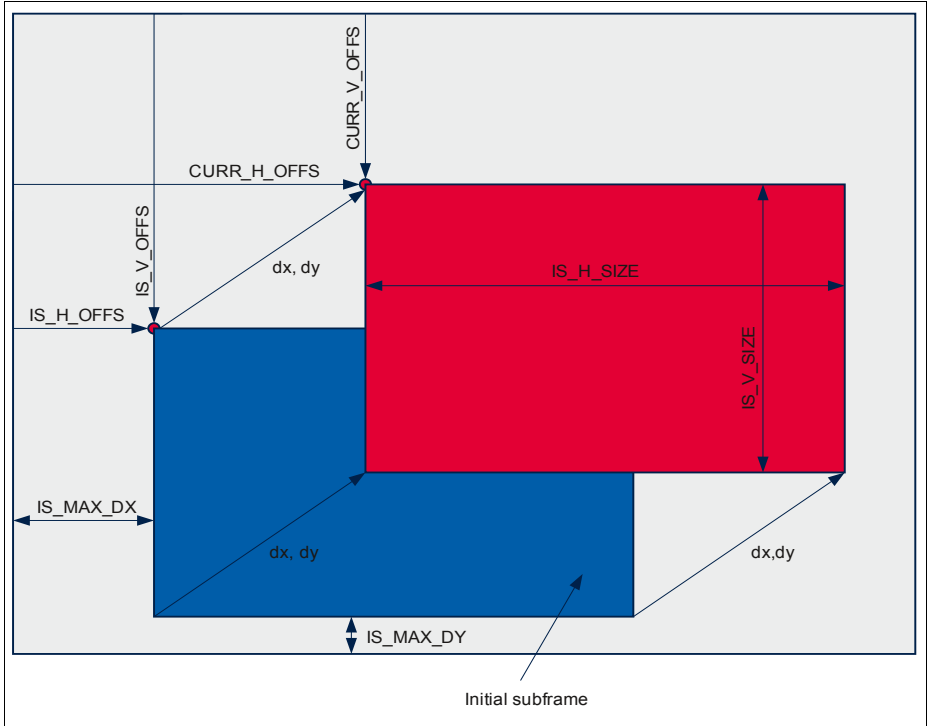


Figure 36-27 Illustration of image stabilization parameters

Data Mode

In data mode all input data are considered as bytes. No line or frame organization is expected any more, so the states of the hsync and vsync hardware lines are considered as "byte enable" and "transfer indicator". This mode allows connecting the CIF to any source delivering a data stream (e.g. camera sensor including a JPEG encoder). This mode is selected by programming a "4" into the `isp_mode` field of `ISP_CTRL`. In this mode no limitation for the amount of data exists. The data will be stored in system memory in the region "main Y buffer".

As the synchronization signal polarities are programmable, the same applies to byte enable and transfer indicator as well. Also the sample edge can be selected by programming the register `ISP_ACQ_PROP` accordingly.

To start a transfer sequence, the transfer indicator (vsync) and byte enable (hsync) may be asserted (driven to an active level, usually low) at the same time or one after another in the following order: first assert vsync, then assert hsync. The transfer end condition is detected by the CIF when these signals are de-asserted as shown in [Figure 36-28](#). Detecting the transfer end condition will lead to a main-path frame-end interrupt as soon as the last byte of this transfer is written to the system memory. During a transfer sequence, sampling of the input data can be paused by de-asserting the hsync for the desired amount of clock cycles.

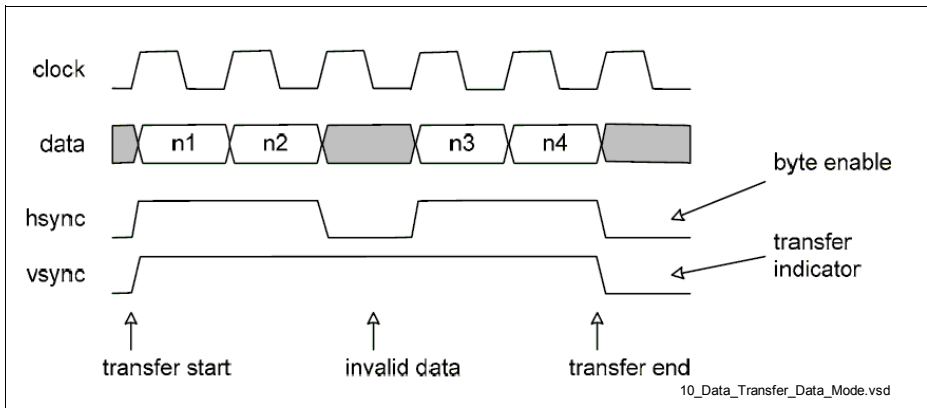


Figure 36-28 Data Transfer in Data Mode with Active High Level

Note that because there are no line and frame structures in the data mode any more, there is also no frame synchronized start mechanism as described above. Thus, the time of enabling the ISP is important here, because if the transfer indicator line

Is asserted at the moment of enabling the ISP, data capturing would start right away. If the entire amount of bytes of a transfer sequence is to be captured, the ISP must be enabled while the transfer indicator is de-asserted.

In situations with periodically repeated transfer sequences, the suggested strategy of capturing one entire sequence is as follows:

- Setup the complete processing chain including input acquisition, data path and output buffer size and address.
- Setup the memory interface to skip the next incoming frame (i.e. suppress the actual writing in system memory)
- Start the ISP for one frame only and wait for the frame-end interrupt. Because starting the ISP was done unsynchronized with the transfer sequence repetition, the CIF usually won't sample the entire number of bytes of this sequence. But this is not important, because we have suppressed writing to memory anyway, and are only interested in the frame-end-notification interrupt.

- As soon as the frame-end interrupt occurs, we know for sure that the last transfer sequence has just been completed. Now re-program the output buffer address, start the ISP for another single frame, and wait again for the frame-end notification.
- Upon occurrence of the frame-end interrupt, the entire number of data bytes from the last transfer sequence can be found in the output buffer.

In data mode, the values of the registers `ISP_ACQ_V_OFF`, `ISP_ACQ_H_OFF`, `ISP_ACQ_H_SIZE` and `ISP_ACQ_V_SIZE` are irrelevant.

RAW Picture Mode

RAW picture mode allows sampling of the input image according to the setting of the BT.601 sync signals and providing it directly at the output of the ISP. No data processing is done at all allowing to store the 8 up to 16-bit data samples from the sensor device directly in system memory. No luminance/chrominance separation and re-ordering are done. After writing a 0 into the `isp_mode` field of `ISP_CTRL` the ISP is set to RAW picture mode.

The maximum size of data is limited by the offset and size counters. This means a maximum of 4095x4095 bytes are possible. The image data will be stored in the main Y buffer. In case of more than 8 bits data, the data bits are stored in 16-bit words in memory. The bits are stored MSB aligned.

ITU-R BT 601/656 YCbCr4:2:2 Mode

This ITU-R BT 601/656 mode allows the connection of an image sensor with integrated ISP functionality. In this case $YCbCr$ data is expected at the sensor interface. In ITU-R BT 656 mode the synchronization signals are not considered as all timing reference data are encoded into the data stream (SAV and EAV codes). For ITU-R BT 601 mode the vertical and horizontal synchronization signals are sampled together with the image data. The synchronization signals can be interpreted as synchronization pulses, or as reference signals to be seen in the following figure.

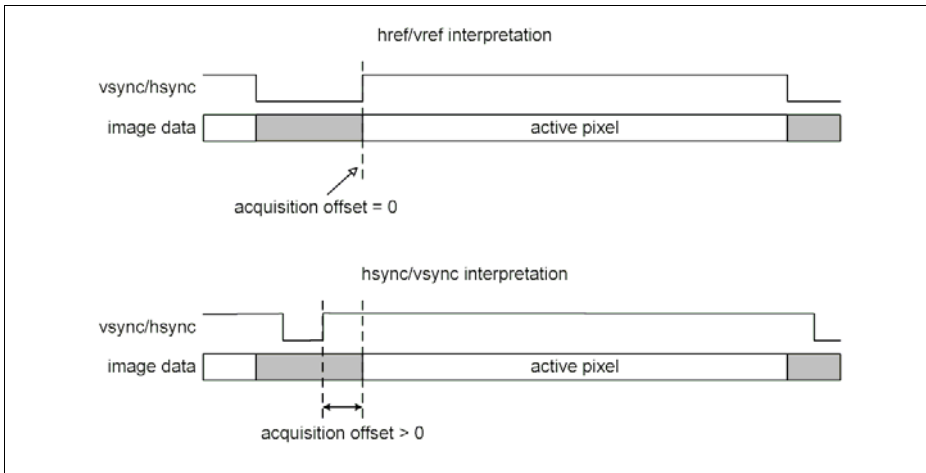


Figure 36-29 Reference and Synchronization Signals for ITU-R BT 601 Mode

Memory Interface Programming

The CIF memory interface unit (MI) is responsible for reading/writing image data from/to the system memory. As shown in Figure [“Memory Interface Unit overview” on Page 36-255](#), the MI has three main tasks, which can operate independently:

- Take image data from the main path, either in YCbCr or RAW format and write it into certain data buffers located in the system memory.

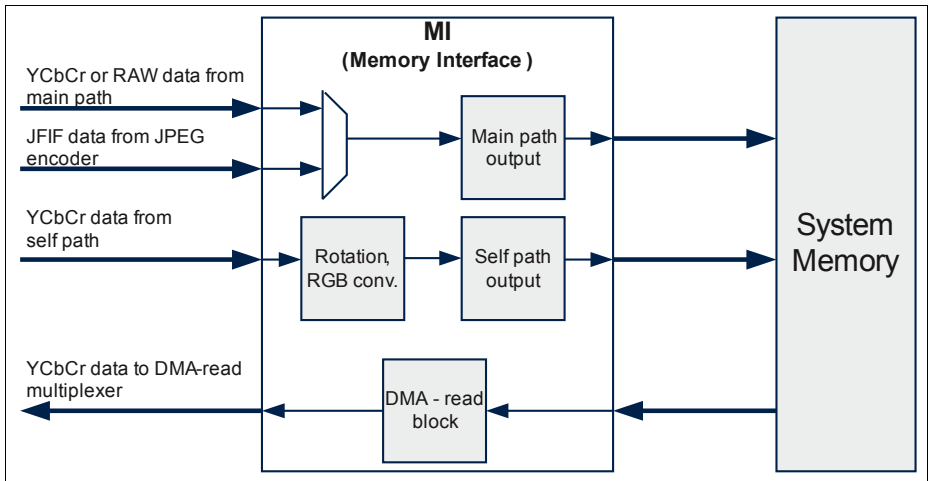


Figure 36-30 Memory Interface Unit overview

External Output Buffers

Depending on which paths are enabled and which output format is selected, the MI unit can simultaneously output image data to up to 6 independent buffers. The buffers are named according to their relation to the main data path and their usage in the (most common) YCbCr planar output format: main-Y-buffer, main-Cb-buffer, main-Cr-buffer.

The buffers itself are not part of the CIF IP; they are system memory areas accessible from the bus that is also connected to the CIF. One can think of the MI as a kind of DMA controller, which writes to the configured memory locations without any CPU assistance. To configure the memory locations, each of the buffers mentioned earlier has a set of registers assigned to it.

- MI_xx_xx_BASE_AD_INIT
 - This is the base address of the memory area to use for that buffer. (e.g. the lowest address of that buffer) Note that buffer base addresses must be 4-byte-aligned. In case the target system uses some kind of PMMU, the physical address must be specified, because virtual-to-physical translation is not possible for this kind of DMA-like transfer.
- MI_xx_xx_SIZE_INIT
 - This is the size of the memory area in bytes. Note that the size is also required to be a multiple of 4. It is ensured that for write operations in this particular buffer, the MI will only write to memory locations from the base address to the base address + the size - 1. If the MI reaches the end of the memory area while continuously writing data, the buffer "wraps around" and it proceeds to write starting from the

base address again. This wrap around event can also be used to trigger an interrupt. See the MI_RIS register for details.

- MI_xx_xx_OFFS_CNT_INIT
 - This is the initial offset (in bytes) for writing to the buffer after e.g. a soft reset. Usually, to write to the buffer from the very beginning, an offset of zero is to be programmed into this register.
- MI_xx_xx_OFFS_CNT_START
 - This is a read-only register, which holds the offset at which the MI had written the last processed frame. This register is updated at frame end, so at any given time, it contains always the starting offset of the last completely processed frame.

One of the buffers, the main-Y-buffer, is additionally able to trigger an interrupt as soon as a particular filling level is reached. This level can be configured in terms of an offset count (in bytes) from the base address using the MI_MP_Y_IRQ_OFFS_INIT register.

Further, the MI supports updating the buffer configuration seamlessly between two consecutive frames without dropping a frame by using the shadow register concept described in chapter **“Configuration and Shadow Registers” on Page 36-238**. Two bits in the MI_CTRL register (init_offset_en and init_base_en) can be used to select whether the base address and size, or the offset counters, or both are being updated at the next configuration update pulse. It should be noted that it is only possible to update the selected parts of all the buffers at once. It is not possible to re-configure only specific buffers and leave the others left untouched.

As mentioned earlier, there are some restrictions permitting only 4-byte-alignment buffer base addresses, and both sizes and offsets also need to be a multiple of 4. This is because the MI writes the data to the memory areas in 4-byte-accesses. It is possible to change the endianness of these accesses with the byte_swap bit in the MI_CTRL register. **Figure 36-31** illustrates this with the example use case of writing 8-bit Y-samples from a frame with a width of 5 pixels. The setting of the byte_swap bit affects the write accesses to all buffers regardless of whether YCbCr or RAW data is to be written. Note that "addr" is a 4-byte aligned address in this example. For all other output schemes illustrated and discussed in this chapter, a setting of byte_swap = 0 is assumed.

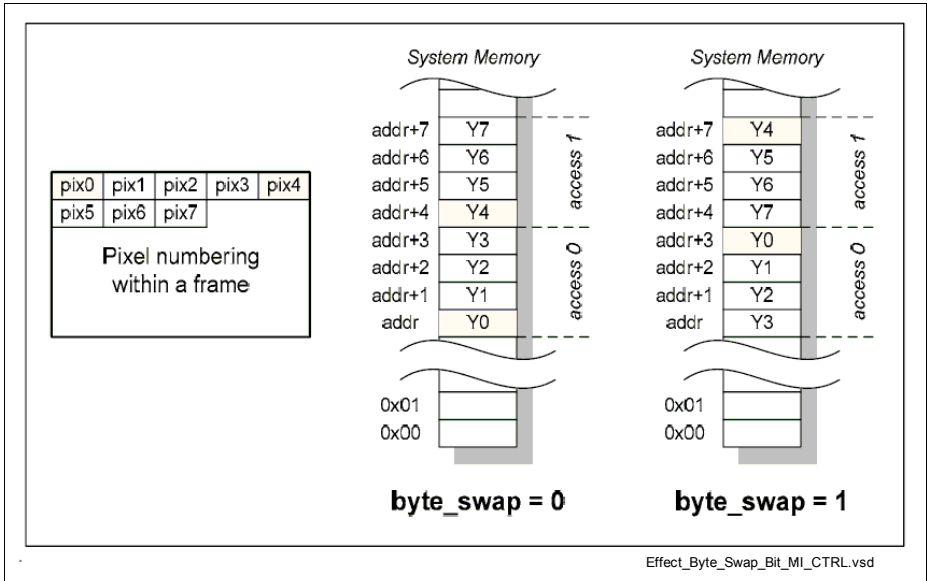


Figure 36-31 Effect of the byte_swap bit in MI_CTRL

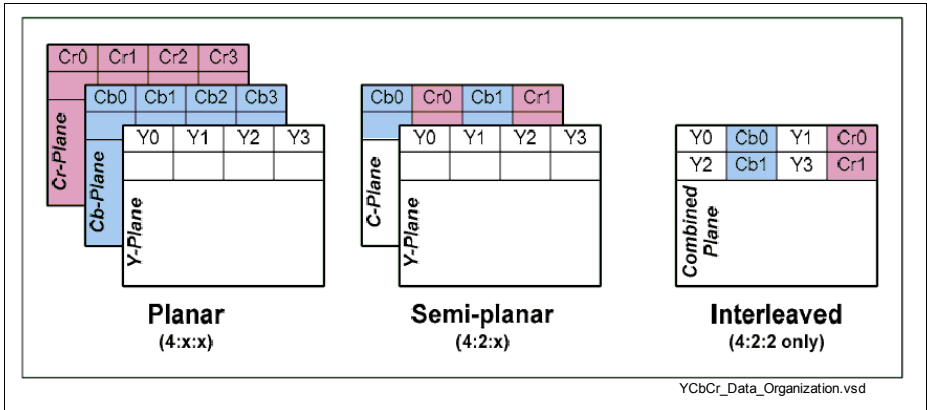


Figure 36-32 YCbCr Data organization

Main Path output programming

The main path output sub-module of the MI is intended to be used for video stream generation (YCbCr for further compressing via software encoders), or high resolution still

image snapshot . To achieve this, it can be configured to output one out of the following data sources:

- YCbCr data from the main path scaler
 - To enable this, bit `mp_enable` in the `MI_CTRL` register is to be set. The data can be output in planar, semi-planar or interleaved format, as shown in [Figure 36-32](#). The selection is done with the bits `mp_write_format` in the `MI_CTRL` register. In semi-planar mode, the main-Cb-buffer is used for the multiplexed Cb/Cr values and the main-Cr-buffer remains unused. In interleaved mode, only the main-Y-buffer is used for all the multiplexed values.
- RAW data from the main path scaler.
 - This mode is enabled by setting the bit `raw_enable` in the `MI_CTRL` register. RAW data is assumed to be received via the same way as YCbCr data, but is written to the main-Y-buffer only. in RAW mode, the `mp_write_format` bits of the `MI_CTRL` register are used to select between 8-bit mode (which will write only the 8 most significant bits of every sample) up to 16-bit mode (which will e.g. write the full 16-bit msb-aligned into 2 bytes of the output buffer). This implies that for the higher than 8-bit modes, a buffer size twice as much is necessary as for the 8-bit mode, and the least significant bits of every 16-bit value are unused and remain zero.

In some RAW data modes the actual number of bytes to write per frame depends on the input signal and/or the compression ratio, and is not known in advance. Thus, this number is given in the read-only `MI_BYTE_CNT` register, which is updated at frame end only. So this register always shows the number of bytes written to the main-Y-buffer for last completely processed frame.

It is assumed that all the sub-modules within the CIF's processing chain located in front of the MI are configured in a way that the input data to the MI main path corresponds to what it is configured for. If this is not the case, the behavior of the MI is not predictable.

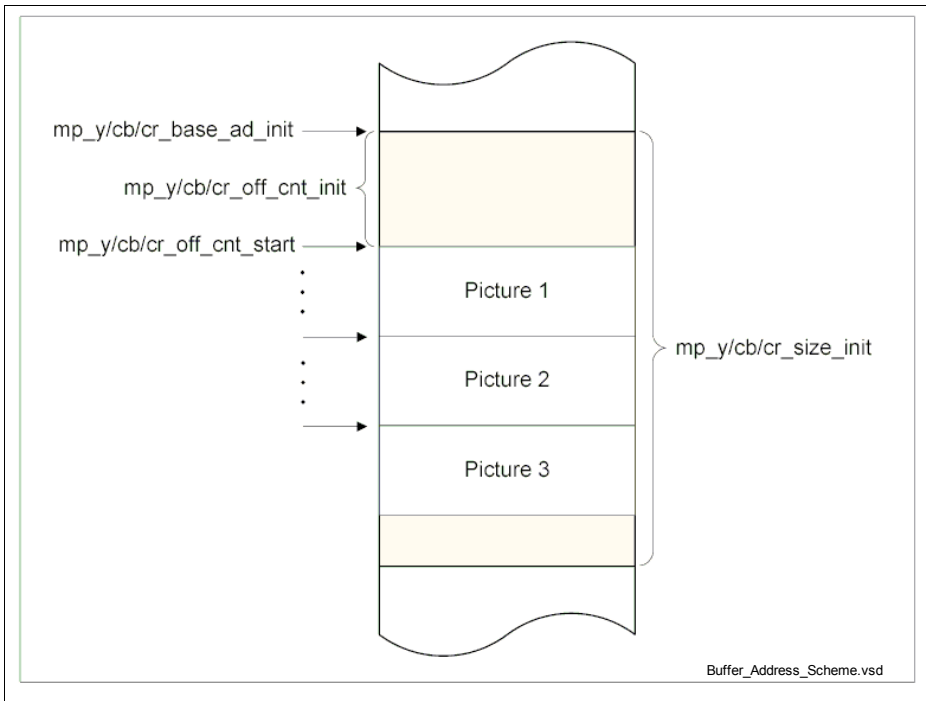


Figure 36-33 Buffer address scheme, valid for main picture buffers

Getting Started - First steps for startup

This chapter describes the first steps to startup the CIF in YCbCr mode. The procedure and the register contents are presented. Generally, all sub modules (like MI, EP, LDS) within CIF can be configured in arbitrary order but the ISP sub module must be enabled last.

General Hints

Activity of the input port (data and sync lines) can be observed through register ISP_FLAGS_SHD.

Recipe 1: YCbCr Bypass Mode

In YCbCr Bypass mode the image sensor is expected to deliver pre-processed image data in YCbCr format at its parallel output port. Because of the pre-processing, most of the CIF's ISP functionality can be bypassed, so only a small number of registers must be programmed to put this mode into operation. The Main Scaler unit will also be bypassed, so that the incoming data will be presented almost unmodified at the output. All that's left to do in the processing chain is YC-sequence reordering.

Requirements / Preconditions

In this example, we assume a sensor delivering VGA-sized (640x480) frames continuously using YCbCr data format with 4:2:2 color sub sampling. Synchronization is assumed to be done according to ITU.BT601 (i.e. using separate wires carrying hsync and vsync signals, both active high in this example).

It is further assumed that the sensor is already up and running and that it is configured to the mode described above. Its pixel clock is running and meets the CIF's AC-requirements, and the sync signals and image sensor data bus are routed to the CIF's input ports.

Basic Initialization

1. Enable the main clock in VI_CCL VI_CCL = 0000 0000_H
2. Enable all clocks of all sub-modules VI_ICCL = FFFF FFFF_H
3. Trigger the asynchronous system reset of the CIF
Typically the asynchronous reset line is connected to a central module of your SOC, handling resets for all components. Usually there is one bit reserved for this purpose. There may be some delay required in the software before the following access to the CIF (time until the central module clears reset, time until the reset signal is synchronized to the sensor clock domain of the CIF).

ISP input acquisition

1. Switch to ITU.BT601 compatible mode (ISP still disabled yet) ISP_CTRL = 0000 0004_H
2. Set the input acquisition properties according to what is delivered by the sensor: sampling at falling edge of pixel clock; h- and v-sync high active; sample sequence Y-Cb-Y-Cr; don't care about odd or even fields; 8-bit data interface.
ISP_ACQ_PROP = 0000 0300_H
3. Set the input acquisition window according to what is delivered by the sensor.(here, we assume no delay between the sync signals and the active pixel data, that's why the offset values can remain zero. Horizontal size is twice the image width, because for every 2 incoming pixels, there are also 2 chrominance values):
ISP_ACQ_H_SIZE = 0000 0500_H(2*640 = 1280)
ISP_ACQ_H_OFS = 0
ISP_ACQ_V_SIZE = 0000 01E0_H(480)
ISP_ACQ_V_OFS = 0

ISP output formatter

1. We want to output the complete incoming frame from the sensor (no cropping) so the whole sensor resolution need to be programmed in the output window
ISP_OUT_H_SIZE = 0000 0280_H(640)
ISP_OUT_V_SIZE = 0000 01E0_H(480)
ISP_OUT_H_OFFS = 0 ISP_OUT_V_OFFS = 0

Data Path

1. Select parallel interface and main data path, enable output of main data path
 $VI_DPCL = 0000\ 0001_H$ $MI_CTRL = 0000\ 0001_H$

Memory Interface

1. Set Y, Cb and Cr buffer position and size (32bit aligned). Update shadow registers with next configuration update:

```

MI_MP_Y_BASE_AD_INIT = <Y base address>
MI_MP_Y_SIZE_INIT = <sizeY>
MI_MP_CB_BASE_AD_INIT = <Cb base address>
MI_MP_CB_SIZE_INIT = <sizeC>
MI_MP_CR_BASE_AD_INIT = <Cr base address>
MI_MP_CR_SIZE_INIT = <sizeC>

```

Set to main picture data mode (do not swap bytes, do not mirror) and update shadow register immediately. (This also transfers the base address and the size to their respective shadow registers).

```

MI_CTRL = 0030\ 0001_H      MI_INIT = 0000\ 0010_H

```

Interrupt Processing

1. Setup interrupt service routines, clear and enable interrupts if needed, e.g. frame end interrupt of memory interface:

```

MI_ICR = 0000\ 0001_H  MI_IMSC = 0000\ 0001_H

```

Start the CIF

1. Enable input acquisition and output formatter (to start the output formatter there is no configuration update necessary). As soon as the CIF is started the lower two bits of ISP_FLAGS_SHD are set.

$ISP_CTRL = ISP_CTRL | 00000011_H$ With the initialization sequence above, the CIF is set to sample all incoming frames and presents them at the output. If this is not the desired use case and e.g. a single frame is to be captured instead, this can be configured using the input acquisition register $ISP_ACQ_NR_FRAM$. If this register is nonzero, only the given number of frames is being captured after the CIF has been started.

36.1.5.8 Use Case Description

The CIF is designed to support for example the following mobile phone use cases:

- Data transfer from external devices into system memory
- Viewfinder mode for still image capture
- Still image capture
- Video encoding

The above mentioned use cases are described in detail in the following chapters. Furthermore information about power management and configuration accesses are also given in this section.

Data Transfer

This use case transfers byte data, like JPEG compressed data from a camera sensor with integrated JPEG encoder, into system memory (main Y buffer). Further processing must not be done as these data are no plain image or video data. The **Figure 36-34** shows the blocks active during data transfer. The inactive blocks are bypassed or disabled.

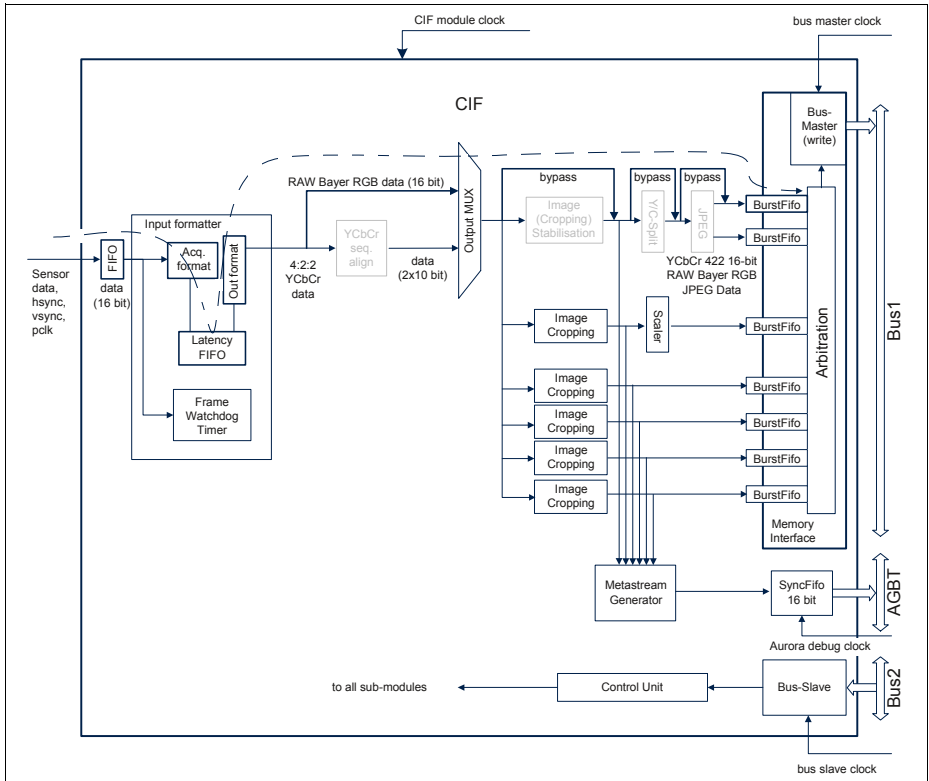


Figure 36-34 Active Blocks for Use Case Data Transfer

A transfer indicator and a byte valid signal (refer to **Figure 36-28**) are used to identify transfer start and stop as well as to identify valid data bytes. The ISP has to be set to data mode (mode field in ISP_CTRL), all other functions (e.g. offset and size registers) are not used. The data path must be switched to 8 bit data/RAW data mode (VI_DPCL).

At the memory interface the Y buffer must be defined (MI_MP_Y_BASE_ADR, MI_MP_Y_SIZE) and the RAW data port must be enabled (raw_data_en in MI_CTRL).

After transfer being indicated by the “frame end” interrupt of the memory interface the number of transferred bytes can be read from MI_BYTE_CNT.

Viewfinder Mode

This use case is to be used for displaying a video stream at an attached LCD. The CIF is responsible for capturing the video data at the sensor interface and transferring a continuous image data stream to the system memory.

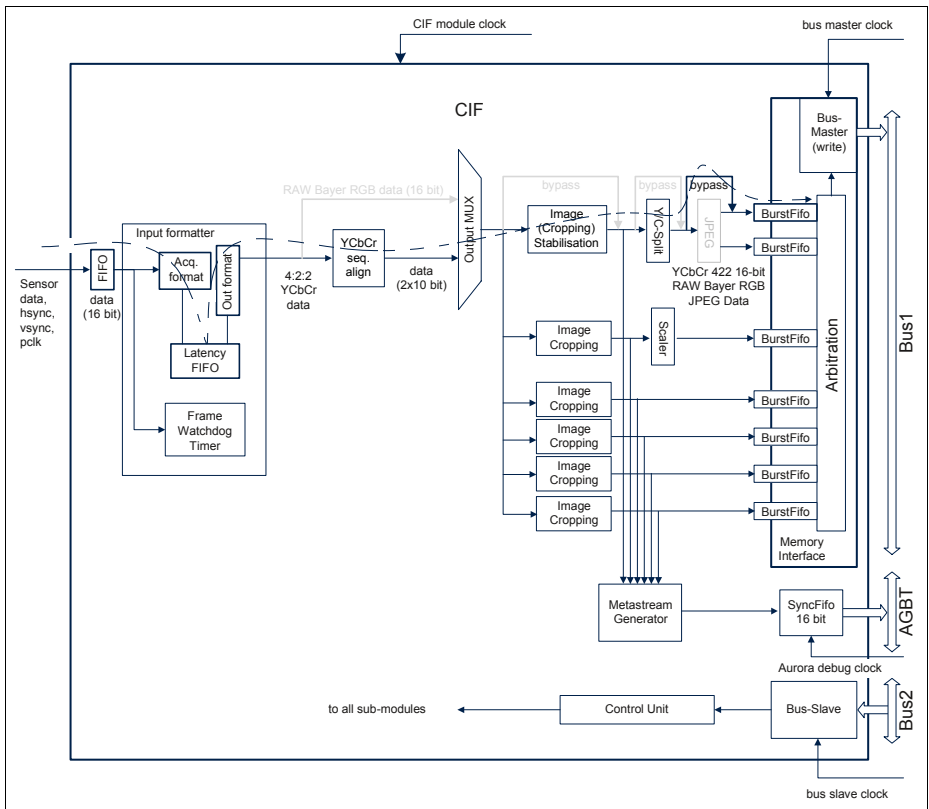


Figure 36-35 Active Blocks for Use Case Viewfinder

Image data are captured at the ISP. The mode depends on the camera sensor being used and can be YCbCr 4:2:2 with embedded timing control signals or external synchronization signals (selected by the `isp_mode` field in `ISP_CTRL`).

Still Image Capture

This use case is meant for capturing a still image.

The use case viewfinder always precedes the capturing use case.

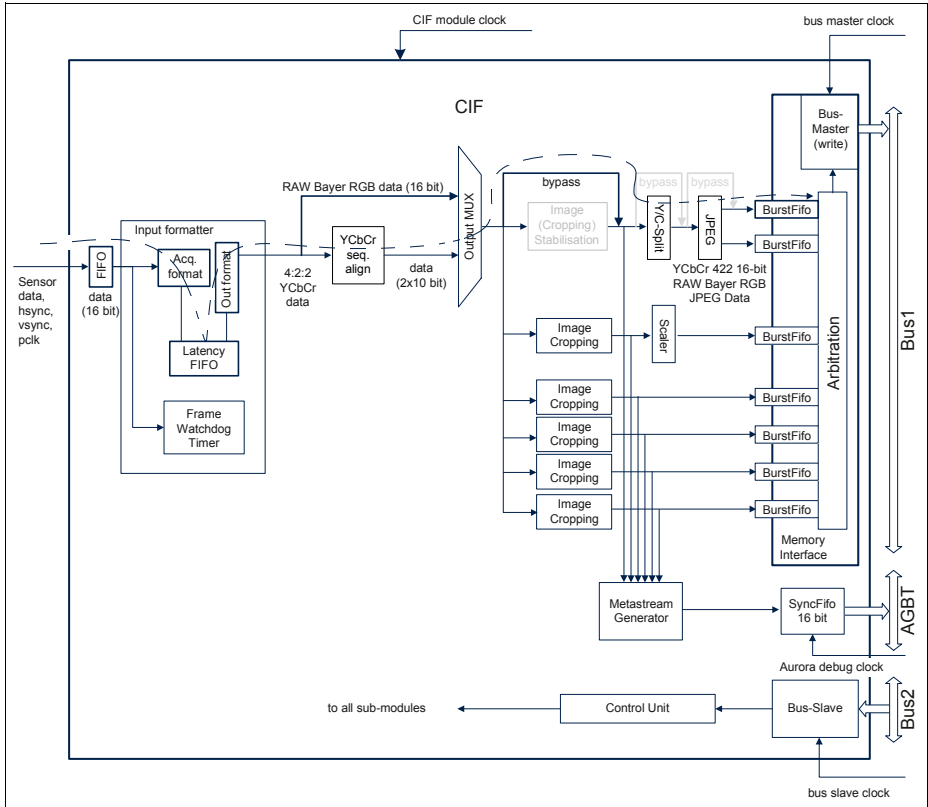


Figure 36-36 Active Blocks for Use Case Still Image Capture

In this use case the ISP samples the image and color processing can be done. The data path must be set to main 8bit luminance and chrominance processing format.

36.1.5.9 Power Management

There are two main power modes for the CIF: running and switched-off. In running mode the master clock provided at the Bus interface is passed to the CIF blocks. In switched-off mode the master clock is disabled (gated) in the CIF control unit. So only register

accesses are possible, but no processing. Controlling the main power modes is done using the VI_CCL configuration register.

In running mode a further method exists to minimize power consumption of the CIF. Separate clock gating exists for all blocks inside the CIF (except the Y/C-splitter). This is controlled by the VI_ICCL configuration register. For each module that is not needed for the actual processing (bypassed or disabled) the module clock can be disabled by clock gating as shown in Figure **“Clock Gating Scheme for the CIF” on Page 36-266**.

Clock gating for static configuration registers is handled automatically. Their clocks are enabled during read or write accesses via the Bus slave interface only, otherwise they are switched off.

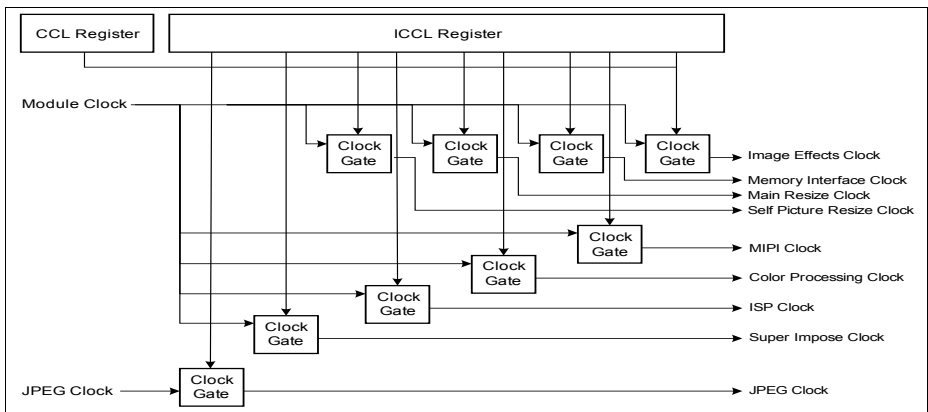


Figure 36-37 Clock Gating Scheme for the CIF

36.1.5.10 Basics on Configuration Access

Configuration accesses are done by using the Bus slave interface. Single beat transactions with a data width of 32 bits are supported.

Each CIF sub-module contains its own local configuration registers. For optimized power consumption these registers are clocked during read/write data transfers only. This clock gating is handled automatically by the control unit and is fully transparent to the application.

36.2 Extended Memory (EMEM)

The Extended Memory (EMEM) contains RAM blocks (Tiles) which can be used as global memory. It has interfaces to the LMU and the BBB. The EMEM architecture is similar to the EMEM of the Emulation Devices (ED).

Applications:

- Data and program code storage
- FFT data buffering
- Image buffering

Features:

- 512 Kbyte RAM in total
- Concurrent access from LMU and BBB to different tiles possible
- BBB burst transfers supported
- ECC with SECCDED (Single Error Correction, Double Error Detection)

36.2.1 Block Diagram

The Extended Memory consists of a regular memory array (ADM - Application Data Memory) build out of 64 Kbyte RAM Tiles. Additionally on certain products of the family the optional extension memories XAM is offered.

The Extended Memory provides two internal interfaces:

- LMU Interface
- BBB Interface with master and slave interface

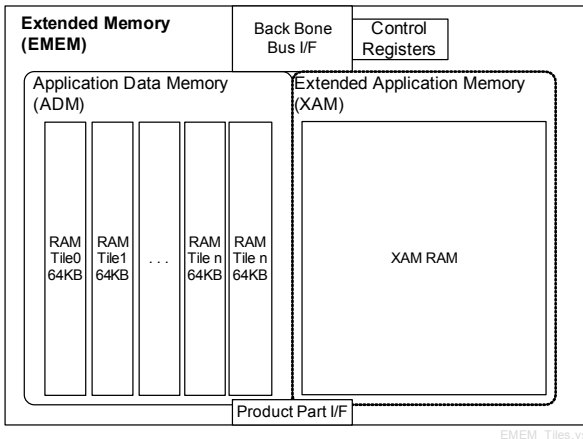
The two interfaces allow a concurrent access to two different Tiles.

36.2.1.1 Extended Application Memory (XAM)

The XAM feature is a family concept for products, which require more than 1 MB Application RAM. The first EMEM RAM block (ADM) is partitioned into 16 Tiles. The following XAM EMEM block is without partitioning. A wrapper routes the accesses to ADM or XAM as a single continuous address range. The availability of XAM for different members of the product family is listed in [Table 36-4](#).

36.2.1.2 Family Overview

[Figure 36-38](#) shows an EMEM configuration example and [Table 36-4](#) the Extension Memory configurations for the different products of the family.


Figure 36-38 EMEM Configuration Example with ADM, XAM Tiles
Table 36-4 Extension Memory availability for the Product Family

Product	Total RAM Size	ADM Size	XAM Size	XAM Addr. LMU/BBB
TC29x	2048 KB	1 MB	1 MB	xx10 0000 ¹⁾
TC27x	1024 KB	1 MB	-	-
TC26x	512 KB	512 KB	-	-
TC24x	256 KB	256 KB	-	-
TC23x	512 KB	512 KB	-	-

1) More information can be found in [Table 36-6](#).

36.2.2 Operational Overview

The Extended Memory as a whole can be in Standby Locked Mode or in Enabled Mode. Immediately after Power On Reset, the entire Extended Memory is in Unused Mode, thus the Tiles are not yet reserved for application mode. At any time, each Tile can be individually assigned to Application or again Unused Memory by setting its mode.

If a Tile has to be configured to Application Mode to activate the Extended Memory for the application.

This two step approach with selecting the global assignment for the Extended Memory of a Tile first and then operating it supports a safe operation.

Note: Before using the EMEM the boundary to this memory must be enabled by de-asserting the OSCU control bit OSTATE.EECDIS. Failing to do so will result in a hang-up of the CPU/DMA on the first read access to the EMEM.

Tile in Unused Mode

After Power On Reset, the Tiles are still in Unused Mode and are not yet reserved. They cannot be accessed before being assigned to Extended Memory by software. Later on, each Tile can be individually returned to Unused Memory space.

Tile in Application Mode

A RAM Tile in Application Mode can be accessed from the LMU or the BBB. The arbitration for concurrent accesses is done within the Extended Memory by delaying the BBB access.

Simultaneous Accesses

Up to two different Tiles can be accessed in parallel via the two interfaces, if these Tiles are configured in Application mode.

36.2.3 Extended Memory

By writing to Register **TILECONFIG**, unused Tiles can be assigned to Extended Memory. In Application Mode, write and read accesses can be performed by the LMU or BBB interfaces in arbitrary sequence. The arbitration is done internally with higher priority for the LMU interface.

36.2.4 Access Error Generation

If a Tile is accessed over an interface which is not supported in the current mode of the Tile, an error is signalled to the access initiator.

Any access attempt to RAM or a register during Standby Locked Mode is answered with an access error.

Table 36-5 Error Signaling for RAM Accesses

	LMU I/F	BBB I/F
Application Mode	Ok ¹⁾	Ok ¹⁾

1) If an ECC error is detected, an access error is signalled to the corresponding interface. If the width of a write access is smaller than the addressed memory word, an internal read-modify-write access is performed which potentially results in an ECC error for the read part. An ECC error is also generated, if a RAM location was not initialized before!

In Locked Mode, the registers **CLC**, **ID**, and **SBRCTR** are accessible via the LMU through the BBB master interface, access to other registers raises an error.

36.2.5 Tiles and RAM Address Ranges

The Extended Memory (ADM RAM area) can be accessed from two interfaces:

- LMU Interface
- BBB Interface

Accesses to the Extended Memory via the LMU are done by one of the CPUs or other SRI bus masters. To cater for cached/non-cached access the Extended Memory is mapped to two segments by the LMU.

The addressing of ADM and XAM depends on the interface. LMU and BBB have the same addressing view, just with a different base address.

The LMU/BBB addressing is based on the 1 MB ADM address area, followed by XAM always starting at the 1 MB address.

Assignment of Tiles to Trace Memory usually starts with Index 0 ([Table 36-6](#)).

Table 36-6 EMEM Tiles and Address Ranges

Tile	LMU ¹⁾ TC26x	BBB	SBDAP Address
t; (ADM) t=0...7	sF0t0000 _H sF0tFFFF _H	AF0t0000 _H AF0tFFFF _H	000t0000 _H 000tFFFF _H

1) s = B_H: non-cached access by CPU, s = 9_H: cached access by CPU

For register accesses see [Table 36-7](#).

[Figure 36-39](#) shows TC29x and TC26x as an example.

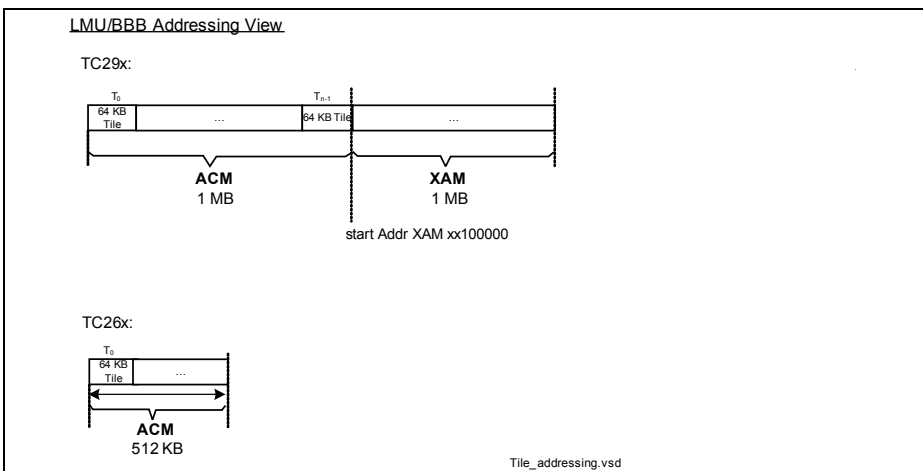


Figure 36-39 TCM and AXM interface addressing view.

More details about ADM and XAM addressing can be found in [Table 36-6](#).

36.2.6 EMEM Initialization

After Power-on Reset the EMEM is locked and requires an unlock sequence similar to the EMEM of Emulation Device to get the memory initialized.

Unlocking the EMEM

To unlock EMEM, the following sequence is written to the [SBRCTR](#) register.

1. 0000 0002_H,
2. 0000 0006_H,
3. 0000 000E_H

Writing any other value to [SBRCTR](#) before the complete sequence was written will reset the state machine, i.e. the sequence must be started again from the beginning. Writing to other registers however does not influence the sequence checker state machine.

The current state is signalled with bit [SBRCTR.STBLOCK](#).

36.2.7 Usage Constraints

In this section assumptions are listed for using the EMEM module.

36.2.7.1 Clock Constraints

The clock for the BBB must not be faster than the EMEM clock. For the TC26x B-Step the LMU clock is equal to SRI clock.

36.2.7.2 BBB Interface

The BBB slave interface supports burst transfers, e.g. for image buffering.

36.2.8 Power Saving

In order to minimise power consumption, the clocks of inactive Tiles can be gated.

If clock gating is enabled ([SBRCTR.ACGEN](#)), all Tiles in Unused Mode are no longer clocked.

Tiles in Application are not gated directly. For this purpose they can first be set to Unused Mode.

The current clock gating status is individually indicated for each Tile ([SBRCTR.ACGSTi](#)).

36.2.9 EMEM Configuration

For power saving reasons, each tile is initially in unused mode. The register bits **TILECONFIG** allow to allocate the specific tiles as application memory. The configuration state of each Tile is indicated in register **TILESTATE**.

36.2.10 EMEM Interfaces

As each Tile is arbitrated individually, transactions from the different interfaces into different Tiles are handled concurrently, i.e. up to two.

According to the word-lengths of the addressed RAM blocks, measures are taken for optimizing RAM accesses. Internal read-modify-write transactions are only required (due to ECC) if the word-length of data for the physical write access is smaller than the word-length of the memory.

36.2.10.1 LMU Interface

Read and write transactions are initiated by the LMU. Burst and read-modify-write transactions are not supported, but are split into basic and concatenated transactions.

Wide Read Transactions

Read accesses wider than the read-data word-length at the LMU are transformed into multi-beat transactions in subsequent cycles, taking into account wrapping around for unaligned addresses.

Concatenated read accesses of small words are not expected due to the wide-read access feature. If concatenated read accesses should occur anyhow, the EMEM processes them just like sequences of single read accesses without waiting for the master request to become inactive in between.

Concatenated Write Transactions

For write-widths smaller than the word-width of the memory, generally a complete memory word has to be synthesized before physically writing to memory, due to the required ECC which refers to complete memory words only. Whenever a write access is received from the LMU, a speculative read is immediately started and a ready signal returned to the LMU. In the next cycle, the read data is updated with the previous write data and written back, if the first write access is single. If instead another concatenated write transaction is received, the just obtained read data is not used, but the two subsequent write data portions are combined and written to memory. Thus, for a concatenated write, only the first of two consecutive write accesses causes a speculative internal read, because the protocol does not yet allow to decide for a first write, whether a second concatenated one is about to follow.

36.2.10.2 BBB Slave Interface

Transactions from the BBB to EMEM memory are initiated by any BBB master. In order to avoid internal read-modify-write accesses to EMEM-RAM, BBB write data received by the EMEM slave interface in burst operations is accumulated before being written into the memory. The EMEM control decides from the FPI opcode and the given memory word size whether a combination is possible or an internal read is required. These optimizations include unaligned burst transfers.

36.2.11 Feed-Through Transactions

The EEC must also be accessed from the product chip part. For this purpose, the EMEM feeds through transactions received from the LMU to the BBB Master Interface, if none of the EMEM's RAMs is addressed. Thus LMU and EMEM jointly act as bus-bridge between the SRI bus and the Backbone bus.

When acting as BBB Bus Master the EMEM tag and Supervisor Mode (SVM) is used.

36.2.11.1 EMEM Register Accesses

The SFRs of the EMEM are accessed via the BPI part of the BBB Slave Interface by any component connected to the BBB through a master interface, such as IOC32, and also the EMEM itself.

A register access from the product side is performed via LMU, the BBB Master Interface to the BBB, and from there via the BBB Slave Interface.

36.2.12 Address Map

All EMEM registers are prefixed "EMEM_" in the register map of the device.

EMEM occupies an address window of 256 bytes. The relative register addresses in the register descriptions below are given as offset addresses to the base address of the window.

The base addresses for register accesses are identical for LMU and BBB. Internally in the EMEM, a window of 2 MB, `addr[21:0]` is available for addressing registers via BBB.

Note: This window is mapped to another segment when accessed via the SRI. Please, check production device address mapping to find the correct offset.

Access rights within the address range of EMEM:

- Read access to defined register addresses: U, SV
- Write access to defined register addresses: U, SV (no BE even if read-only register)
- Read access to empty addresses: No error (returns 0)
- Write access to empty addresses: BE
- Read or write when the kernel clock is turned off: BE (unless **CLC**)

Table 36-7 Registers Address Space

Module	Base Address	End Address	Note
EMEM	F90E 6000 _H	F90E 60FF _H	CPU address

Table 36-8 Registers Overview

Register Short Name	Register Long Name	Offset Address	Page Number
CLC	Clock Control Register	00 _H	Page 36-278
ID	Module Identification Register	08 _H	Page 36-279
TILECONFIG	Tile Configuration Register	20 _H	Page 36-275
TILESTATE	Tile Status Register	2C _H	Page 36-276
SBRCTR	Standby RAM Control Register	34 _H	Page 36-276

36.2.13 EMEM Register Description

As detailed in the OCDS chapter, several reset domains exist on TC26x B-Step. All registers of the EMEM module belong to the Power On Reset domain and are reset by PORST. The **CLC** register additionally belongs to the EEC Reset domain.

TILECONFIG

Tile Configuration Register (20_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r		r		r		r		r		r		r		r	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T7	T6	T5	T4	T3	T2	T1	T0								
w		w		w		w		w		w		w		w	

Field	Bits	Type	Description
Tx (x = 0 - 7)	[2*x+1: 2*x]	w	Tile x Allocation 00 _B Allocate tile 01 _B Write has not effect. 10 _B Reserved 11 _B Unused <i>Note: The Tile allocation after reset requires "00" to be written to allocate the tile!</i>
0	[31:16]	r	Reserved returns 0 if read; should be written with 0.

TILESTATE
Tile Status Register
(2C_H)
Reset Value: 0000 FFFF_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TILE7	TILE6	TILE5	TILE4	TILE3	TILE2	TILE1	TILE0								
rh	rh	rh	rh	rh	rh	rh	rh								

Field	Bits	Type	Description
TILE_x (x = 0 - 7)	[2*x+1: 2*x]	rh	Usage of Tile x 00 _B Allocated 01 _B Reserved 10 _B Reserved 11 _B Unused Mode
0	[31:16]	r	Reserved returns 0 if read; should be written with 0.

SBRCTR
Standby RAM Control Register
(34_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ACG ST7	ACG ST6	ACG ST5	ACG ST4	ACG ST3	ACG ST2	ACG ST1	ACG ST0
r	r	r	r	r	r	r	r	rh	rh	rh	rh	rh	rh	rh	rh
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	ACG EN		0		0		STBSLK				STBULK			STB LOC K
r	r	rw		r		r		w				w			rh

Field	Bits	Type	Description
STBLOCK	0	rh	Lock Flag Shows the current lock state of the Extended Memory. 0 _B Extended memory is locked 1 _B Extended memory is unlocked
STBULK	[3:1]	w	Unlock Lock Flag In order to unlock the Extended Memory in Locked Mode in three consecutive ¹⁾ cycles the following pattern have to be written into this bit field: <ul style="list-style-type: none"> • 001_B • 011_B • 111_B At the same time 0 has to be written always into bit field STBSLK. If any of STBSLK is set when writing a non-zero pattern to STBULK, this is treated as invalid pattern and the Extended Memory in Locked Mode will not be unlocked. Reading this bit field always will return 0s
STBSLK	[7:4]	w	Set Lock Flag In order to lock the Extended Memory in operating mode 1001 _B has to be written into this bit field. At the same time 0 has to be written into the bit field STBULK. If any of bits STBULK is set when writing 1001 _B to STBSLK, this is treated as invalid pattern and the Lock Flag is not set. Reading this bit field always will return 0s
ACGEN	12	rw	Automatic Clock Gating Enabling 0 _B Disabled 1 _B Enabled
ACGSTx (x = 0 - 7)	x + 16	rh	Automatic Clock Gating Status of Tile x 0 _B Clock off 1 _B Clock on <i>Note: These bits are cleared automatically when entering Locked Mode.</i>
0	[11:9], 8, 15, [14:13], [31:24]	r	Reserved returns 0 if read; should be written with 0.

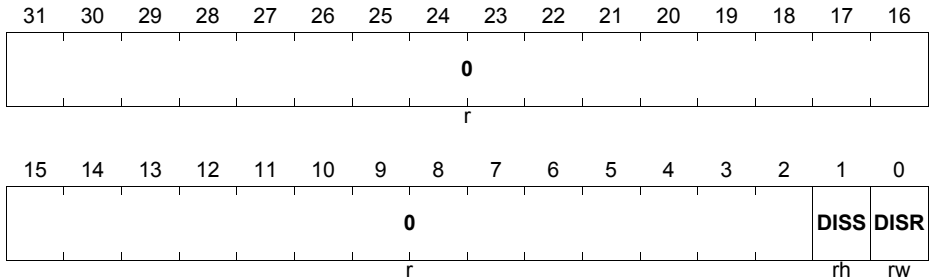
1) Intermediate read or write cycles not accessing **SBRCTR** are allowed.

CLC

Clock Control Register

(00_H)

Reset Value: 0000 0003_H



Field	Bits	Type	Description
DISR	0	rw	Disable Request Bit 0 _B Module disable is not requested 1 _B Module disable is requested
DISS	1	rh	Disable Status Bit 0 _B Module enabled 1 _B Module disabled
0	[31:2]	r	Reserved returns 0 if read; should be written with 0

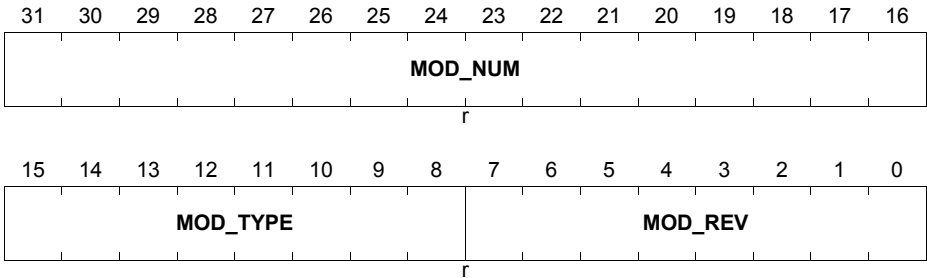
The clock control register allows to switch the memory subsystem on or off. After Power On Reset the memory subsystem is disabled. The memory subsystem can be enabled by clearing bit DISR. It is ensured that the **CLC** is still writable when the clock is switched off.

ID

Module Identification Register

(08_H)

Reset Value: 00E0 C004_H



Field	Bits	Type	Description
MOD_REV	[7:0]	r	Module Revision Number
MOD_TYPE	[15:8]	r	Module Type C0 _H for 32 bit peripheral
MOD_NUM	[31:16]	r	Module Number For module identification

36.3 FFT Engine (FFT)

This chapter describes the FFT Engine, an algorithm specific accelerator. The FFT Engine is not available in all devices and only those with the ADAS feature set.

The FFT Engine utilises the SRI connectivity of the LMU. The FFT Engine is clocked at the same frequency as the SRI and all cycle counts are in terms of SRI clocks.

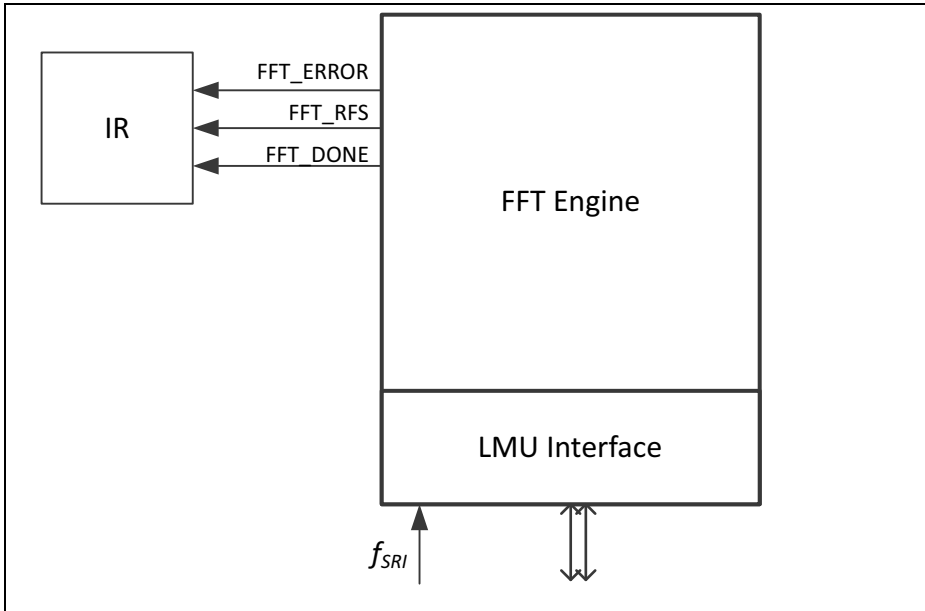


Figure 36-40 FFT Engine Integration into the LMU

36.3.1 Overview

The FFT Engine calculates forward and inverse FFTs of lengths 8 to 1024 (powers of 2 only). It operates on real and complex data which can have co-efficients of 16 or 32-bits in length.

The actual forward and inverse transforms calculated by this core are given by

$$X[k] = \frac{1}{N} \sum_{n=0}^{N-1} x[n] \exp(-j2\pi kn/N) \quad \text{Forward}$$

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \exp(j2\pi kn/N) \quad \text{Inverse}$$

In usual DSP literature the forward transform is not divided by N, but this is performed in hardware for consistency between transforms and to implement a transform that cannot overflow.

36.3.2 Feature List

General features:

- Per transform selection of FFT or inverse FFT (IFFT).
- Per transform selection of FFT/IFFT size from 8, 16, 32, 64, 128, 256, 512 or 1024 points.
- Per transform selection of real or complex input operands.
- Per transform selection of input operands with 16-bit or 32-bit precision.
- Per transform selection of output results with 16-bit or 32-bit precision.
- Internal calculation to 32-bits of precision.

In addition to standard FFT calculations, the FFT Engine also supports a window function to be applied to input operands. The window function utilises a window RAM to hold coefficients which allows any real symmetrical function to be used. Use of this features is described in [Section 36.3.7](#). Another important performance enhancement is the use of double buffering to support the overlap of loading and unloading the FFT Engine while computation continues

Input operands for FFT or inverse FFT calculations which are loaded with 16-bits of precision are converted to 32-bits prior to storage inside the FFT engine. Calculations are carried out to 32-bits of precision using block floating point and adaptive scaling to ensure there is no loss of dynamic range during the intermediate calculations. Results are adjusted to natural scaling for software convenience before being read out. Output results which are unloaded with 16-bits of precision are converted from the 32-bit internal result by rounding (to nearest even) with saturation as they are transferred out.

36.3.3 FFT Engine Performance

After operands are loaded into the FFT Engine, the number of cycles the engine takes to complete a computation is a function of the length of the transform it has been programmed to execute. If N is the length of the transform, then the number of clock cycles to complete is:

$$(N+15) * \text{ceil}(\log_4(N))$$

The cycle count till availability of results for the supported FFT/IFFT lengths is show in [Table 36-9](#).

Table 36-9 FFT Engine Execution Times

Transform Length	Cycles	Transform Length	Cycles
8	46	128	572
16	62	256	1084
32	141	512	2635
64	237	1024	5195

36.3.4 System Performance

The complete steps required for the execution of an FFT/IFFT are:

- writing to the FFT_CSR to configure the required operation (input format, length, window operation, output format, start) with a single write
- transferring the required number of operands into the FFT Engine
- waiting for the engine to complete (or being interrupted)
- unloading the results from the FFT Engine

Total execution time for an FFT/IFFT calculation must take into consideration the time for storing of operands into the FFT Engine, and reading of the results out as well as the calculation time shown in the table FFT Engine Execution Times.

36.3.4.1 Data Movement Optimisation

The loading and unloading cycles will be a significant part of the total execution time required and is one of the major optimisation opportunities. Using the minimum operand size for input or output (16-bit versus 32-bit) can halve the amount of time in loading and unloading.

FFT transforms based on real operands only, could be calculated by loading complex values with imaginary components of zero along with real component values. However the FFT Engine supports optimising out this unnecessary transfer, and setting FFT_CSR.IN_FMT appropriately for real operands will halve the loading time. Both these transform options will significantly reduce the end to end cycle count for an operation, despite the fact that they have no effect on the transform calculation time.

Programmed loading and unloading by TriCore CPUs can transfer 64-bits at a time. For smaller transform lengths this may be most efficient. For longer transforms, the optimum bandwidth to load and unload the FFT Engine is obtained using the DMA to transfer bursts of four 64-bit words at a time.

36.3.4.2 Double Buffering Optimisation

As well as the opportunity to optimise the data movement time, the FFT Engine also provides the mechanisms to overlap FFT/IFFT execution with loading and unloading results. The FFT Engine is provided with two buffers each capable of supporting a maximum length FFT/IFFT transform. These two buffers are operated in a ping-pong arrangement and allow the overlap of transfer in and out of the FFT Engine, with the calculation of a transform with an earlier loaded set of operands. Thus, the throughput of FFT/IFFT operations is higher than determined by the total latency (loading, execution, unloading) of an individual transform.

36.3.5 Address Spaces and Access Protection

The FFT Engine responds to three address spaces in the system address map:

- a register space located from F870_0C00h to F870_0CFFh,
- a data space located at BE00_0000h to BE07_FFFFh,
- a window coefficient space located at BE10_0000h to BE17_FFFFh.

All three address spaces within the FFT Engine are protected by the access enable registers of the LMU which protects the resources against writes from unauthorised SoC masters.

36.3.5.1 Register Address Space

The FFT register space consists of 32-bit registers that only respond correctly to 32-bit read and write operations.

- any read from register space which is an 8-, 16- or 64-bit operation will result in an undefined value,
- any read from register space which is a burst transfer will result in a transaction error,
- any write to register space which is a burst transfer or an 8-, 16- or 64-bit operation will result in a transaction error,
- any atomic operation to register space will result in a transaction error.

36.3.5.2 Data Space

The data space only supports 64-bit and burst-4 transfers. Accesses that are 8-, 16- or 32-bit or burst-2 transfers to the data space will have undefined results.

The data space does not have standard memory characteristics. Instead this region operates as access to two separate FIFOs. Write accesses (regardless of addresses used) will place operands in order into one of the ping-pong buffers; for example for a 512 length FFT, the first data transferred will be 'sample 0', and the 512th sample transferred will be 'sample 511' regardless of the addresses used (provided they are all

within the FFT Engine data space). There is no ability to direct accesses to either of the ping-pong buffers, this is purely controlled by the FFT Engine.

Read accesses to the data space will return results out of one of the FIFOs in the same manner. The first read of a completed transform will receive 'result 0', and the Nth read will receive 'result N-1', again the addresses used will not influence the result element returned.

Bit FFTPFDIS should be cleared (0) in register LMU_MEMCON (see Local Memory Unit chapter) before accessing the data space.

Intermixing single transfers of 64-bits or burst-4 transfers during the same loading phase, or same unloading phase will have undefined results.

36.3.5.3 Window Coefficient Space

The window coefficient space only supports 64-bit and burst-4 transfers. Accesses that are 8-, 16- or 32-bit or burst-2 transfers to the window space will have undefined results. Within the window coefficient address space is a RAM holding the window coefficients - the window RAM.

The window RAM occupies 1-KByte (with corresponding aliases) of the window address space. Read and writes to this 1-KByte RAM will have standard memory characteristics only when validly accessed.

Accesses to the window RAM are only well behaved (reads to an address will return the same data last written) when

- the FFT Engine is idle,
- the FFT Engine does not have any results to be unloaded and
- bit FFTPFDIS should be set (1) in register LMU_MEMCON (see Local Memory Unit chapter).

For this reason writing to the window RAM should only be performed when the three conditions above have been met.

Note: The entire window RAM must be initialised before the FFT Engine can run a transform. This requirement is regardless of whether the window function is enabled or not for the transform.

36.3.6 Data Value Formats

The data formats supported by the FFT Engine for operands are identical to the corresponding TriCore architecture fractional formats for ease of manipulation. While, the window coefficient format which is manipulated less frequently uses a different format to slightly enhance precision.

36.3.6.1 Time and Frequency Sample Format

Data with 16-bit precision corresponds to TriCore signed fractional data with 16-bits (also known as 1.15 or Q15 format). Data with 32-bit precision corresponds to TriCore signed fractional data with 32-bits (also known as 1.31 or Q31 format).

Data values in this format have a single high-order sign bit, where 0 represents positive (+) and 1 represents negative (-), followed by an implied binary point and fraction. Their values are therefore in the range $[-1, 1)$.

Complex data corresponds to an array of two elements in little-endian order, where the first element (index 0 in 'C' notation) is the real coefficient, and the second element (index 1 in 'C' notation) is the imaginary coefficient.

36.3.6.2 Window Coefficient Format

A valid window function will have coefficients in the range $[0, 1]$. However, strictly only the central window coefficient requires to be 1.0 and the remainder are in the range $[0, 1)$. If the 1.15 or Q15 format was used for window coefficients, then as all the coefficients are positive, the MSBs of co-efficients stored in the window RAM (the central coefficient is not loaded), would be redundant.

To take advantage of this fact, the FFT Engine calculate with unsigned window coefficients and the stored values to the window RAM are the 16 fractional bits of a nominal 1.16 format.

36.3.7 Window Usage

The window RAM allows the user to load a window function of their choice and apply it (or not) to subsequent transforms. The parameters in the window RAM are only changed when the window function is required to be changed, they otherwise persist between transform operations (i.e. have standard memory characteristics).

The window RAM only contains half the maximum length transform entries (see [Section 36.3.6.2](#) for the format) and exploits symmetry to generate a complete window for that length of transform. If a window function is to be used for any length transform, it must be fully loaded and the FFT Engine will select the appropriate coefficients.

The expected usage is that a window function will be selected for a good match to the system and application properties, loaded at system initialisation, and then either applied or not (using FFT_CSR.WIN_BYN clear or set respectively) to subsequent FFT operations. While the window function can be changed during system operation, any change must respect the requirements listed in [Section 36.3.5.3](#). Thus, any change will incur some latency till the next transform operation is possible; either with or without the new window function applied.

Note: As specified in [Section 36.3.5.3](#) the window RAM must be initialised before any FFT Engine transform operation.

36.3.7.1 Window Parameters

If the window function is to be used, the coefficients for a maximum length transform must be computed. First generate a window of odd length and with a central maximum value of 1.0. For example if using MATLAB then this command

```
w = hamming(1024, 'periodic');
```

or

```
w = hamming(1025, 'symmetric');
```

will return the first 1024 points of a 1025 point periodic window. Next select the first 512 values of w and scale them up and round to an unsigned 16-bit integer

```
wi = round(65536*w(1:512));
```

These are the values to load into the window RAM. Note that this procedure is generic for any window, always start with a length 1025 symmetric or 1024 period variant. A common window in radar is Blackman-Harris, which would be generated in MATLAB with

```
w = blackmanharris(1025);  
wi = round(65536*w(1:512));
```

Note that the 1.0 central coefficient is internally generated by the hardware so it is essential that the window is correctly scaled to 1.0.

For FFT transform sizes less than the maximum the hardware will index the full window table to generate the correct coefficients for a smaller FFT size.

Although the hardware will not stop you applying a window in the frequency domain it may not be desirable.

36.3.8 Interrupt and DMA Operation

As well as inspecting the CSR register to determine when FFT/IFFT operations may be commenced, the FFT Engine allows interrupt driven data movement. There are two SRC registers in the IR connected to triggers from the FFT Engine. These are SRC_FFTRFS and SRC_FFTDONE. As is standard, each of these SRCs can select either a CPU or DMA channel to service the event.

A trigger will be sent to SRC_FFTRFS whenever FFT_CSR.RFS transitions from 0 to 1, this alerts the system to the fact the FFT_ENGINE can be loaded with another transform. Since there are two (ping-pong) buffers, this event means the FFT Engine may still be busy computing a previously loaded transform.

A trigger will be sent to SRC_FFTDONE whenever a transform calculation has completed and the unload operation can be performed.

A possible operational sequence utilising both of these trigger conditions is described below

1. A CPU reads the FFT_CSR register and looks for RFS bit to be set. Software then writes to the FFT_CSR register with START set to 1, and the desired values for IFFT, LENGTH, WIN_BY, IN_FMT, OUT_FMT.
2. The same CPU programs DMA channel(s) to send data to the FFT Engine, and configures the service request control (SRC) registers in the interrupt router SRC_FFTRFS and SRC_FFTDONE as required.
At this stage the CPU could return to some other task (perhaps preparing operands or examining results).
3. A CPU receiving the interrupt corresponding to SRC_FFTRFS, can write the next START command and initiate the DMA to transfer the next transforms operands to the FFT Engine.
Again at this stage the CPU is then free to return to some other task.
4. An interrupt corresponding to SRC_FFTDONE can trigger an already configured DMA channel to read back data from the FFT Engine, which when it has unloaded the results can then trigger a CPU to inform it of the unload completion.
5. Loop back to 3

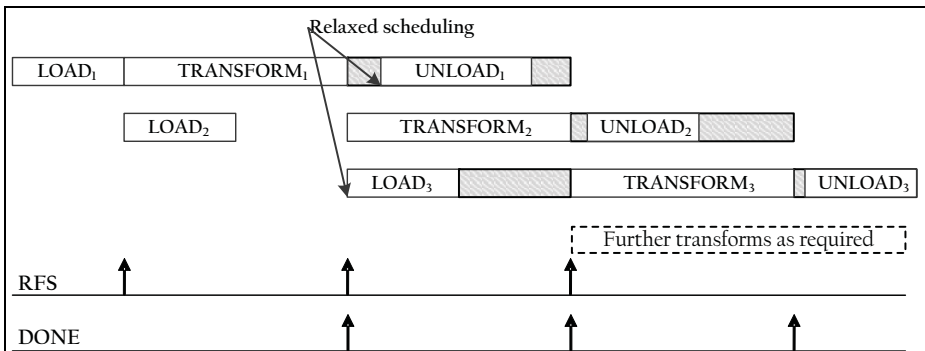


Figure 36-41 Three Consecutive Transforms

The double buffer architecture allows overlapping FFT execution with the DMA load/unload operation. This is illustrated in [Figure 36-41](#) where this example shows three transforms. The use of two memory buffers allows more relaxed scheduling of the DMA load/unload during overlapped transforms as shown by the unload operation being delayed until after the load operation for the third transform has commenced.

36.3.9 FFT Engine Registers

This section described the register interface to the FFT Engine. This address region should not be unintentionally read or written because this might cause side effects on FFT Engine state.

All registers are reset by the application reset apart from **FFT_ODA** and **FFT_OCS**. These registers are cleared by Debug Reset and by each System Reset when OCDS is disabled. They are not changed by System Reset when OCDS is enabled.

Where a register is defined as "ACCEN Enable Protected" (designated with "P" for writes), then the LMU ACCEN settings will be used.

Table 36-10 Registers Address Space

Module	Base Address	End Address	Note
FFT	F870 0C00 _H	F870 0CFF _H	-

Table 36-11 Registers Overview

Register Short Name	Register Long Name	Offset Address	Read Access	Write Access	Reset Value
FFT_CLC	FFT Clock Control Register	00 _H	U/SV	SV,E,P	0000 0002 _H
FFT_ID	FFT Identification Register	08 _H	U/SV	R	00E2 C001 _H
FFT_CSR	FFT Control and Status Register	40 _H	U/SV	U/SV,P	0010 0000 _H
FFT_HISTORY0	FFT History Register 0	60 _H	U/SV	R	0000 0000 _H
FFT_HISTORY1	FFT History Register 1	70 _H	U/SV	R	0000 0000 _H
FFT_ODA	FFT OCDS Debug Access Register	E4 _H	U/SV	SV,P	0000 0000 _H
FFT_OCS	FFT OCDS Control and Status Register	E8 _H	U/SV	SV,P	0000 0000 _H
FFT_KRSTCLR	FFT Kernel Reset Status Clear Register	EC _H	U/SV	SV,P	0000 0000 _H
FFT_KRST1	FFT Kernel Reset Register 1	F0 _H	U/SV	SV, P	0000 0000 _H
FFT_KRST0	FFT Kernel Reset Register 0	F4 _H	U/SV	SV, P	0000 0000 _H

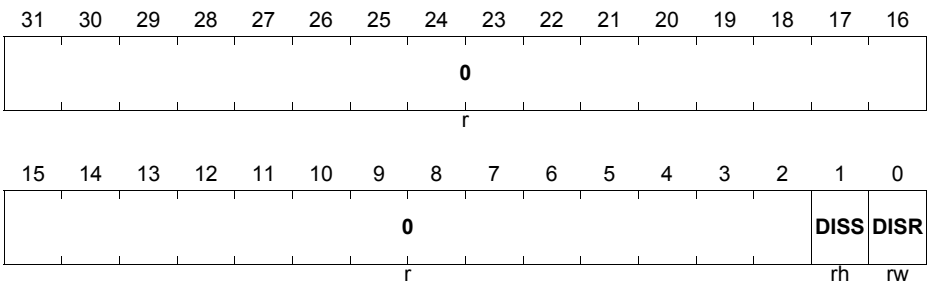
36.3.9.1 FFT Clock Control Register

FFT Clock Control Register (FFT_CLC)

The FFT Clock Control Register, FFT_CLC, acts globally and allows the complete FFT Engine to be disabled to reduce power consumption when the FFT is not required. When the FFT Engine is disabled (DISS=1_B), only register accesses to the FFT Engine are permitted. All other accesses to the FFT address space will receive a transaction error.

FFT_CLC

FFT Clock Control Register (0000_H) **Reset Value: 0000 0002_H**

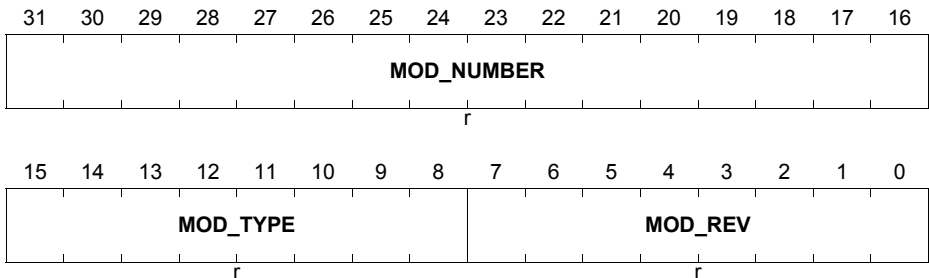


Field	Bits	Type	Description
DISR	0	rw	FFT Disable Request Bit Used for enable/disable control of the FFT Engine.
DISS	1	rh	FFT Disable Status Bit Bit indicates the current status of the FFT Engine. DISABLED after reset.
0	31:2	r	Reserved Read as 0; should be written with 0.

36.3.9.2 FFT Module Identification Register

FFT Identification Register (FFT_ID)

The FFT identification register allows the programmer version-tracking of the module. The table below shows the identification register which is implemented in the FFT Engine.

FFT_ID
FFT Identification Register
(0008_H)
Reset Value: 00E2 C001_H


Field	Bits	Type	Description
MOD_REV	7:0	r	Module Revision Number This bit field defines the module revision number. 01 _H : the value for max. 1K point engine 02 _H : the value for max. 2K point engine.
MOD_TYPE	15:8	r	Module Type The bit field is set to C0 _H which defines the module as a 32-bit module.
MOD_NUMBER	31:16	r	Module Number Value This bit field defines a module identification number. The value for the FFT Engine is 00E2 _H .

36.3.9.3 FFT Control and Status Register

FFT Control and Status Register (FFT_CSR)

This is the main register to control FFT/IFFT operation and provide feedback on the current state of the FFT Engine.

FFT_CSR
FFT Control and Status Register
(0040_H)
Reset Value: 0010 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											RFS	BUS Y	OUT _FM T	IN_FMT	
r											rh	rh	rw	rw	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		WIN _BY P	IFFT	LENGTH				0				STA RT			
r		rw	rw	rw				r				rwh			

Field	Bits	Type	Description
START	0	rwh	Start Transform Start a new transform - cleared by hardware.
LENGTH	11:8	rw	Length of Transform Log base 2 of the required transform size. Length must be between 3 (transform of 8) and 10 (transform of 1024).
IFFT	12	rw	Inverse FFT 0 _B perform FFT. 1 _B perform IFFT.
WIN_BYP	13	rw	Window Bypass 0 _B use window function 1 _B bypass window function
IN_FMT	17:16	rw	Input Format 0 _D 1 complex 32-bit datum per 64-bit word 1 _D 2 real 32-bit data per 64-bit word 2 _D 2 complex 16-bit data per 64-bit word 3 _D 4 real 16-bit data per 64-bit word
OUT_FMT	18	rw	Output Format 0 _D 1 complex 32-bit datum per 64-bit word 1 _D 2 complex 16-bit data per 64-bit word
BUSY	19	rh	FFT Engine Busy 0 _D : FFT Engine Idle - no pending transforms 1 _D : FFT Engine has at least one transform in operation.

Field	Bits	Type	Description
RFS	20	rh	Ready For Start 0_D : cannot accept a START operation 1_D : can accept a START operation.
0	31:21, 15:14, 7:1	r	Reserved Read as 0; should be written with 0

This register may be written at any time without setting the START bit, but this will have no effect on the FFT Engine. Only updates with the START bit set will change the FFT Engine state, and must be done before data can be loaded into the FFT Engine, to inform it of the input data formats and operational mode for that transform.

However the FFT Engine must be ready to accept a new START request. This register can be read and the RFS - ready for start - bit examined. If the RFS is true then a new transform can be started. A trigger will be sent to SRC_FFERR if START is set when the FFT Engine is not ready. The commencement of new transforms in an interrupt driven manner is described in [Section 36.3.8](#).

The IN_FMT field determines how data on the little-endian 64-bit data bus is loaded into the FFT Engine.

Table 36-12 Input Data Formats

IN_FMT	Halfword[3]	Halfword[2]	Halfword[1]	Halfword[0]
0	imag[0]		real[0]	
1	real[1]		real[0]	
2	imag[1]	real[1]	imag[0]	real[0]
3	real[3]	real[2]	real[1]	real[0]

The OUT_FMT field determines how data on the little-endian 64-bit data bus is unloaded from the FFT Engine.

Table 36-13 Output Data Formats

OUT_FMT	Halfword[3]	Halfword[2]	Halfword[1]	Halfword[0]
0	imag[0]		real[0]	
1	imag[1]	real[1]	imag[0]	real[0]

The BUSY bit reports the status of the FFT engine. If BUSY is set then at least one transform has either been requested with a START or is executing or is waiting for unloading

36.3.9.4 FFT History Registers

History registers allow the last few commands issued to the CSR to be retained for debug, the registers have no other effect upon the FFT Engine. They are reset to 0, and contain a non-zero value whenever they have been loaded by performing a write to the CSR register with START set. This write may or may not have setup a transform operation correctly depending upon the state of CSR.RFS at the time the START was issued.

FFT History Register 0 (FFT_HISTORY0)

The value of FFT CSR control fields when START was last issued.

FFT_HISTORY0

FFT History0 Register

(0060_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											RFS	BUS Y	OUT _FM T	IN_FMT	
r											rh	rh	rh	rh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	WIN _BY P	IFFT	LENGTH								0				
r	rh	rh	rh								r				

Field	Bits	Type	Description
LENGTH	11:8	rh	Length of Transform value of FFT_CSR.LENGTH when START command was last issued.
IFFT	12	rh	Inverse FFT value of FFT_CSR.IFFT when START command was last issued.
WIN_BYP	13	rh	Window Bypass value of FFT_CSR.WIN_BYP when START command was last issued.
IN_FMT	17:16	rh	Input Format value of FFT_CSR.IN_FMT when START command was last issued.

Field	Bits	Type	Description
OUT_FMT	18	rh	Output Format value of FFT_CSR.OUT_FMT when START command was last issued.
BUSY	19	rh	FFT Engine Busy value of FFT_CSR.BUSY when START command was last issued.
RFS	20	rh	Ready For Start value of FFT_CSR.RFS when START was last issued.
0	31:21, 15:14, 7:0	r	Reserved Always read as 0.

FFT History Register 1 (FFT_HISTORY1)

The value of FFT History Register 0 when START was last issued.

FFT_HISTORY1

FFT History1 Register

(0070_H)

Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0											RFS	BUS Y	OUT _FM T	IN_FMT	
r											rh	rh	rh	rh	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		WIN _BY P	IFFT	LENGTH				0							
r		rh	rh	rh				r							

Field	Bits	Type	Description
LENGTH	11:8	rh	Length of Transform value of FFT_HISTORY0.LENGTH when START command was last issued.
IFFT	12	rh	Inverse FFT value of FFT_HISTORY0.IFFT when START command was last issued.

Field	Bits	Type	Description
WIN_BYP	13	rh	Window Bypass value of FFT_HISTORY0.WIN_BYP when START command was last issued.
IN_FMT	17:16	rh	Input Format value of FFT_HISTORY0.IN_FMT when START command was last issued.
OUT_FMT	18	rh	Output Format value of FFT_HISTORY0.OUT_FMT when START command was last issued.
BUSY	19	rh	FFT Engine Busy value of FFT_HISTORY0.BUSY when START command was last issued.
RFS	20	rh	Ready For Start value of FFT_HISTORY0.RFS when START was last issued.
0	31:21, 15:14, 7:0	r	Reserved Always read as 0.

36.3.9.5 FFT Debug Control Registers

FFT OCDS Debug Access Register (FFT_ODA)

The FFT OCDS Debug Access (FFT_ODA) register is cleared by Debug Reset and by each System Reset when OCDS is disabled. It is not affected by System Reset when OCDS is enabled.

This register is used to control whether or not register bits which change state on read do so when read by the on-chip debug system.

Write access is 32 bit wide only and requires Supervisor Mode.

FFT_ODA
FFT OCDS Debug Access Register (00E4_H)
Reset Value: 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
0																	
r																	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	DRDIS	DDREN
0																	
r														rw	rw		

Field	Bits	Type	Description
DDREN	0	rw	Destructive Debug Read Enable <i>Note: For details see Table 36-14</i> 0 _B Destructive read for debug read accesses are disabled 1 _B Destructive read for debug read accesses are enabled
DRDIS	1	rw	Destructive Read Disable <i>Note: For details see Table 36-14</i> 0 _B Destructive reads for read accesses are allowed. 1 _B Destructive reads for read accesses are disabled
0	31:2	r	Reserved Read as 0; should be written with 0.

Table 36-14 Destructive Read Control encoding

DRDIS	DDREN	Description
0	0	Destructive read access are enabled for all masters except the OCDS master.
0	1	Destructive read access are enabled for all masters
1	-	Destructive read access are disabled for all masters

FFT OCDS Control and Status Register (FFT_OCS)

The FFT OCDS Control and Status (FFT_OCS) register is cleared by Debug Reset and by each System Reset when OCDS is disabled. It is not affected by System Reset when OCDS is enabled (signal eec_reset_n_i).

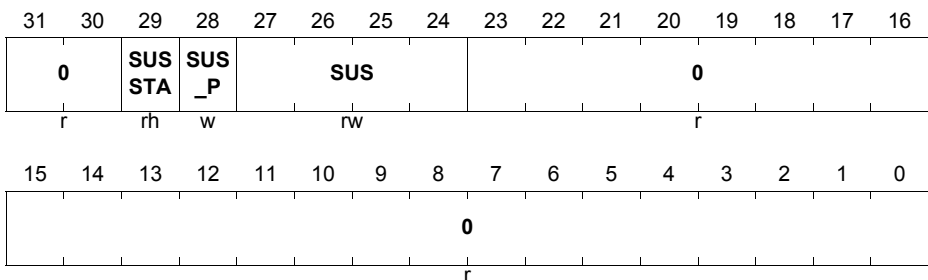
The FFT_OCS control register bits are only effective while the system is in debug mode. While not in debug mode, OCS reset values/modes are effective.

When a triggered by a high level on the suspend signal, the FFT/IFFT calculation will be halted as soon as practicable. Operation should resume when the suspend input is cleared.

Write access is 32 bit wide only and requires Supervisor Mode.

FFT_OCS

FFT OCDS Control and Status (00E8_H) **Reset Value: 0000 0000_H**



Field	Bits	Type	Description
SUS	27:24	rw	OCDS Suspend Control Controls the sensitivity to the suspend signal coming from the OCDS Trigger Switch (OTGS). 0 _H Will not suspend 1 _H Reserved for Hard suspend which is not implemented. 2 _H Soft suspend (Halt Mode) others , reserved
SUS_P	28	w	SUS Write Protection SUS is only written when SUS_P is 1, otherwise unchanged. Read as 0.
SUSSTA	29	rh	Suspend State 0 _B Module is not (yet) suspended 1 _B Module is suspended

Field	Bits	Type	Description
0	23:0, 31:30	r	Reserved Read as 0; must be written with 0.

36.3.9.6 FFT Kernel Reset Registers

FFT Kernel Reset Status Clear Register (FFT_KRSTCLR)

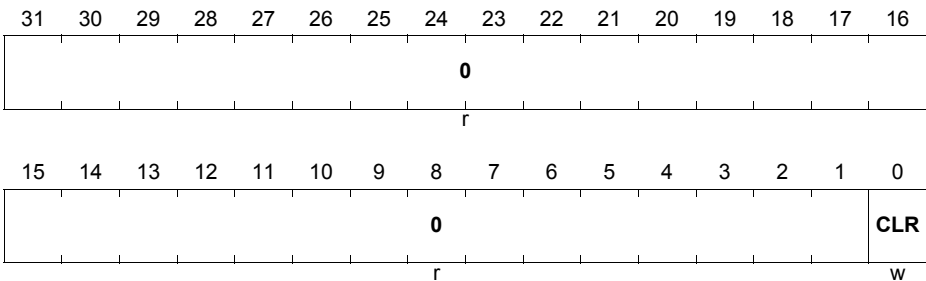
The FFT Kernel Reset Status Clear register is used to clear the Kernel Reset Status bit (KRST0.RSTSTAT).

Write access is 32 bit wide only and requires Supervisor Mode.

FFT_KRSTCLR

FFT Kernel Reset Status Clear Register(00EC_H)

Reset Value: 0000 0000_H

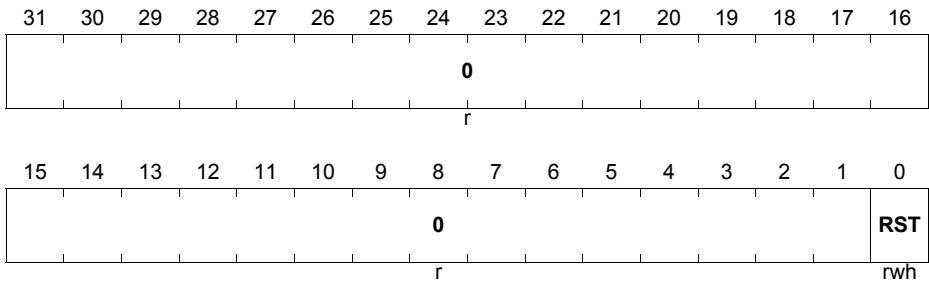


Field	Bits	Type	Description
CLR	0	w	Kernel Reset Status Clear <i>Note: Read always as 0_B.</i> 0 _B No action 1 _B Clear Kernel Reset Status KRST0.RSTSTAT
0	31:1	r	Reserved Read as 0 _B ; should be written with 0 _B .

FFT Kernel Reset Register 1 (FFT_KRST1)

The FFT Kernel Reset function is used to synchronously reset all the FFT Engine registers (except those connected to Debug reset) and the FFT Engine operation. To activate the kernel reset, it is necessary to set the RST bits by writing the value 1_B in both Kernel Reset Registers. The RST bit will be reset by hardware at the end of the kernel reset sequence.

Write access is 32 bit wide only and requires Supervisor Mode.

FFT_KRST1
FFT Kernel Reset Register 1
(00F0_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request a kernel reset. The kernel reset will be executed if the reset bits of both kernel reset registers are set. The RST bit will be cleared (re-set to 0 _B) after the kernel reset is executed. 0 _B No kernel reset was requested 1 _B A kernel reset was requested
0	31:1	r	Reserved Read as 0 _B ; should be written with 0 _B .

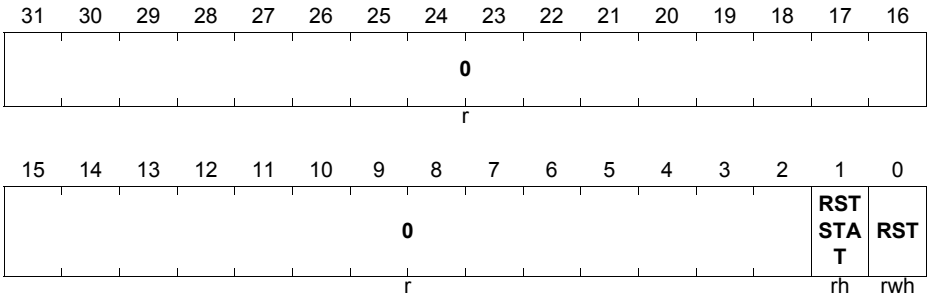
FFT Kernel Reset Register 0 (FFT KRST0)

The FFT Kernel Reset function is used to synchronously reset all the FFT Engine registers (except those connected to Debug Reset) and the FFT Engine operation. To activate the kernel reset, it is necessary to set the RST bits by writing the value 1_B in both Kernel Reset Registers. The RST bit will be reset by hardware at the end of the kernel reset sequence.

Kernel Reset Register 0 includes a kernel reset status bit that is set to 1_B in the same clock cycle the RST bit is reset. This bit can be used to detect that a kernel reset was processed. The bit can be re-set to '0' by writing '1' to the KRSTCLR.CLR register bit.

During the execution of the kernel reset; until RSTSTAT is set, all write accesses to the kernel registers will result in an error acknowledge.

Write access is 32 bit wide only and requires Supervisor Mode.

FFT_KRST0
FFT Kernel Reset Register 0
(00F4_H)
Reset Value: 0000 0000_H


Field	Bits	Type	Description
RST	0	rwh	Kernel Reset This reset bit can be used to request for a kernel reset. The kernel reset will be executed if the reset bits of both kernel registers are set. The RST bit will be cleared (reset to 0 _B) b after the kernel reset was executed. 0 _B No kernel reset was requested 1 _B A kernel reset was requested
RSTSTAT	1	rh	Kernel Reset Status This bit indicates whether a kernel reset was executed or not. This bit is set after the execution of a kernel reset in the same clock cycle both reset bits are cleared (FFT_KRST0.RST and FFT_KRST1.RST). This bit can be cleared by writing with '1' to the CLR bit in the related KRSTCLR register. 0 _B No kernel reset was executed 1 _B Kernel reset was executed
0	31:2	r	Reserved Read as 0; should be written with 0.

www.infineon.com

Published by Infineon Technologies AG